CYFIRMA
DECODING THREATS

INFOSTEALER PRYNT MALWARE
A DEEP DIVE INTO IT'S PROCESS INJECTION TECHNIQUE

Executive Summary

CYFIRMA Research team has seen an uptick in threat actor orchestrated cyber campaigns aimed at stealing confidential and sensitive information. Infostealers like “Prynt” are used to exfiltrate information as the first step leading into orchestration of sophisticated attacks which may include deployment of ransomwares.

In this report, we analyse infostealer “Prynt” which is often configured by the threat actors via a "builder" so that they can subsequently configure the malware easily. It may be noted that “Prynt” Stealer builder is used to create infostealer “Prynt” with a hidden backdoor functionality.

CYFIRMA Research team analysed an infostealer “Prynt” sample which was gathered from a public repository. The sample was found to be written in C/C++ and is a 32-bit console binary. Infostealer “Prynt” has the capability to steal system information from infected systems, which includes files from the targeted directories and credentials from web browsers.

In our analysis, we focus on the process injection technique leverage by the infostealer “Prynt” utilizing reverse engineering & memory forensics analysis, which involves injecting the infostealer “Prynt’s” malicious code into the legitimate AppLaunch.exe (Microsoft .NET ClickOnce Launch Utility) process; and later exits after injecting the code. Running the malicious code in the context of another process may allow access to the process's resources like - memory, system, and network.

The infostealer “Prynt” has been observed to have the following capabilities:

- Gathering System Information.
- Enumerating through files and processes.
- Hiding the processes.
- Injecting the code into PE files.
- Registry changes.
- Network communication through backdoor.
- Capture screenshots.

ETLM Attribution

Infostealer “Prynt” is a commodity malware and leveraged as part of Malware-as-a-Service (MaaS). In this case malware authors went ahead and wrapped up stealer with a backdoor even for their paying customers. The backdoor sends copies of victims' exfiltrated data gathered by other threat actors to a private Telegram chat monitored by the Prynt Stealer developers.

Supported by our analysis, the infostealer “Prynt” evades detection using the process injection technique by injecting itself into a legitimate “AppLaunch.exe” process – which is a common technique used by malware. The Prynt Stealer builder used to create the infostealer “Prynt” has been derived from open-source code bases like AsyncRAT and StormKitty (an information stealer).

Such infostealer campaigns are used to exfiltrate confidential and sensitive information to resell for financial gains in dedicated forums and Telegram channels. Threat actors are also known to use various ways to distribute stealers, including - social engineering, phishing emails, compromised websites, and embedded images.

As part of CYFIRMA tracked campaigns observed in the last 6 month, we can see infostealer “Prynt” has been leveraged by threat actors originating from the following geographical regions (Confidence Level: Low):

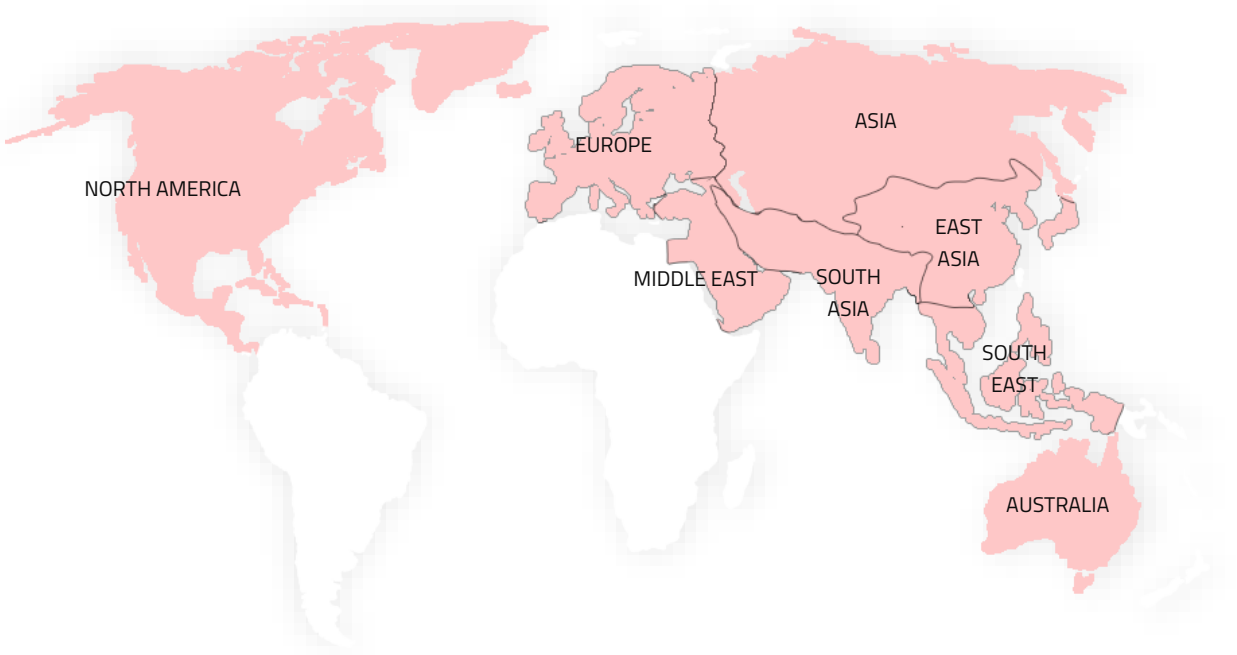
Threat Actor Demographics

Russian Speaking
Threat Actor Groups

Mandarin Speaking
Threat Actor Groups

Korean Speaking
Threat Actor Groups

Geographies Targeted Include (40+ Nations)



Industries Targeted Include

Multiline Retail	Health Care	Automotives	Government
Industrial Conglomerates	IT Services	Diversified Financial Services	Transpiration Infrastructure
Media & Entertainment	Oil & Gas	Real Estate	Food & Beverages
Hospitality	Construction	Technology	Household Product

What's Seen Brewing in Underground Forums

CYFIRMA researchers while monitoring cybercriminal activities in the underground forums, noticed that a new version of the Prynt Advanced Stealer 4.2.2 is available on underground forums.

