Security vulnerabilities - basic insights

Software vulnerabilities can be interpreted as a malfunction or weakness in software or operating systems. With the development of new attack technologies, the severity of software vulnerabilities is growing exponentially.

Software vulnerabilities can be interpreted as a malfunction or weakness in software or operating systems.With the development of new attack technologies, the severity of software vulnerabilities is growing exponentially.Of course, all systems contain vulnerabilities, but the problem lies in whether these vulnerabilities are exploited and causing damage.

Network security disasters caused by software vulnerabilities are often explained by three ideal factors.That is:

1. Existence: That is the existence of a flaw in software.
2. Access: It is likely that hackers can access a security hole.
3. Exploitation: It is the ability that hackers can leverage and profit from that vulnerability through tools or with certain techniques.

Today, many organizations are witnessing holes in their systems being exploited.For example, here is a table showing the top 10 software products with the most security vulnerabilities in 2016 referenced by the CVSS system:

Numerical order

Software name

Developers

Number of vulnerabilities

first

Android

Google

523

2

Debian Linux

Debian

327

3

Ubuntu Linux

Ubuntu

278

4

Flash Player

Adobe

266

5

Leap

Novell

260

6

Opensuse

Novell

228

7

Acrobat Reader Dc

Adobe

277

8

Acrobat Dc

Adobe

277

9

Acrobat

Adobe

224

ten

Linux Kernel

Linux

217

Software error

Even simple mid-range software, which only serves a few specialized tasks, is made up of a large amount of code. The software structure is designed by humans, and the code in it is also written by humans, so the occurrence of errors is inevitable. In most cases, if a software is professionally produced - these errors cannot have too big an impact, especially in terms of security. At the most we will see some functions that are inactive, sometimes the software *'hangs'* while working or working slowly .

But saying that doesn't mean serious errors related to security can't happen. To be more specific, it is software bugs that outsiders can exploit to change the way software operates, adding self-written code, viewing data that the software manages. . In addition to subjective reasons such as careless use of users (click on strange links, download malware), these errors are one of the main slits that hackers often focus on. cascade to infiltrate machinery systems - from servers to the end user's personal computers. If this vulnerability belongs to an unpopular software that only serves a few simple tasks and does not play an important role in the system, then the security risk is still there but not serious. But the more complex and bulky the software system is, the more obvious it is to control the occurrence of these errors - no matter how highly qualified design engineers. And it is these software that often play a key role, as well as affect many corners of the system. By creeping through the gaps created by these software's bugs, bad guys can make certain changes to a user's device, or gain control of and access sensitive information.

5 important security holes and ways to attack

To build safe software, it is indispensable to understand software vulnerabilities.Here, we will learn a brief overview of important and dangerous security vulnerabilities.

SQL injection

SQL injection vulnerabilities provide an opportunity for hackers to insert malicious code into an SQL statement.SQL Injection is one of the types of web hacking by injecting SQL query / commandcodesinto input before transferring to the web application, you can login without a username and password, remote execution (remote execution), dump data and retrieve the root of SQL server .An attack tool is any web browser, such as Internet Explorer, Netscape, Lynx .

Position in the vulnerability listing (CWE)

Rating

ID

Name

first

CWE-89

'Failure to maintain the SQL query structure (also known as SQL injection)'

The programming languages ??are affected

Any coding language that can be used directly with SQL databases is vulnerable to this type of attack.Naturally, here are some of the most popular languages:

1. High-level languages: Perl, Ruby, Python, Java, VB.Net, SQL
2. Server Page: ASP, JSP, ASP.NET, PHP
3. Lower-level languages: C, C ++

OS Command Injection

The OS Command Injection vulnerability occurs when user-managed data integration software in a command, these data are processed in the command interpreter.If the data is not checked, a hacker can use shell super characters to change the command being executed.

