Hacking: The Basics

Zachary Wilson

Hacking: The Basics

Zachary Wilson
April 4, 2001

Updated by Martin Poulin, GSEC, GCWN, GCIH, CISSP
June 27, 2006

Preface

This paper contains information on the tools and skills a hacker uses to infiltrate computer systems and networks. If you are interested in learning more about how hackers work, and also how to protect your own computers and networks you should consider taking the SANS SEC 504 Hacker Techniques, Exploits and Incident Handling course.

Introduction

An intrusion can be defined as an attempt to break into or misuse a computer system. The word "misuse" is broad, and can mean something as severe as stealing confidential data, or something as minor as misusing your email system for spam. Today, both the Internet and corporate intranets are simply crawling with people from all walks of life who are continuously trying to test the security of various systems and networks. Some of these people are seeking some sort of intellectual high, while others are fueled by more treacherous motives such as revenge or stealing for profit. In any event, no intrusion is innocent and no intrusion is benign. There is no silver bullet available out there that will totally secure our networks and systems. The only thing we can do as IT professionals is to make sure that all of the doors are locked, that the alarm is turned on, and to educate ourselves on what to look for. The primary focus of this practical paper is to educate the less security-conscious IT professionals and end-users on exactly who is out there, and what they are doing to get in. By attempting to establish this baseline of security knowledge, we extend the arm of IT security to include those who present the greatest danger today: the uneducated user.

Attacker Profiles

There are two words to describe people who are trying to get into systems and networks: **hacker** and **attacker**. A hacker is a generic term for a person who likes getting into things. The benign hacker is the person who likes to get into his/her own computer and understand how it works. The malicious hacker is the person who likes getting into other people's systems. The benign hackers wish that the media would stop bad-mouthing all hackers and use the term 'attacker' instead. Unfortunately, this is not likely to happen. In any event, the word used to denote anybody trying to get into your system in this paper is 'attacker'. "Script Kiddie" is a term used to describe a class of attacker who does not have sophisticated technical knowledge, but rather simply has a collection of tools created by advanced hackers, and the basic knowledge to use these tools to perform an attack.

Attackers can be classified into two categories.

Insiders – these are attackers who have legitimate reasons to use/access your internal network. These include users who misuse privileges or who impersonate higher privileged users. According to a frequently quoted statistic, insiders commit 80% of security breaches. An insider is usually motivated by greed (cases of embezzlement or fraud) or revenge (disgruntled employees or former employees).

Outsiders – these attackers from outside your network attempt to attack your external presence by defacing web servers, forwarding spam through e-mail servers, etc. They may also attempt to go around the firewall to attack machines on the internal network. Outside attackers may come from the **Internet**, **wireless networks**, **dial-up lines**, **physical break-ins**, or from a **partner** (vendor, customer, reseller, etc.) network that is linked to your corporate network. They may be advanced attackers specifically targeting your corporate network for various reasons such as greed (e.g. credit card theft, corporate espionage) or "hacktivism" (defacement of public websites due to perceived social / political issues); or (and far more commonly) they may be Script Kiddies randomly attacking your systems based on the latest vulnerabilities.

Intrusion Techniques

These are the primary ways an attacker can get into a system:

Physical Intrusion – If an attacker has physical access to a machine (i.e. they can use the keyboard or take apart the system), they will be able to get in. Techniques range from special privileges granted at the console, to the ability to physically take apart the system and remove the disk drive (and read/write it on another machine).

System Intrusion – Also known as "Privilege Escalation", this type of hacking assumes the attacker already has a low-privilege user account on the system. If the system doesn't have the latest security patches, there is a good chance the attacker will be able to use a known exploit in order to gain additional administrative privileges.

Remote Intrusion – This type of hacking involves an attacker who attempts to penetrate a system remotely across the network. The attacker begins with no special privileges. There are several forms of this type of hacking. Note that Network Intrusion Detection Systems are primarily concerned with Remote Intrusion.