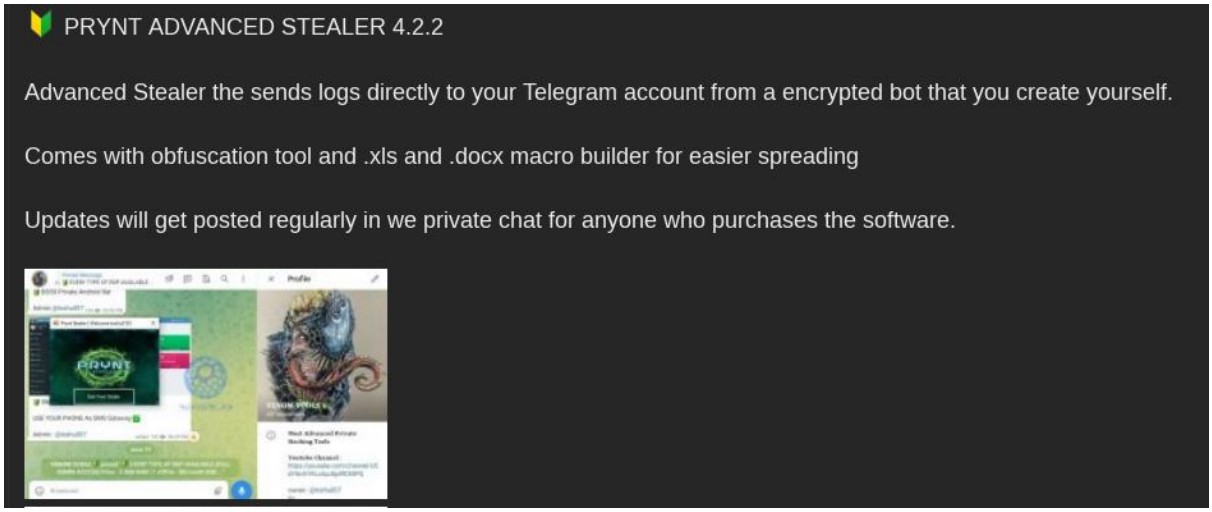


Figure 1: PRYNT ADVANCED STEALER 4.2.2

Following are the features available in the new version.

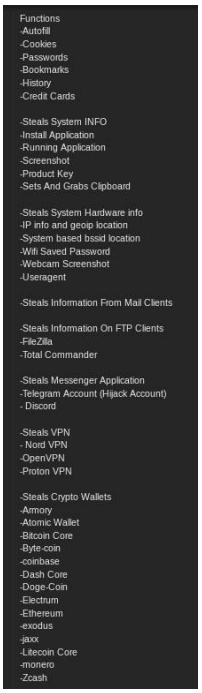


Figure 2: PRYNT ADVANCED STEALER 4.2.2 features

Prynt Stealer new version is targeting the following browsers to steal credentials.

Figure 3: PRYNT ADVANCED STEALER 4.2.2 Browser List

Static Analysis

File: Prynt.Exe

Subsystem: Console

MD5: Bcd1e2dc3740bf5eb616e8249d1e2d9c

SHA1: 230f401260805638aa683280b86af2231cf73f93

SHA256: 04b528fa40c858bf8d49e1c78f0d9dd7e3bc824d79614244f5f104baae628f8f

File Type: PE32 Executable (Console) Intel 80386, For MS Windows

File Type

File type of the Prynt.exe is a PE32 executable.

```
Prynt.exe: PE32 executable (console) Intel 80386, for MS Windows
```

Figure 4: File Type

Packing

As can be observed, Prynt.exe is not packed.

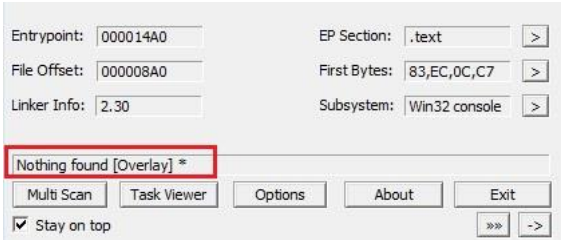


Figure 5: Packing Info

As can be observed, Prynt.exe is written in C/C++.

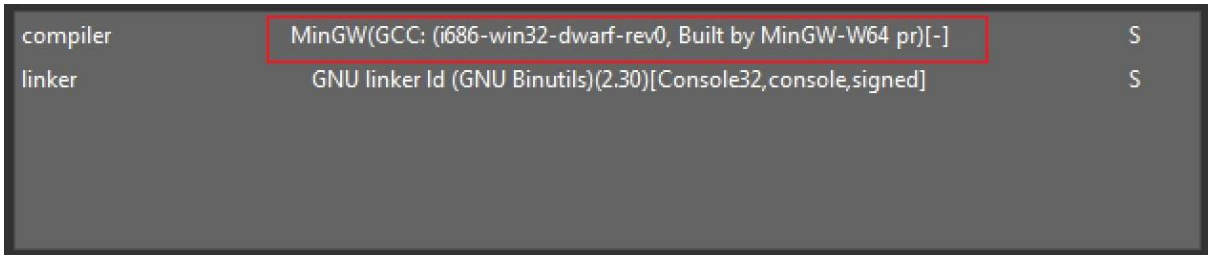


Figure 6: Compiler Info

Reverse Engineering

Infostealer “Prynt” process injects malicious code into the legitimate AppLaunch process to hide its activity. Threat actors deploy the infostealer Prynt.exe in the victim's environment with an aim to steal confidential and sensitive information.