Position in the vulnerability listing (CWE)

Rating

ID

Name

2

CWE 78

'OS Command Injection'

Overflow Buffer

Buffer overflow is a famous security hole.It occurs when a program tries to load more data into the buffer, exceeding its allowable storage capacity. External data can corrupt programs, corrupt data, and even create conditions for executing malicious code.Languages ??like Java, Python, Visual Basic and C # include constraint check arrays and original string types.Therefore, buffer overflows are considered impossible in environments written in these languages.

Position in the vulnerability listing (CWE)

Rating

ID

Name

3

CWE-120

'Classic Buffer Overflow'

The following table shows the relevant items of this security hole in the CWE list:

ID

Name

CWE 121

'Stack-based Buffer Overflow'

CWE 122

'Heap-based Buffer Overflow'

CWE 123

'Write-what-where Condition'

CWE 124

'Boundary Beginning Violation'

CWE 125

'Out of bounds Read'

CWE 128

'Wrap around L?i'

CWE 129

'Unchecked Array Indexing'

CWE 131

'Incorrect Calculation of Buffer Size'

CWE 193

'Off by One Error'

CWE 466

'Return of Pointer Outside Value of Expected Range'

The programming languages ??are affected

1. Languages: C, Fortran, Assemblys.
2. Environment: Application servers, web servers and web applications.

Uncontrolled Format String

This vulnerability involves accepting unchecked or unauthorized input as a format string to execute a function.This weakness may lead to the execution of malicious code and may even damage the system.

Position in the vulnerability listing (CWE)

Rating

ID

Name

23

CWE 134

'Uncontrolled Format String'

The programming languages ??are affected

1. Direct Impact: C, C ++.
2. Indirect effects: Perl (if read in a fake data type).

Integer Overflow

The integer overflow vulnerability exists when a calculation tries to increase the integer value higher than the integer used to store in the related expression.When this error occurs, integer values ??can be converted to negative or very small numbers.This weakness becomes an important security issue when computational results are used to handle control loops, determine dimensions or perform tasks such as copying, allocating memory, pairing, etc. and make a decision.

Position in the vulnerability listing (CWE)

Rating

ID

Name

24

CWE 190

'Integer Wraparound or Overflow'

The following table shows the relevant items of this security hole in the CWE list:

ID

Name

CWE 682

'Incorrect Calculation'

CWE 191

'Integer Underflow'

CWE 192

'Coercion Error'

The programming languages ??are affected

Almost all languages ??are affected;however, the consequences will also vary depending on how the language handles integers.

1. Seriously affected: C, C ++

Zero-Day Exploits - Silence attack

In fact, vulnerabilities can be exploited for malicious purposes that exist on any software. There are even parts of the design that are hard to blame until there are technologies that allow outsiders to exploit it - making the author redesign how his products work. When updating new software, in addition to sometimes seeing new functions, or improved performance, many times you see the changelog (list of changes) appearing a series of fixes. Most recent error. Those who create a product must of course be the one who understands their best child - and will do his best to fix the error every time he finds out (at least in most cases). With products popular in the market, released by companies - organizations that operate professionally, this is even more true.

But nothing is absolute. There will be times when the author discovers an error after an outsider, or even is unable to detect it. It is not all of a sudden that big companies often organize contests about exploiting vulnerabilities in their products, and recruiting personnel from those competitions, as well as recruiting payroll hackers. The reality is always the same: there are talented people, some people are not. There will even be times when the manufacturer discovers the error, but the time to complete the repair is longer than the time the hackers need to write down the exploit tool, and complete the destructive work, good spying. Stealing with that tool. It is also one of the reasons why the articles about security vulnerabilities often appear only months after the error has been fixed. White-hat hackers understand that fixing errors is sometimes more difficult and complicated than taking advantage of errors for bad purposes, so they often give the manufacturer months to fix their mistakes first. when disclosing details about the flaw I found out for research purposes.