Possible Vulnerabilities and Ways to Exploit Them

Software bugs

Software always has bugs. System administrators and programmers can never track down and eliminate all possible software vulnerabilities, and attackers have only to find one hole in order to break in. Software bugs are often exploited in the server daemons, client applications,

operating systems, and the network stack. Software bugs can be classified in the following manner:

Buffer overflows - Almost all the security holes you read about are due to this problem. A typical example is a programmer who sets aside 256 characters to hold a login username. However, if an attacker tries to enter in a false username longer than that, you might have a problem. All the attacker has to do is send 300 characters, including code that will be executed by the server, and voila, game over. Hackers find these bugs in several ways. First, the source code for a lot of services is available on the net. Hackers routinely look through this code searching for programs that have buffer overflow problems. Secondly, hackers may look at the programs themselves to see if such a problem exists. Thirdly, hackers will examine every place the program has input and try to overflow it with random data. If the program crashes, there is a good chance that carefully constructed input will allow the attacker to gain access.

Unexpected combinations - Programs are usually constructed using many layers of code, including the underlying operating system as the bottom most layer. Attackers can often send input that is meaningless to one layer, but meaningful to another layer. A very common example of this is found in a SQL injection attack, where an attacker executes unauthorized SQL commands through an input field on a website by entering extra characters such as single-quotes (') or a semicolon (;) into the input field. The attacker can then append additional commands after these special characters, and they will be interpreted by the SQL server.

Unexpected input - Most programs are written to handle valid input. Most programmers do not consider what happens when somebody enters input that doesn't match the specification.

System Configuration Bugs

Default configurations - Most systems are shipped to customers with default, easy-to-use configurations. Unfortunately, "easy-to-use" means "easy-to-break-in". In recent years, operating system vendors have recognized this problem, and are starting to ship more secure systems out-of-the-box, but there will always be some risks associated with a default configuration.

Running unnecessary services - Virtually all programs can be configured to run in a non-secure mode. Sometimes administrators will inadvertently open a hole on a machine. Most administration guides will suggest that administrators turn off everything that doesn't absolutely, positively need to run on a machine in order to avoid accidental holes. Note that

security-auditing packages (such as Nessus) can usually find these holes and notify the administrator.

Trust relationships - Attackers often "island hop" through the network exploiting trust relationships. A network of machines trusting each other is only as secure as its weakest link.

Design flaws

Even if a software implementation is completely correct according to the design, there still may be bugs in the design itself that leads to intrusions.

TCP/IP protocol flaws - The TCP/IP protocol was designed before we had much experience with the wide-scale hacking we see today. As a result, there are a number of design flaws that lead to possible security problems. Some examples include smurf attacks, ICMP Unreachable disconnects, IP spoofing, and SYN floods. The biggest problem is that the IP protocol itself is very "trusting": hackers are free to forge and change IP data with impunity. IPsec (IP security) has been designed to overcome many of these flaws, but it is not yet widely used.

Poor system administrator practices - A surprising number of machines are configured with an empty or easy-to-guess root/administrator password, possibly because the administrator is too lazy to configure one right away and wants to get the machine up and running quickly with minimal fuss. Unfortunately, they never get around to fixing the password later, allowing attackers easy access. One of the first things an attacker will do on a network is to scan all machines for empty or commonly used passwords.

Password Cracking

Passwords are possibly the single weakest link in the security chain. Any system worth protecting should be protected by some form of multi-factor authentication scheme, such as smart cards, tokens, biometrics, or digital certificates. Passwords are simply too easily compromised to be relied upon as a single factor for authentication. However, implementing multi-factor authentication can be difficult, expensive, and some systems may not fully support it. For that reason, it is still important to understand the different methods of cracking or guessing passwords:

Easy-to-guess passwords - These are passwords where people use the names of themselves, their children, spouse, pet, or car model as their password. Then there are the users who choose "password", “administrator”, or simply blank passwords. An attacker will

almost always try these combinations first, before proceeding with any other password attacks.

Dictionary attacks - With this attack, the attacker will use a program that will try every possible word in the dictionary. Dictionary attacks can be done either by repeatedly logging into systems, or by collecting encrypted passwords and attempting to find a match by similarly encrypting all the passwords in the dictionary. Attackers usually have a copy of the English dictionary as well as foreign language dictionaries for this purpose. They all use additional dictionary-like databases, such as names (see above) and lists of common passwords.

Brute force attacks - Just as in a Dictionary attack, an attacker may try all possible combinations of characters. Using a single modern CPU, a short 4-letter password consisting of lower-case letters can be cracked in just a few minutes. A longer 8-character password consisting of upper and lower case letters, as well as numbers and punctuation can take several hours or more to crack. However, this time can be greatly reduced using distributed methods, where many computers work on the problem simultaneously.