Here we have analyzed the infostealer Prynt.exe on x64 Windows system.

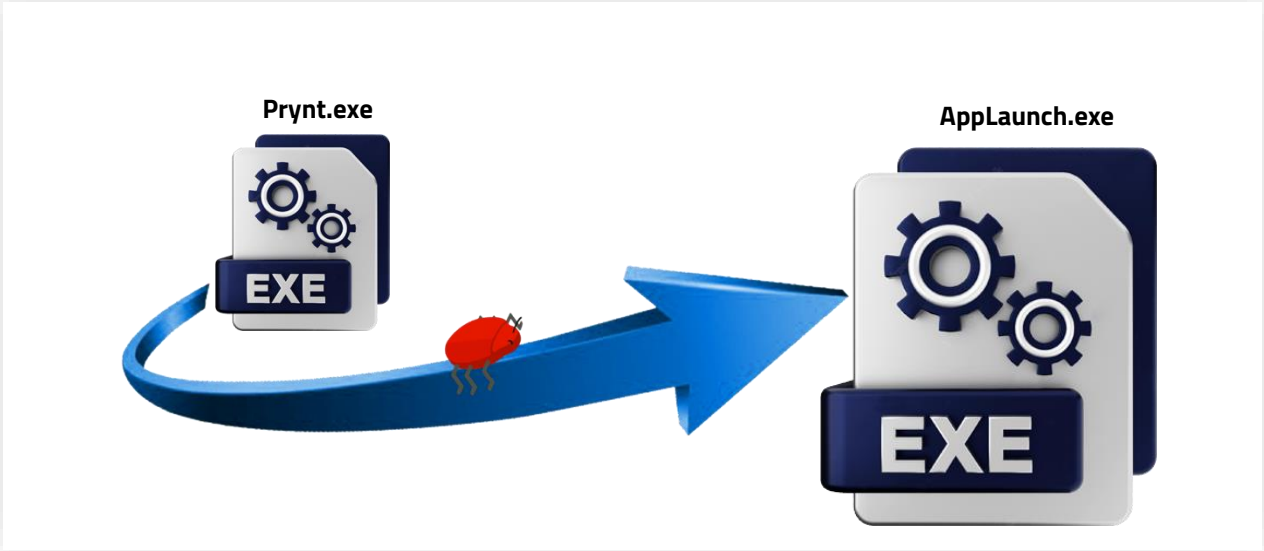


Figure 7: Process Injection

Process injection is a technique which injects malicious code into another running process and the targeted process executes the malicious code. This technique conceals malicious behaviour of their code and may bypass firewalls and other process-specific security mechanisms. Prynt.exe authors used a combination of VirtualAlloc/VirtualAllocEx, VirtualProtect, and WriteProcessMemory APIs to inject code into a remote process.

The infostealer “Prynt” first creates an AppLaunch process to host the malicious code in suspended mode:

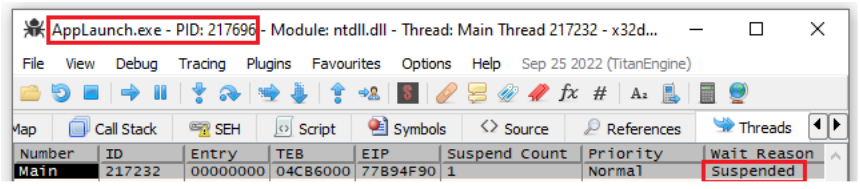


Figure 8: AppLaunch process in Suspended mode from x32dbg

AppLaunch process in suspended mode as seen from Task Manager: -

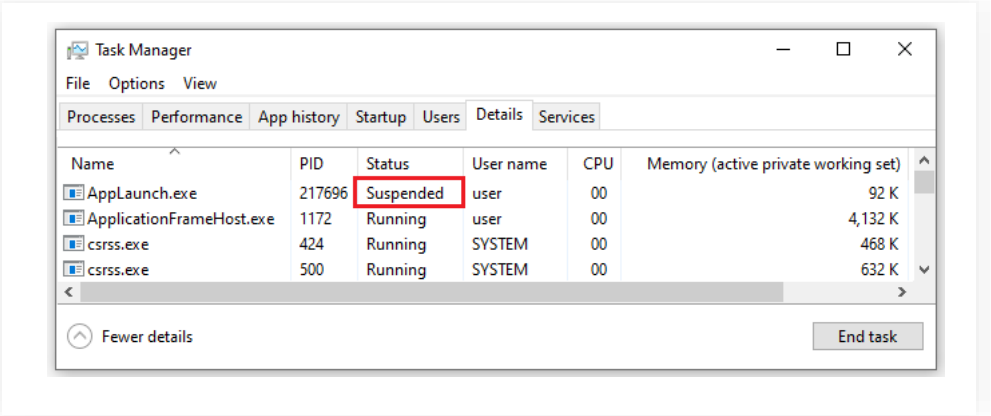


Figure 9: AppLaunch process in Suspended mode from Task Manager

Prynt process creates multiple threads via CreateRemoteThreadEx for establishing foundations for process injection and code execution. CreateRemoteThreadEx also creates a thread that runs in the virtual address space of another process.
VirtualProtect API used to change the permissions on a page in memory through PAGE_EXECUTE_READWRITE parameter: -

