**Threat Brief: Ongoing Russia and Ukraine Cyber Conflict**

**HelloXD Malware Overview**

HelloXD is a ransomware family first observed in the wild on Nov. 30, 2021. This ransomware family uses a modified ClamAV logo in their executables. ClamAV is an open-source antivirus engine used to detect malware. We also have observed additional samples with different versions of the logo, which led us to believe the ransomware developer may like using the ClamAV branding for their ransomware. Additionally, some of the observed samples include properties information that can be observed in Figure 1.

The file description included the entry VlahmAV, a play on words on ClamAV, and the developer named the ransomware HelloXD and used another potential alias, uKnow, as the developer of HelloXD in the copyright section.

When executed, HelloXD tries to disable shadow copies to inhibit system recovery before encrypting files using the following commands embedded in the sample:

HelloXD uses the commands shown to disable shadow copies

Additionally the ransomware does a ping to 1.1.1[.]1 and asks to wait a timeout of 3000 milliseconds between each reply, quickly followed with a delete command to delete the initial payload.

cmd.exe /C ping 1.1.1[.]1 -n 1 -w 3000 > Nul & Del /f /q

"C:\Users\admin\Desktop\xd.exe"

Two of the initial set of samples identified create a unique mutex containing the following message:

mutex: With best wishes And good intentions...

After those commands are executed, the ransomware finishes by appending the file extension .hello, alongside a ransom note titled Hello.txt (Figure 2).

The ransom note was modified between the observed samples. In the first sample we encountered (Figure 3, left), the ransom note only linked to a TOX ID, whereas a later observed sample (Figure 3, right) links to an onion domain as well as a TOX ID (different from the one in the first version). At the time of writing, this site is down.

The ransomware creates an ID for the victim which has to be sent to the threat actor to make it possible to identify the victim and provide a decryptor. The ransom note also instructs victims to download Tox and provides a Tox Chat ID to reach the threat actor. Tox is a peer-to-peer instant messaging protocol that offers end-to-end encryption and has been observed being used by other ransomware groups for negotiations. For example, LockBit 2.0 leverages Tox Chat for threat actor communications.

When observing both variants executing on virtual environments, we noted that the more recent variants changed the background to a ghost – a theme we’ve noticed in this threat actor’s work since our earliest observations of it. However, the previous version didn’t change the background at all – it simply left the ransom note we observed previously (Figure 4).

Packer Analyses

During analysis and threat intel gathering, we discovered two main packers used for HelloXD ransomware binaries, as well as for other malware samples linked to the potential author (Figure 5).

The first type of packer is a modified version of UPX. The code is extremely similar between UPX-packed binaries and the custom packer, however the custom packer avoids using identifiable section names such as .UPX0 and .UPX1, and leaves the default .text, .data, and .rsrc names unchanged. There are also no magic bytes within the packed payload, unlike UPX-packed binaries, which contain the magic bytes UPX!.

However, a dead giveaway that the sample is packed is the raw size of the .text section, which is zeroed out, while the virtual size is much larger, as expected; this is identical to a .UPX0 section. As there is no data within the .text section on disk, the entry point of the unpacking stub is within the .data section, which will unpack the malicious code into the .text section on runtime.

All of these details point toward the threat actor having modified or copied certain elements from the UPX packer, which can be further confirmed by comparing a UPX-packed binary with a custom-packed HelloXD binary.

The second packer we discovered consists of two layers, with the second being the custom UPX packer discussed above. This particular packer seems to be more common on x64 binaries and involves decrypting embedded blobs using a seemingly custom algorithm containing unconventional instructions such as XLAT (Figure 6).

Aside from storing the encrypted second layer, there is little to no obfuscation within the packer; API calls such as VirtualAlloc and VirtualProtect are clearly visible, and there is no control flow obfuscation.

We have observed two different samples of the HelloXD ransomware publicly available, indicating it is still under development by the author(s). The first sample is fairly rudimentary, with minimal obfuscation and typically paired with an obfuscated loader responsible for decrypting it through the use of the WinCrypt API before injecting it into memory. The second sample, on the other hand, is far more obfuscated, and is executed in memory by a packer rather than a full-scale loader.

While the obfuscation and execution may differ between the two, both samples contain very similar core functionality, due to the author copying the leaked Babuk/Babyk source code in order to develop the HelloXD ransomware (Figure 7). As a result, a lot of the function structure overlaps with Babuk, after getting past the obfuscation.