Pre-computed tables – Popularly known as "Rainbow tables", this is essentially a brute-force attack where the work has been done ahead of time. Tables of all possible password hashes are pre-computed using the power of distributed computing. Once the tables have been generated, the amount of time to find a password of any strength is negligible -- even complex passwords can often be found within a matter of minutes. Popular tools such as Rainbow Crack, Ophcrack and Cain & Abel use pre-computed tables, and the tables themselves can easily be found online. Some sites will offer to crack password hashes for you (for a price), or sell the pre-computed tables.

The bottom line is that passwords are no longer an effective barrier against a determined attacker, and other methods of authentication should be implemented wherever possible.

Acquiring Passwords

Clear-text sniffing - A number of protocols (Telnet, FTP, HTTP Basic) use clear-text passwords, meaning that they are not encrypted as they go over the wire between the client and the server. An attacker with a protocol analyzer can watch the wire looking for such passwords. No further effort is needed; the attacker can start immediately using those passwords to log in.

Encrypted sniffing -Most protocols, however, use some sort of encryption on the passwords. In these cases, the attacker will need to carry out a Dictionary- or Brute Force-attack on the password in order to attempt decryption. Note that you still don't know about the attacker's presence, as he/she has been completely passive and has not transmitted anything on the wire.

Password cracking does not require anything to be sent on the wire since the attacker's own machine is being used to authenticate your password.

Replay attack - In some cases, attackers do not need to decrypt the password. They can use the encrypted form instead in order to log in to systems. This usually requires reprogramming their client software in order to make use of the encrypted password.

Password-file stealing - The entire user database is usually stored in a single file on the disk. In UNIX, this file is /etc/passwd (or some mirror of that file), and under Windows, this is the SAM file or the Active Directory database file, ntds.dit. Either way, once an attacker gets hold of this file, he/she can run cracking programs in order to find some weak passwords within the file.

Observation - One of the traditional problems in password security is that passwords must be long and difficult to guess (in order to make Dictionary- and Brute Force -cracks unreasonably difficult). However, such passwords are often difficult to remember, so users write them down somewhere. Attackers can often search a person's work site in order to find passwords written on little pieces of paper (usually under the keyboard). Attackers can also train themselves to watch typed-in passwords behind a user's back.

Social Engineering - One successful and common technique is to simply call the helpdesk and say "Hi, this is Ron Smith, the senior director for IT in San Jose. I have a presentation to give my boss, the CIO, and I can't log into server XYZ to get my notes. Would you please reset my password now? I have to be in this meeting in 2 minutes." Many unsuspecting operators would simply reset Ron's password in this situation. Most corporations have a policy that directs users/operators/helpdesk to never give out or reset passwords, even to their own IT director, but this technique is still successful. "Phishing" schemes also fall under this category. Phishing involves posing as a trusted source, usually through email, to trick users into revealing confidential information such as passwords or credit card numbers.

Keystroke logging – By recording a user's keystrokes, either with software installed on the workstation or with a piece of hardware that plugs in between the keyboard and the computer, an attacker can easily gather plenty of useful information, including passwords. Obviously connecting hardware can be more difficult since it requires physical access to the site, so software keystroke loggers are generally much more prevalent. One of the more interesting methods of getting the software installed is to leave a "candy dish" full of cheap USB pen drives near the entrance to a building, or to give them out in a public area. Once users connect these drives, a Trojan horse program is installed that records keystrokes and sends the data back to the attacker.

Typical Intrusion Scenarios

Reconnaissance - The attacker will find out as much as possible without actually giving himself away. He will do this by finding public information or appearing as a normal user. In this stage, you really can't detect an attacker. He will do a 'whois' look-up on your registered domain names to find as much information as possible about your network and people involved. The attacker might walk through your DNS tables (using 'nslookup', 'dig', or other utilities to do domain zone transfers) to find the names of your machines. The attacker will browse other public information, such as your public web sites and anonymous FTP sites. The attacker might search news articles and press releases about your company.

Scanning - The attacker uses more invasive techniques to scan for information, but still doesn't do anything harmful. He might walk through all your web pages and look for vulnerable CGI scripts. He might do a 'ping' sweep in order to see which machines are alive. He will run utilities like Nessus in order to see what's available and what is vulnerable. At this point, the attacker has done 'normal' activity on the network and has not done anything that can be classified as an intrusion. At this point, an NIDS may be able to tell you that "somebody is checking door handles", but nobody has actually tried to open a door yet.