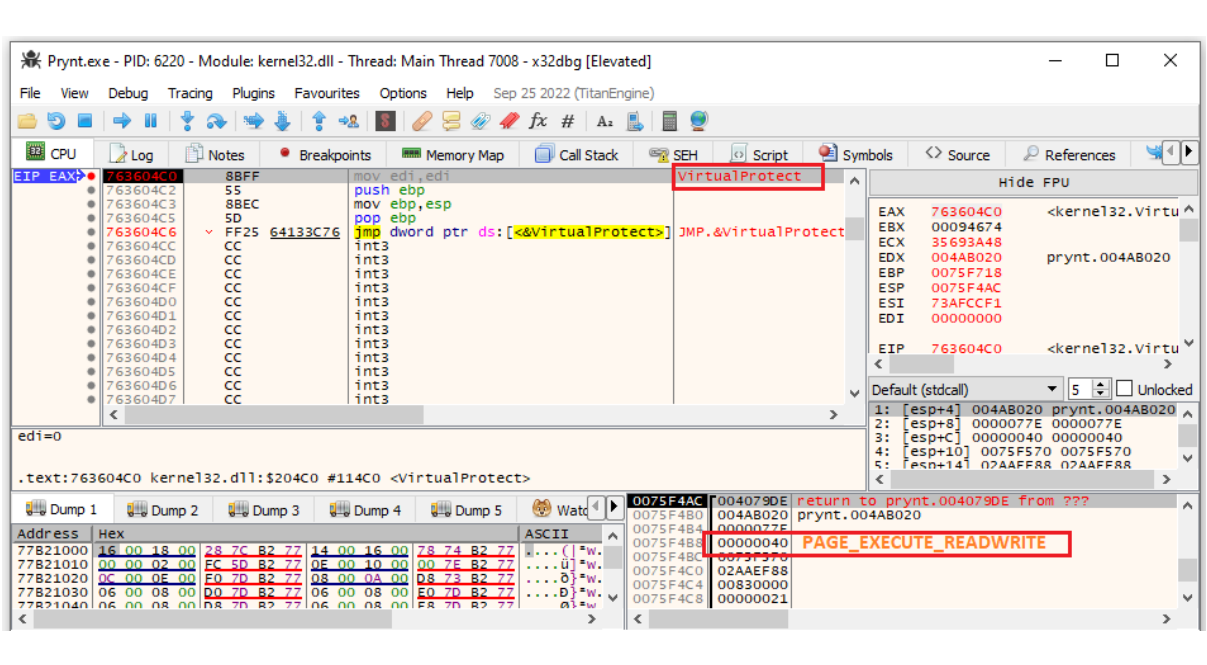


Figure 10: VirtualProtect API

VirtualProtect API changes are noticed as RWX permissions at **0x4AB000** in the Memory tab of the Prynt process: -