What's the worst scenario? The bad guys discovered the error . and of course not published it to anyone, silently closed the cultivation to complete the tool to exploit the error and quietly distribute it (most commonly in the form of a virus, worm, trojan .). Even criminals can bring this information into transactions, exchanging with each other implicitly, or selling in the kit written specifically for the purpose of exploring and exploiting the vulnerability. The manufacturer does not know the existence of the vulnerability, let alone fix it. Only when the consequences have been touched before, can they help the fire to find ways to overcome, compensate users, like the incident of Sony the day before. It was also because the attack was done when the manufacturer was completely unaware of the existence of these vulnerabilities, with *"0 days"* to find a way to fix the problem that

"zero-day" was born.

In summary, the fact that a software error exists is not something too strange, the danger only appears when the manufacturer loses in both races: error detection and error correction.

Mining process

Understand that today's modern security tools such as firewalls, anti-virus software, anti-malware . often have a clever mechanism to detect when a certain code has suspicious behavior. , regardless of whether the code is available in the database of viruses, malware or not. Just like a seasoned scout can detect suspicious signs of a thief without a wanted warrant or a big *'thief'* in front of his forehead. However, as mentioned, the worst case scenario is when hackers discover an unknown error, write a completely new tool to exploit. One person if he can afford to finish first in both races (here not to mention those who use the tool again) must certainly have the experience of avoiding the scrutiny of security tools. . So until the vulnerability is completely patched, all the measures that security tools provide are temporary. Typical sequence of events is usually as follows:

1. Appear a vulnerability that can be exploited by existing technologies.

2. Attackers detect vulnerabilities.

3. This person immediately wrote and distributed tools to exploit this vulnerability.

4. Manufacturer simultaneously detects an error and immediately seeks to fix it.

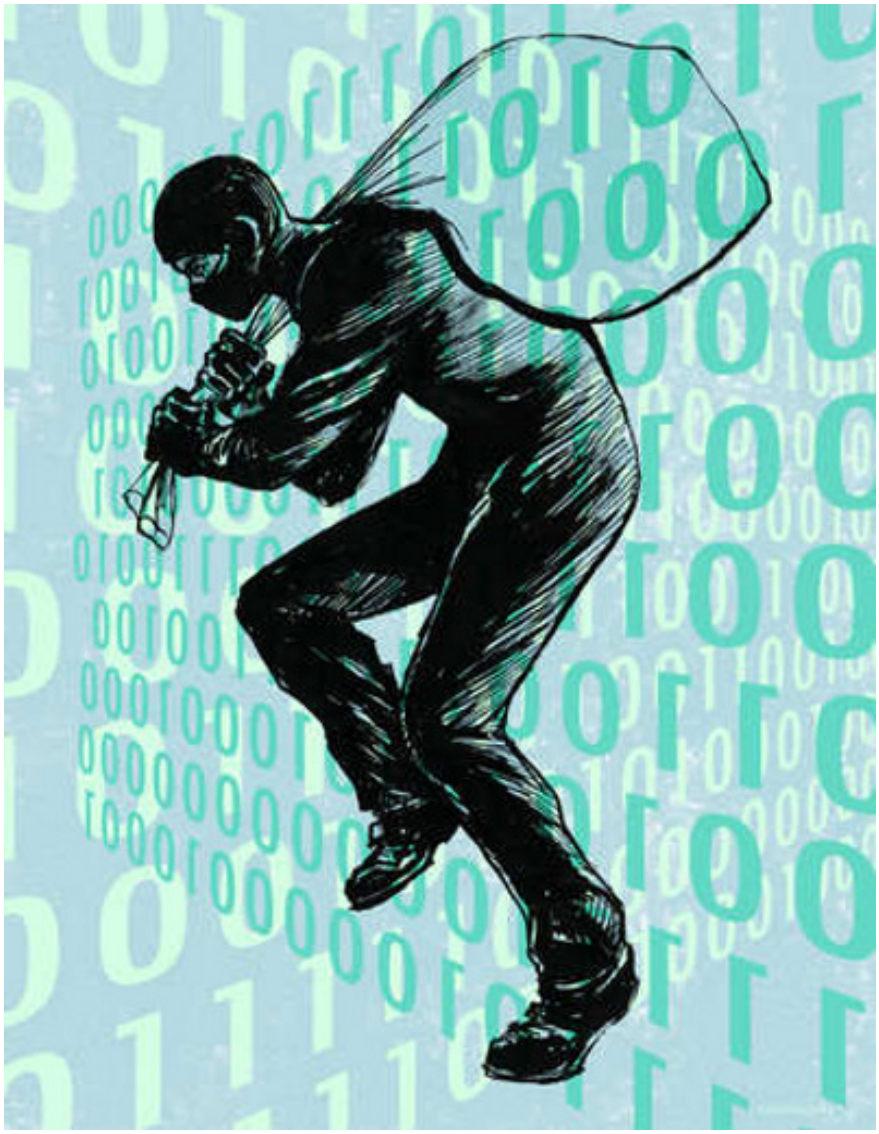
5. The vulnerability is announced out.