Running exploits - The attacker crosses the line and starts exploiting possible holes in the target machines. A sophisticated attacker may user complex techniques to remotely exploit vulnerable services, whereas a less talented Script Kiddie might use automated tools such "Metasploit", which require little more knowledge than how to use a web browser and the IP address of a victim's system in order to exploit systems.

Establishing a foothold - At this stage, the attacker has successfully gained a foothold in your network by hacking into a machine. The attacker's main goal is to hide evidence of the attacks (doctoring the audit trail and log files) and to make sure he can get back in again. He may install 'rootkits' that give him access and hide his tracks, replace existing services with his own Trojan horses that have backdoor passwords, or create his own user accounts. A good host-based IDS can often detect an attacker at this point by noting the changed system files, but a good attacker can cover his tracks to avoid detection. The hacker will then use the system as a stepping-stone to other systems, since most networks have fewer defenses from inside attacks.

Playing for profit - The attacker finally takes advantage of his status to steal confidential data, misuse system resources (i.e. stage attacks at other sites from your site), or deface web pages.

Common Intrusion Signatures

There are three types of attacks:

Reconnaissance - These attacks include ping sweeps, DNS zone transfers, e-mail recons, TCP or UDP port scans, and possibly indexing of public web servers to find CGI holes.

Exploits – Attackers will take advantage of hidden features or bugs to gain access to the system.

Denial-of-service (DoS) attacks – Where the attacker attempts to crash a service (or the machine), overload network links, overload the CPU, or fill up the disk. The attacker is not trying to gain information, but to simply act as a vandal to prevent you from making use of your machine.

Common Reconnaissance Scans

One of the most common tools used in reconnaissance is Nessus. This free tool, which runs on many platforms including Linux and Windows, combines many different methods to identify remote systems and vulnerabilities. Some of the most common scan types include:

Ping sweeps – This simple scan simply pings a range of IP addresses to find which machines are alive. Note that more sophisticated scanners will use other protocols (such as an SNMP sweep) to do the same thing.

TCP scans – Probes for open (listening) TCP ports looking for services the attacker can exploit. Scans can use normal TCP connections or stealth scans that use half-open connections (to prevent them from being logged) or FIN scans (never opens a port, but tests if someone is listening). Scans can be sequential, randomized, or configured lists of ports.

UDP scans – These scans are a little bit more difficult because UDP is a connectionless protocol. The technique is to send garbage UDP packets to the desired port. Most machines will respond with an ICMP "destination port unreachable" message, indicating that no service is listening at that port. However, many machines throttle ICMP messages, so you can't do this very fast.

OS identification – By sending illegal (or strange) ICMP or TCP packets, an attacker can identify the operating system. Standards usually state how machines should respond to legal packets, so machines tend to be uniform in their response to valid input. However, standards omit (usually intentionally) the response to invalid input. Thus, each operating system's unique responses to invalid inputs form a signature that attackers can use to figure out what the target machine is. This type of activity occurs at a low level (like stealth TCP scans) that systems do not log.

Common Exploits

Once vulnerable systems have been discovered, the attacker will want to exploit the vulnerabilities. An advanced attacker may use sophisticated techniques to break into a system, while a less talented attacker might use popular automated tools like Metasploit. Internet

worms will also use common exploits to compromise a system in order to continue spreading. Common exploits include:

Server-side scripts – CGI programs and ASP scripts are notoriously insecure. Typical security holes include passing tainted input directly to the command shell via the use of shell metacharacters, using hidden variables specifying any filename on the system, and otherwise revealing more about the system than is good. These attacks can be detected by examining web server logs, and by monitoring network traffic with a NIDS.

Web server attacks – Beyond the execution of server-side scripts, web servers have other possible holes. The most common bugs are buffer overflows in the request field or in one of the other HTTP fields. Servers have long had problems with URLs. For example, the "death by a thousand slashes" problem in older Apache would cause huge CPU loads as it tried to process each directory in a thousand slash URL. New vulnerabilities in web servers are constantly being discovered. The best way to protect your web server is to keep up with the latest security patches from the vendor.

Web browser attacks – new security holes are constantly being discovered in all web browsers. This includes URL, JavaScript, CSS, Java, and ActiveX attacks.

URL fields can cause a buffer overflow condition, either as it is parsed in the HTTP header, as it is displayed on the screen, or processed in some form (such as saved in the cache history). Also, an old bug with Internet Explorer allowed interaction with a bug whereby the browser would execute .LNK or .URL commands.