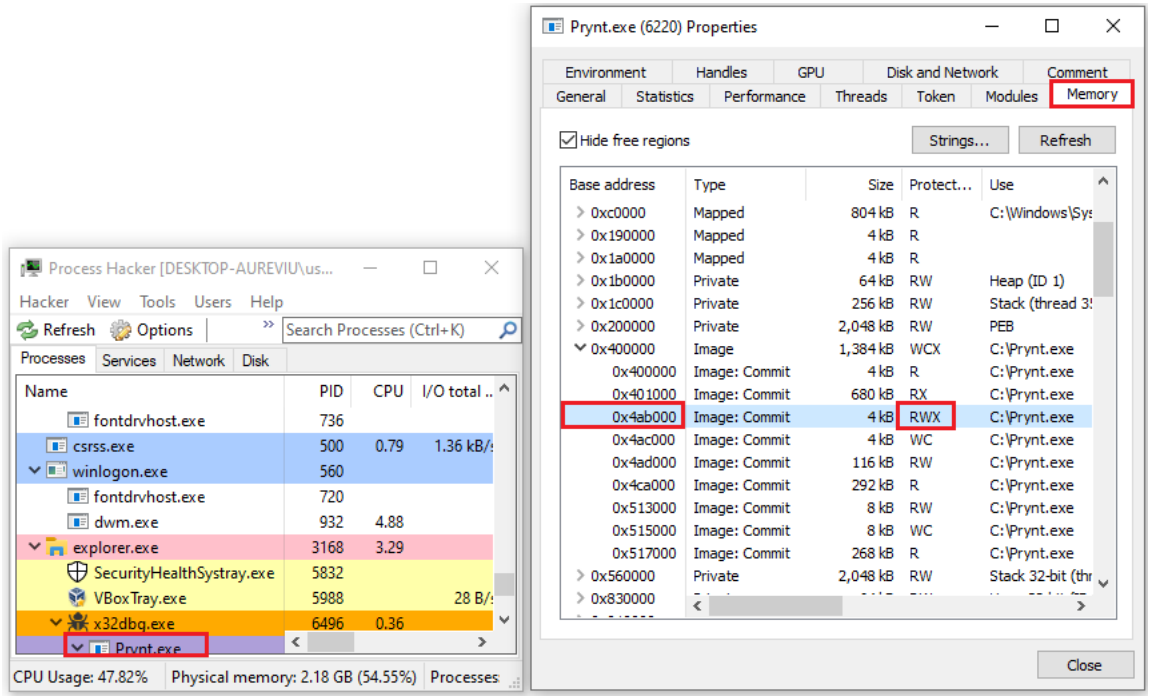


Figure 11: RWX permissions in Prynt

Prynt process allocates new memory using VirtualAlloc API: -

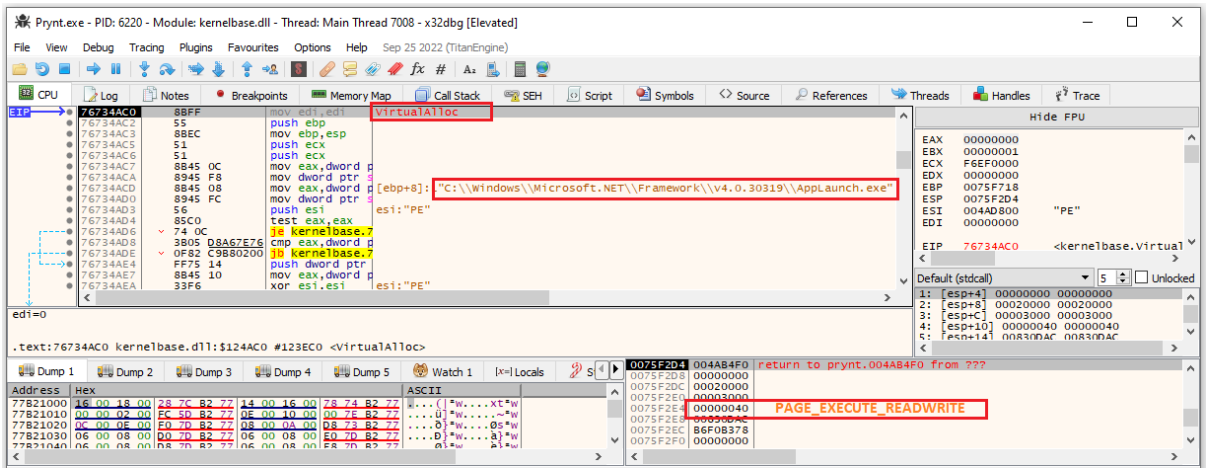


Figure 12: VirtualAlloc API

Prynt is seen allocating new memory in a remote AppLaunch process using VirtualAllocEx API: -

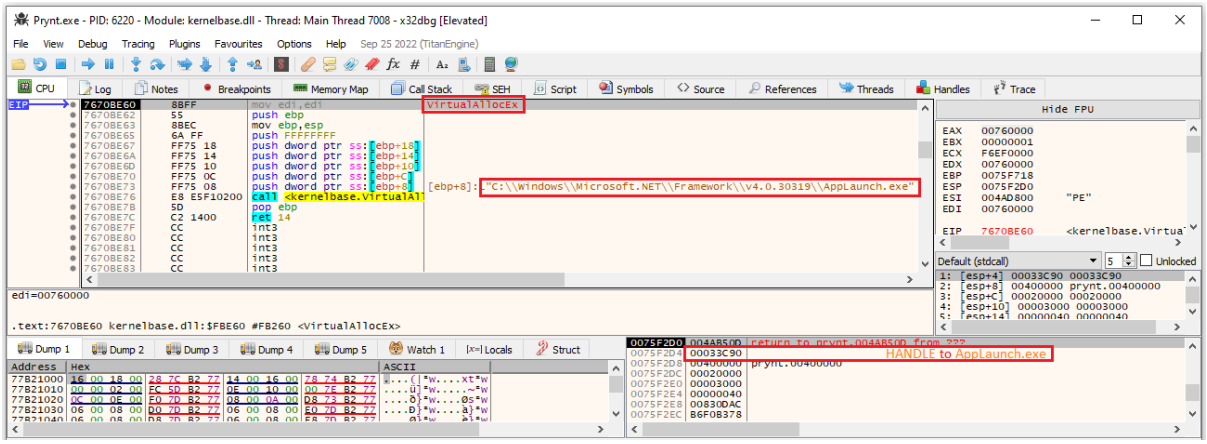


Figure 13: VirtualAllocEx API

Handles tab of the Prynt process showing the process handle value **0x33c90** associated with the AppLaunch: -

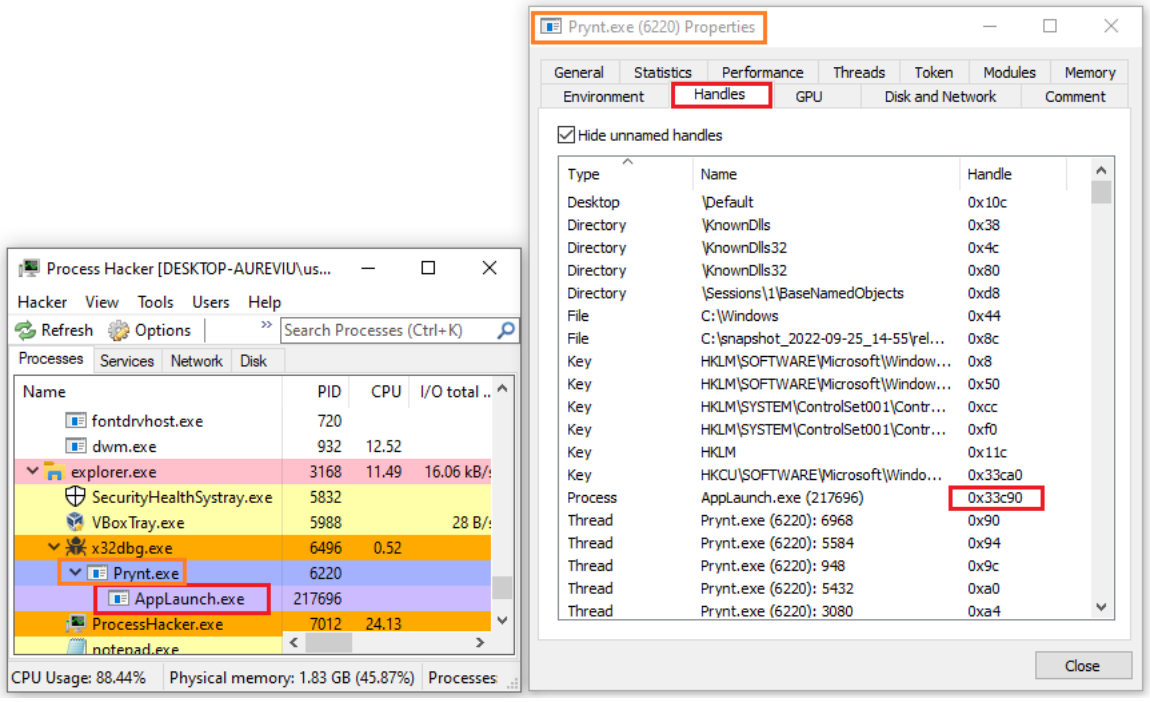


Figure 14: AppLaunch process handle

WriteProcessMemory API is used to write data into the allocated space :-

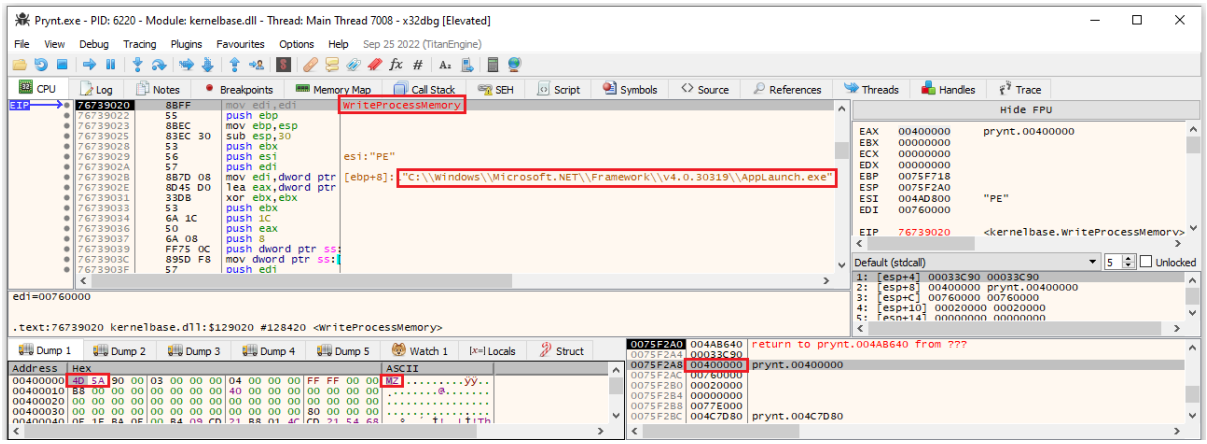


Figure 15: WriteProcessMemory API

AppLaunch process was injected at **0x400000** address with PAGE_EXECUTE_READWRITE (RWX) permission:-

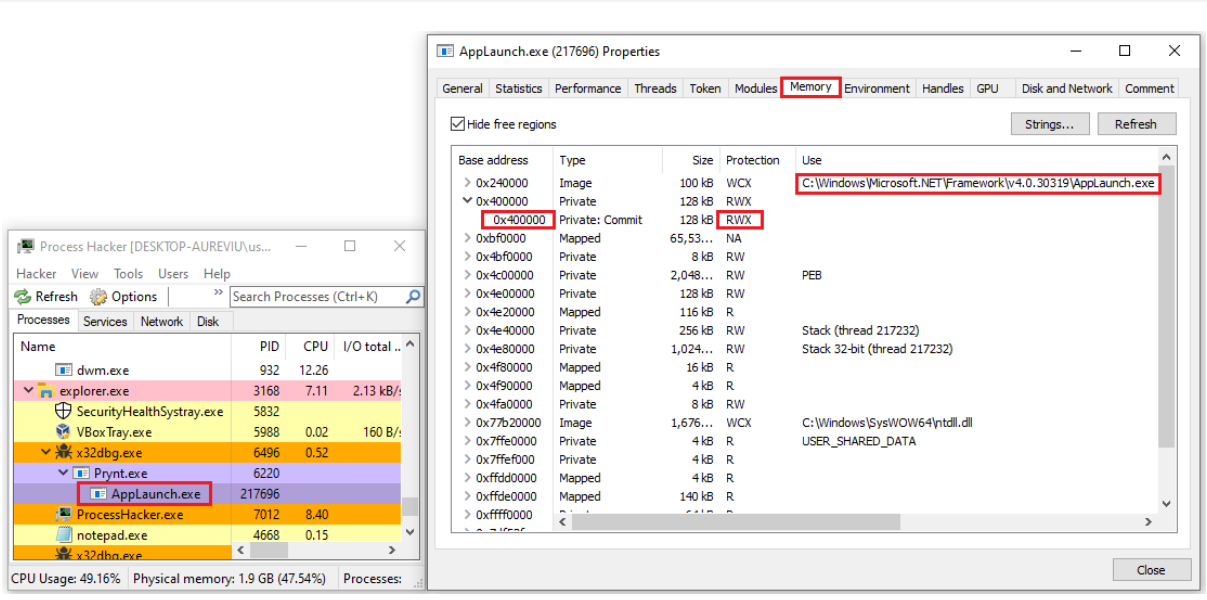


Figure 16: AppLaunch Process Injection

Prynt process calling SetThreadContext API to point the entry point to a new code section and run the injected code: -

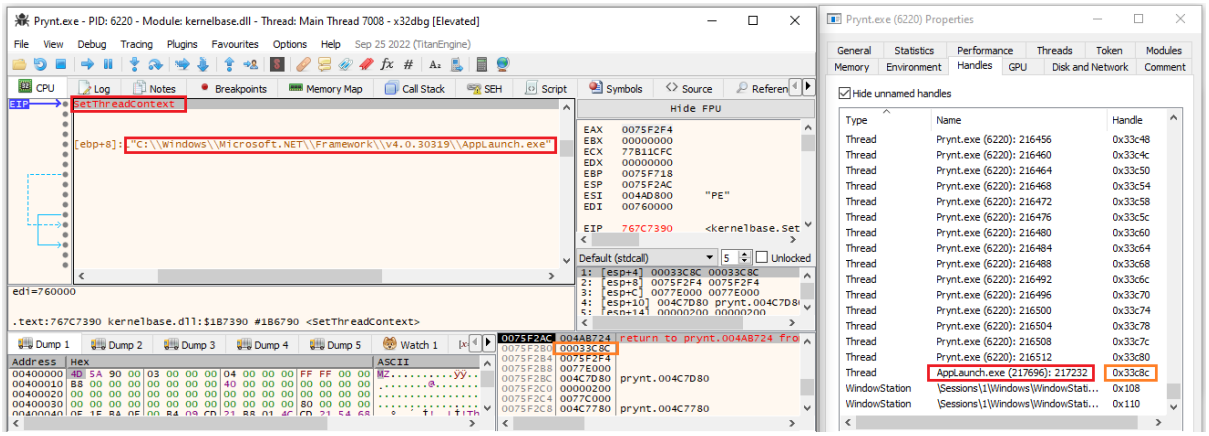


Figure 17: SetThreadContext API

Prynt resumes the suspended thread by calling NtResumeThread API to take the AppLaunch process out of the suspended state: -