While there is a lot of overlap between HelloXD and Babuk, there are some small but crucial differences to take note of between Babuk and the two different variants.

HelloXD version 1 is the least modified, utilizing Curve25519-Donna and a modified HC-128 algorithm to encrypt file data, while also containing the same CRC hashing routine incorporating the string dong, possibly referencing Chuong Dong, who had previously analyzed and reported on the first version of Babuk (Figure 8).

The HelloXD author(s) did modify the infamous file marker and mutex however, opting for dxunmgqehhehyrhtxywuhwrvzxqrcblo as the file marker and With best wishes And good intentions… as the mutex (Figure 9).

With HelloXD version 2, the author(s) opted to alter the encryption routine, swapping out the modified HC-128 algorithm with the Rabbit symmetric cipher. Additionally, the file marker changed again, this time to what seems to be random bytes rather than a coherent string. The mutex is also modified, set to nqldslhumipyuzjnatqucmuycqkxjon in one of the samples (Figure 10).

Both versions have been compiled with the same compiler (believed to be GCC 3.x and above based on the mangling of export names), resulting in very similar exports between not only the ransomware variants, but also other malware that we have linked to the potential author (Figure 11).

The biggest difference between the versions is the interesting addition of a secondary payload embedded within version 2. This payload is encrypted using the WinCrypt API, in the same fashion as the obfuscated loader discussed above. Once decrypted, the payload is dropped to System32 with the name userlogin.exe before a service is created that points to it. userlogin.exe is then executed (Figure 12).

What is peculiar about this file is it is a variant of the open-source MicroBackdoor, a backdoor allowing an attacker to browse the file system, upload and download files, execute commands and remove itself from the system (Figure 13). As the threat actor would normally have a foothold into the network prior to ransomware deployment, it raises the question of why this backdoor is part of the ransomware execution. One possibility is that it is used to monitor ransomed systems for blue team and incident response (IR) activity, though even in that case it is unusual to see offensive tools dropped at this point in the infection.

WTFBBQ Pivots

While analyzing the ransomware binaries, we discovered a unique string prevalent in almost all of the samples: :wtfbbq (stored as UTF-16LE). Querying VirusTotal with this string led to the discovery of eight files, six of which could be directly attributed to x4k through their own VirusTotal graph mapping out their infrastructure. The discovered samples are primarily Cobalt Strike Beacons, utilizing heavy control flow obfuscation – unlike the HelloXD ransomware samples we had previously seen.

Unfortunately, this specific string is not completely unique to x4k, and is instead found on several GitHub repositories as part of a technique to allow a running executable to delete itself from disk through renaming primary data streams within the file to :wtfbbq. Running a search for the non-UTF-16LE string results in multiple files, and filtering for executables yields 10 results, the majority of which are NIM-based binaries – potentially linked to this GitHub repository.

While the :wtfbbq string is not unique to x4k, by searching for the UTF-16LE version found inside the analyzed HelloXD ransomware samples, we only came across binaries linked to x4k’s infrastructure, providing a fairly strong link between HelloXD and x4k.

Hunting for Ransomware Attribution

The backdoor provided extremely useful insight into the potential threat actor behind the ransomware, as it had the following hardcoded IP address to use as the command and control (C2): 193[.]242[.]145[.]158. Upon navigating to this IP address, we observed an email address – tebya@poime[.]li – on the page title, the first link in the chain of attribution (Figure 14).

Using the email address as a pivot, we identified additional domains that were linked to tebya@poime[.]li.

Some of them historically resolved to some malicious IPs, which led us to discover additional infrastructure and malware being hosted in other domains (Table 3). Many of these also use the x4k name in the domain.

When looking at this infrastructure on VirusTotal, we observed that some of the domains we found were part of a VirusTotal graph called a.y.e/ created by the user x4k on June 30, 2021. On this graph, we found his infrastructure mapped out and malicious files that were also linked to the domains. This, however, was not the only graph we observed x4k creating. We also encountered additional graphs, mapping different things such as “Russian Hosts,” “DDoS Guard” and others, dating back to August 10, 2020 (Figure 15).

Additionally, we observed the initial email being linked to a GitHub account (Figure 16), as well as various forums including XSS, a known Russian-speaking hacking forum created to share knowledge about exploits, vulnerabilities, malware and network penetration.

From the GitHub page, we also observed a URL to a site – xn--90a5ai[.]com(фсб[.]com) – resolving to the previously mentioned IP 164[.]68[.]114[.]29, which at this point in time only shows an animation of interconnecting points. That being said, when looking at the HTML source code of the site, we discovered a couple of references to the user observed before – x4kme – and other aliases such as uKn0wn, which was observed in the HelloXD ransomware samples.

From the list of aliases used by the threat actor, we were able to observe another GitHub account with the name l4ckyguy, sharing the profile picture, location and URL in the description, with a link to the previously observed account (x4kme), and a name, Ivan Topor, which we believe may be another alias for this threat actor. A further account, l4cky-control, was also discovered. This repository contained a Python script that would decrypt a secondary Python script which reached out to the IP 167[.]86[.]87[.]27 to download and execute another Python script. This particular IP was linked to a Contabo server that x4k had also included within their VirusTotal graph discussed above.

We also found a YouTube account linked to the actor through the initial email tebya@poime[.]li, using another alias, Vanya Topor. It’s worth pointing out that “Vanya” is a diminutive for Ivan.

The YouTube account has no public videos, but we observed this threat actor sharing unlisted links in various hacking forums. The content of the videos were tutorials and walkthroughs, where the threat actor showed his methodology of performing certain actions, depending on the video. The videos had no sound, but the threat actor would type commentary on a terminal to address something the viewer was observing on screen.

The videos found gave us insight into x4k operations before moving into ransomware activity specifically. We learned how this threat actor leverages Cobalt Strike for his operations, including how to set up Beacons as well as how to send files to compromised systems. In one of the videos, we actually observed the threat actor performing a DNS leak test on his Android phone. We also got to observe how the domain фсб[.]com used to look in October 2020 – a blog of sorts titled “Ghost in the Wire.” Where the threat actor keeps alluding to his “Ghost” theme, a similar theme was observed in the HelloXD ransomware samples (Figure 19).

In another video instance, we observed the threat actor submit a LockBit 2.0 sample on Cuckoo sandbox and compare the results with another presumably LockBit 2.0 sample prior to the one submitted in the video. At the time of writing, we don’t believe x4k is related to LockBit 2.0 activity, but we did find the choice of this particular ransomware family interesting (Figure 20). We also noticed this threat actor leveraging the use of other sandboxes besides Cuckoo – such as ANY.RUN and Hybrid Analysis – to test out verdicts and tooling, alongside the use of various virtual machines.

Additionally, this threat actor not only leverages open-source tooling, but also develops his own tools and scripts. We were able to see this threat actor demonstrating some of his tools performing automated actions in his videos, such as obfuscating files, creating executables and code signing (Figure 21).

Taking a closer look at x4k’s main OS, we believe it to be a customized Kali Linux instance, tailored to his needs. Most of his videos, comments, configurations and tutorials are written in Russian – and when combined with knowledge gained from a few OPSEC mistakes – Russia is also where we believe x4k originates from. Additionally, we encountered the ClamAV logo during one of the threat actor walkthrough videos – the same logo used on the HelloXD ransomware samples (Figure 22). This time around, x4k is using the logo as the start menu for his OS enviorment.

On the same taskbar, we also noticed the Telegram icon, which is a very popular messaging app for chatting – but is also used by threat actors such as LAPSUS$ to post news into specific channels. Using his username and alias and pivot, we were able to identify two Telegram accounts sharing the same picture as observed before, and descriptions pointing to the threat actor’s main site фсб[.]com. We noticed that the x4k Telegram account is used actively versus the old account – which, according to Telegram, hasn’t been active in a while (Figure 23).

x4k has a very solid online presence, which has enabled us to uncover much of his activity in these last two years. This threat actor has done little to hide malicious activity, and is probably going to continue this behavior.

Unit 42 research encountered HelloXD, a ransomware family in its initial stages – but already intending to impact organizations. While the ransomware functionality is nothing new, during our research, following the lines, we found out the ransomware is most likely developed by a threat actor named x4k. This threat actor is well known on various hacking forums, and seems to be of Russian origin. Unit 42 was able to uncover additional x4k activity being linked to malicious infrastructure, and additional malware besides the initial ransomware sample, going back to 2020.

Ransomware is a lucrative operation if done correctly. Unit 42 has observed ransom demands and average payments going up in the latest Ransomware Threat Report. Unit 42 believes that x4k, this threat actor, is now expanding into the ransomware business to capitalize on some of the gains other ransomware groups are making.