JavaScript is a perennial favorite, and usually tries to exploit the "file upload" function by generating a filename and automatically hiding the "SUBMIT" button. Many variations of this bug have been fixed, only to have new ways found to circumvent the fixes.

CSS (cross-site scripting) attacks involve directing users to a malicious site via a specially crafted hyperlink. Vulnerable web browsers don't check for these types of hyperlinks, and confidential information can be passed to an attacker's own web server. This type of attack is most commonly used in a Phishing scheme, whereby users are tricked into visiting a malicious site.

Java has a robust security model, but that model has proven to have the occasional bug (though compared to everything else, it has proven to be one of the most secure elements of the whole system). Moreover, its robust security may be its undoing: normal Java applets have no access to the local system, but sometimes they would be more useful if they did have local access -- thus, the implementation of "trust" models that can more easily be hacked.

ActiveX is even more dangerous than Java since it works purely from a trust model and runs native code. You can even inadvertently catch a virus that was accidentally embedded in some vendor's code.

TCP sequence number prediction – in the startup of a TCP connection, you must choose a sequence number for your end, and the server must choose a sequence number for its end. Older TCP stacks choose predictable sequence numbers, allowing attackers to create TCP connections from a forged IP address (for which they will never see the response packets) that presumably will bypass security.

DNS poisoning through sequence prediction – DNS servers will "recursively" resolve DNS names. Thus, the DNS server that satisfies a client request will become itself a client to the next server in the recursive chain. The sequence numbers it uses are predictable. Thus, an attacker can send a request to the DNS server and a response to the server forged to be from the next server in the chain. It will then believe the forged response, and use that to satisfy other clients.

IP spoofing – There is a range of attacks that take advantage of the ability to forge (or 'spoof') your IP address. While a source address is sent along with every IP packet, it isn't actually used for routing. This means an attacker can pretend to be you when talking to a server. The attacker never sees the response packets (although your machine does, but throws them away because they don't match any requests you've sent). The attacker won't get data back this way, but can still send commands to the server pretending to be you.

IP spoofing is frequently used as part of Denial of Service attacks.

Common DoS (Denial of Service) attacks

Ping-of-Death – Sends an invalid fragment, which starts before the end of packet, but extends past the end of the packet.

SYN Flood – Sends TCP SYN packet (which start connections) very fast, leaving the victim waiting to complete a huge number of connections, causing it to run out of resources and dropping legitimate connections. "SYN cookies" are a new defense against this. Each side of a connection has its own sequence-number. In response to a SYN, the attacked machine creates a special sequence number that is a "cookie" of the connection, then forgets everything it knows about the connection. It can then recreate the forgotten information about the connection when the next packets come in from a legitimate connection.

Conclusion

As stated in the first paragraph, no computer or computer network is completely secure. There are new vulnerabilities found or created everyday. The only way we as IT professionals can rest easy when we go home at night is to know that we are employing a minimal amount of security today and working towards more security tomorrow. However, all the hard work and money spent on the best security tools available doesn't do any good if users don't do minimal things such as securing passwords and locking down workstations when they leave at night. It is therefore our responsibility as IT security professionals to educate these users to the best of our ability, thus ensuring that IT security is being used even when nobody is watching. Hopefully, this paper will make less educated users think about everything they do, and impress upon them the need to consider security in their everyday practices.

If you are interested in learning more about how hackers work, and also how to protect your own computers and networks you should consider taking the SANS SEC 504 Hacker Techniques, Exploits and Incident Handling course.

Works Consulted

Computer Incident Advisory Committee (CIAC) (1995). Advisory Notice F-08 Internet Spoofing and Hijacked Session Attacks.
[On-line], Available: http://www.ciac.org/ciac/bulletins/f-08.shtml

Pethia, Richard. "Removing Roadblocks to Cyber Defense." 3/28/2000.
URL: http://www.cert.org/congressional_testimony/Pethia_testimony_Mar28-2000.html

CERT Incident Note 99-07. Distributed Denial of Service Tools. Nov 18, 1999.
URL: http://www.cert.org/incident_notes/IN-99-07.html

Schneier, Bruce. "Security is not a product, it is a process". Crypto-Gram. 15 Dec 1999.
URL: http://www.counterpane.com/crypto-gram-9912.html

Back, Eliot. "Cracking Windows Passwords with Ophcrack and Rainbow Tables". 26 Apr 2006.
URL: http://elliottback.com/wp/archives/2006/04/26/cracking-windows-passwords-with-ophcrack-and-rainbow-tables/