6. Anti-virus software is updated with information to detect when there are codes trying to exploit this vulnerability.

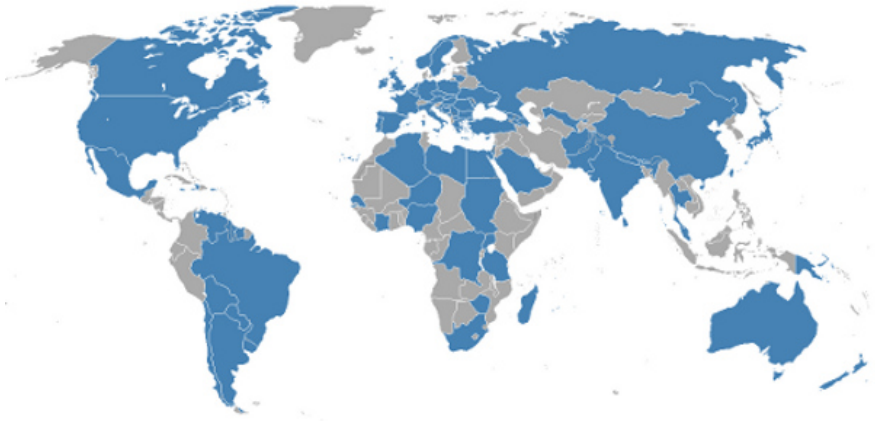
7. The manufacturer completes the patch.

8. The company has completed issuing patches to all customers.

The timing of the first attack is obviously between steps 3 and 5. According to a recent study by Carnegie Mellon University in the US, this period averages 10 months. However, not all end users are always at risk during this period. This type of attack that takes advantage of the time when the manufacturer has not detected (or has not fixed this error) has the greatest advantage of being discreet - suitable for stealing information or undermining undetected. So at this stage the target audience is usually a group of people who can bring specific benefits to the attacker so that he can then withdraw smoothly. That goal can be organizations or corporations who want to sabotage or account information that can be used to make a profit.



Also according to this study, the stage from step 5 to 8 is really dangerous. This is the time when information about the vulnerability is published, and along with anti-virus developers, hackers who are not aware of the bug can also access the information. The wave of attack is no longer silently, but much more intense. If for example the previous attack was as dangerous as a stabbing stab in the back, then this attack was like a direct attack, not effective for those who were cautious but still no less dangerous if Meet the right people who neglect security or miss using low quality security tools, slow updates. The objects who are unable to detect errors, as well as the inability to develop tools, also participate from this point, making the distribution and finding the weak security systems much faster. . As the number of attackers increased, the motives and methods of attack were more diverse, not merely pure and stealing.



After reading here, readers probably understand that when it comes to protecting their information and systems, in addition to updating defense measures, updating information is equally important. Often, serious errors of popular and important systems like Java will be published by the press as soon as the manufacturer announces. However, the more popular and *humble software* is often not favored. So in addition to paying attention to upgrading patches, it is necessary to stop using old software that is no longer cared for and fixed as soon as possible. For example? Microsoft kept screaming for XP and IE6 to rest .