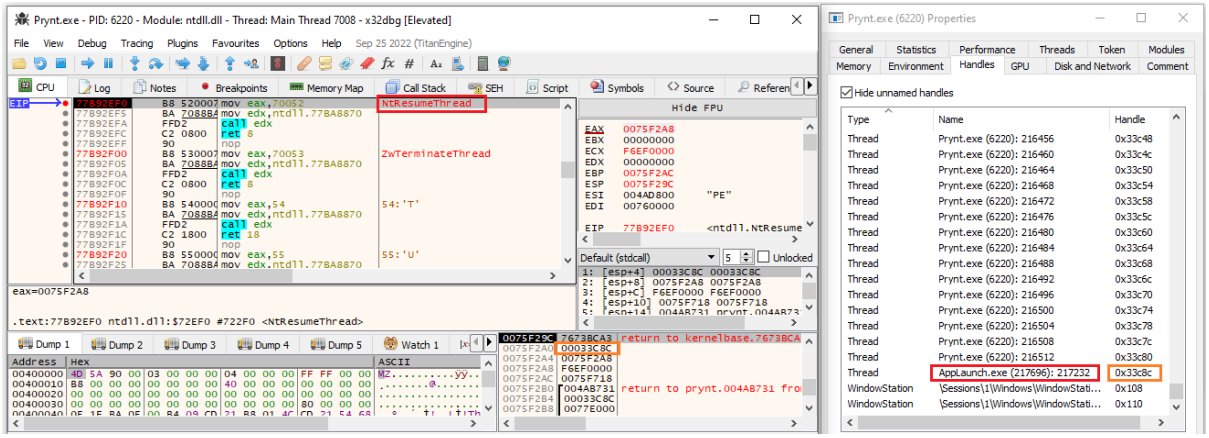


Figure 18: NtResumeThread API

Memory Forensics

Operating System: Windows 7 Service Pack 1 [SP1] (64-Bit)

A memory dump (RAM dump) win7sp1x64.raw was captured for memory analysis using WinPmem on x64 Windows 7 SP1 system.

Process Injection

Injected Process: AppLaunch.exe

Volatility plugin malfind was run against win7sp1x64.raw without specifying the -p option, it automatically identified the suspicious memory regions of all the processes running on the system based on the memory content and the VAD characteristics.

The dlllist plugin, which gets module information from the PEB, shows the full path to AppLaunch.exe (pid 215000).

```
C:\>volatility -f win7sp1x64.raw --profile=Win7SP1x64 dlllist -p 215000
Volatility Foundation Volatility Framework 2.6
************************************************************************
AppLaunch.exe pid: 215000
Command line : "C:\Windows\Microsoft.NET\Framework\v4.0.30319\AppLaunch.exe"
Note: use ldrmodules for listing DLLs in Wow64 processes

Base                             Size          LoadCount Path
-------------------- ---------------- ---------------- ----
0x000000000f30000           0x18000             0xffff C:\Windows\Microsoft.NET\Framework\v4.0.30319\AppLaunch.exe
0x0000000077570000          0x19f000             0xffff C:\Windows\SYSTEM32\ntdll.dll
0x0000000074130000           0x3f000                0x3 C:\Windows\SYSTEM32\wow64.dll
0x000000000740d0000           0x5c000                0x1 C:\Windows\SYSTEM32\wow64win.dll
0x000000000740c0000            0x8000                0x1 C:\Windows\SYSTEM32\wow64cpu.dll
```

Figure 19: dlllist plugin

The malfind plugin identified the suspicious memory protection at the address (**0x400000**) after running against the process AppLaunch.exe (pid 215000).

```
C:\>volatility -f win7sp1x64.raw --profile=Win7SP1x64 malfind -p 215000
Volatility Foundation Volatility Framework 2.6
Process: AppLaunch.exe Pid: 215000 Address: 0x400000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 32, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x00400000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x00400010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x00400020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00400030  00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00   ................
```

Figure 20: malfind plugin

The vadinfo plugin confirms the suspicious memory region protection at the address (**0x400000**).

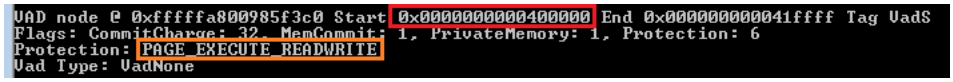


Figure 21: vadinfo plugin

Regular loaded libraries in the address space of a process are of type _MMVAD (Vad) or _MMVAD_LONG (VadL) which represent memory-mapped files.

VadS (_MMVAD_SHORT**)** tag looks suspicious as it represents dynamically allocated memory pages created via VirtualAllocEx /WriteProcessMemory. But VirtualAllocEx API cannot allocate memory with PAGE_EXECUTE_WRITECOPY protection since the executable is loaded into memory with PAGE_EXECUTE_WRITECOPY protection by the operating system.

The volshell plugin's db command dumps the content of the memory address **0x400000**.

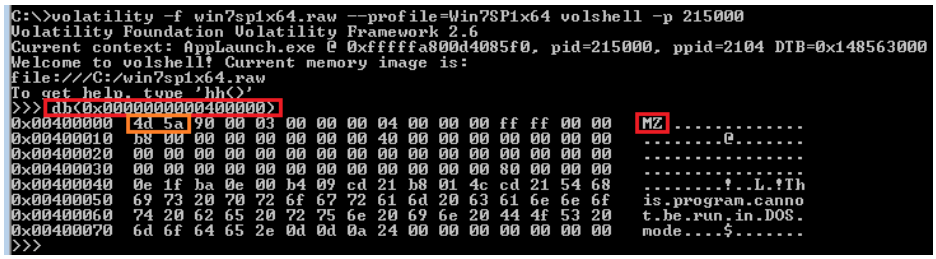


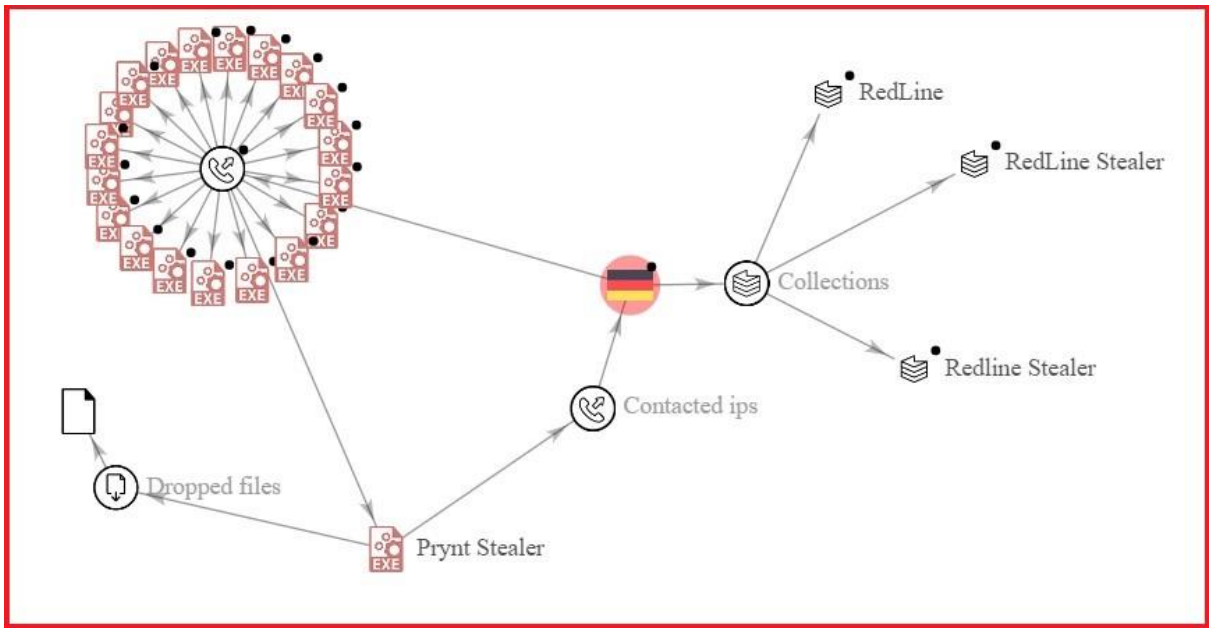
Figure 22: volshell plugin

The AppLaunch process (pid 215000) was injected by allocating memory with the PAGE_EXECUTE_READWRITE permission.

Trends Observed

Infostealer “Prynt” has been quietly used alongside RedLine stealer by the same set of Threat Actors to diversify their payloads. RedLine stealer has received a lot of criticism on the underground forums for being overpriced, so infostealer “Prynt” offers a cheaper alternative, which appears to be tested by Threat Actors. With time, if infostealer “Prynt” proves its utility during campaign deployment, we are likely to see a wider spread footprint, but at the moment it is yet to become a mainstream choice.

As shown in below graph, the infostealer “Prynt” stealer and RedLine stealer have common IP addresses in use – providing credibility to our analysis and hypothesis that slowly threat actors are shifting their focus towards infostealer “Prynt”.



Source: Virus Total

Conclusion

Information stealers are prominent malware in the current threat landscape and provide an opportunity for cyber-criminals to steal confidential and system information which can be leveraged to conduct next stage of cyber- attacks like ransomware. Information stealers include backdoor programs, which open specific ports of the computer so the attacker can take control of the system.

Infostealer “Prynt” is configured through a builder and has a secret backdoor as part of the code which is further available in other variants. Infostealer “Prynt” uses process injection as a defence evasion technique and entails running malicious code within the address space of the legitimate process.

Organizations are advised to ensure that unnecessary ports and services are closed to prevent the risk of discovery and potential exploitation.

List of IOCs

No.	Indicator	Type	Remarks
1	BCD1E2DC3740BF5EB616E8249D1E2D9C	MD5	Prynt.exe

MITRE ATT&CK™ Techniques Detection

No.	Tactic	Technique
1	Execution (**TA0002**)	**T1106:** Native API
2	Privilege Escalation (**TA0004**)	**T1055:** Process Injection
3	Defense Evasion (**TA0005**)	**T1059:** Hijack Execution Flow
		T1036: Masquerading
		T1112: Modify Registry
4	Discovery (**TA0007**)	**T1082:** System Information Discovery
		T1012: Query Registry
5	Command and Control (**TA0011**)	**T1102:** Web Service

CYFIRMA
DECODING THREATS

CYFIRMA is an external threat landscape management platform company. We combine cyber intelligence with attack surface discovery and digital risk protection to deliver the early warning, personalized, contextual, outside-in, and multi-layered insights. Our cloud-based AI and ML-powered analytics platform provide the hacker's view with deep insights into the external cyber landscape, helping clients prepare for impending attacks. CYFIRMA is headquartered in Singapore with offices across APAC, US, and EMEA. The company is funded by Goldman Sachs, Zodius Capital, and Z3 Partners.