Mansukhani, Amesh. "Are Smart Cards the New Way of Life? Solving the Password Problem" April 10, 2006

URL:
http://www.microsoft.com/technet/community/columns/secmgmt/sm0406.mspx

Piscitello, David M. "Anatomy of a Cross-Site Scripting Attack"
URL: http://www.watchguard.com/infocenter/editorial/135142.asp

Last Updated: April 18th, 2019

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Northern Virginia- Alexandria 2019	Alexandria, VAUS	Apr 23, 2019 - Apr 28, 2019	Live Event
SANS Muscat April 2019	Muscat, OM	Apr 27, 2019 - May 02, 2019	Live Event
SANS Pen Test Austin 2019	Austin, TXUS	Apr 29, 2019 - May 04, 2019	Live Event
Cloud Security Summit & Training 2019	San Jose, CAUS	Apr 29, 2019 - May 06, 2019	Live Event
SANS Bucharest May 2019	Bucharest, RO	May 06, 2019 - May 11, 2019	Live Event
SANS Security West 2019	San Diego, CAUS	May 09, 2019 - May 16, 2019	Live Event
SANS Stockholm May 2019	Stockholm, SE	May 13, 2019 - May 18, 2019	Live Event
SANS Dublin May 2019	Dublin, IE	May 13, 2019 - May 18, 2019	Live Event
SANS Perth 2019	Perth, AU	May 13, 2019 - May 18, 2019	Live Event
SANS Milan May 2019	Milan, IT	May 13, 2019 - May 18, 2019	Live Event
SANS Northern VA Spring- Reston 2019	Reston, VAUS	May 19, 2019 - May 24, 2019	Live Event
SANS New Orleans 2019	New Orleans, LAUS	May 19, 2019 - May 24, 2019	Live Event
SANS MGT516 Beta Two 2019	San Francisco, CAUS	May 20, 2019 - May 24, 2019	Live Event
SANS Amsterdam May 2019	Amsterdam, NL	May 20, 2019 - May 25, 2019	Live Event
SANS Autumn Sydney 2019	Sydney, AU	May 20, 2019 - May 25, 2019	Live Event
SANS Hong Kong 2019	Hong Kong, HK	May 20, 2019 - May 25, 2019	Live Event
SANS Krakow May 2019	Krakow, PL	May 27, 2019 - Jun 01, 2019	Live Event
SANS San Antonio 2019	San Antonio, TXUS	May 28, 2019 - Jun 02, 2019	Live Event
SANS Atlanta 2019	Atlanta, GAUS	May 28, 2019 - Jun 02, 2019	Live Event
Security Writing NYC: SEC402 Beta 2	New York, NYUS	Jun 01, 2019 - Jun 02, 2019	Live Event
SANS London June 2019	London, GB	Jun 03, 2019 - Jun 08, 2019	Live Event
SANS Zurich June 2019	Zurich, CH	Jun 03, 2019 - Jun 08, 2019	Live Event
Enterprise Defense Summit & Training 2019	Redondo Beach, CAUS	Jun 03, 2019 - Jun 10, 2019	Live Event
SANS Kansas City 2019	Kansas City, MOUS	Jun 10, 2019 - Jun 15, 2019	Live Event
SANS SEC440 Oslo June 2019	Oslo, NO	Jun 11, 2019 - Jun 12, 2019	Live Event
SANSFIRE 2019	Washington, DCUS	Jun 15, 2019 - Jun 22, 2019	Live Event
SANS Cyber Defence Canberra 2019	Canberra, AU	Jun 24, 2019 - Jul 13, 2019	Live Event
SANS ICS Europe 2019	Munich, DE	Jun 24, 2019 - Jun 29, 2019	Live Event
Security Operations Summit & Training 2019	New Orleans, LAUS	Jun 24, 2019 - Jul 01, 2019	Live Event
SANS Paris July 2019	Paris, FR	Jul 01, 2019 - Jul 06, 2019	Live Event
SANS Cyber Defence Japan 2019	Tokyo, JP	Jul 01, 2019 - Jul 13, 2019	Live Event
SANS Munich July 2019	Munich, DE	Jul 01, 2019 - Jul 06, 2019	Live Event
SANS FOR585 Madrid April 2019 (in Spanish)	OnlineES	Apr 22, 2019 - Apr 27, 2019	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced