WARMCOOKIE One Year Later: New Features and Fresh Insights

A year later: Elastic Security Labs re-examines the WARMCOOKIE backdoor.

10 min readMalware analysis

Revisiting WARMCOOKIE

Elastic Security Labs continues to track developments in the WARMCOOKIE codebase, uncovering new infrastructure tied to the backdoor. Since our original post, we have been observing ongoing updates to the code family and continued activity surrounding the backdoor, including new infections and its use with emerging loaders. A recent finding by the IBM X-Force team highlighted a new Malware-as-a-Service (MaaS) loader, dubbed CASTLEBOT, distributing WARMCOOKIE.

In this article, we will review new features added to WARMCOOKIE since its initial publication. Following this, we'll present the extracted configuration information from various samples.

Key takeaways

- The WARMCOOKIE backdoor is actively developed and distributed
- Campaign ID, a recently added marker, sheds light on targeting specific services and platforms
- WARMCOOKIE operators appear to receive variant builds distinguished by their command handlers and functionality
- Elastic Security Labs identified a default certificate that can be used to track new WARMCOOKIE C2 servers

WARMCOOKIE recap

We first published research about WARMCOOKIE in the summer of 2024, detailing its functionality and how it was deployed through recruiting-themed phishing campaigns. Since then, we have observed various development changes to the malware, including the addition of new handlers, a new campaign ID field, code optimization, and evasion adjustments.

WARMCOOKIE's significance was highlighted in May 2025, during Europol's Operation Endgame, in which multiple high-profile malware families, including WARMCOOKIE, were disrupted. Despite this, we are still seeing the backdoor being actively used in various malvertising and spam campaigns.

WARMCOOKIE updates

Handlers

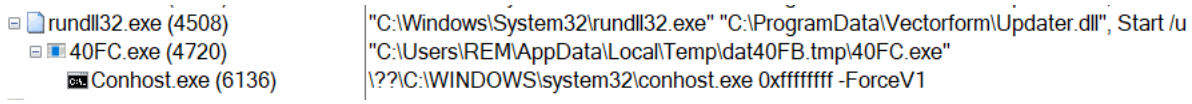
During our analysis of the new variant of WARMCOOKIE, we identified four new handlers introduced in the summer of 2024, providing quick capabilities to launch executables, DLLs, and scripts:

- PE file execution
- DLL execution
- PowerShell script execution
- DLL execution with `Start` export

```
switch ( exec_type )
{
  case 1:
    str_exe = des::StringDecrypt(dword_140025E98);// .exe
    len_temp_file = wcslen(TempFileName);
    wcscpy(&temp_file[len_temp_file], str_exe);
    des::ZeroOutFree(str_exe);
    break;
  case 2:
    str_dll = des::StringDecrypt(dword_140025EB0);// .dll
    len_path_dll = wcslen(TempFileName);
    wcscpy(&temp_file[len_path_dll], str_dll);
    des::ZeroOutFree(str_dll);
    break;
  case 3:
    str_ps1 = des::StringDecrypt(dword_140025EC8);// .ps1
    len_path_ps1 = wcslen(TempFileName);
    wcscpy(&temp_file[len_path_ps1], str_ps1);
    des::ZeroOutFree(str_ps1);
    break;
}
```

Switch statement inside command handler

The most recent WARMCOOKIE builds we have collected contain the DLL/EXE execution functionality, with PowerShell script functionality being much less prevalent. These capabilities leverage the same function by passing different arguments for each file type. The handler creates a folder in a temporary directory, writing the file content (EXE / DLL / PS1) to a temporary file in the newly created folder. Then, it executes the temporary file directly or uses either `rundll32.exe` or `PowerShell.exe`. Below is an example of PE execution from procmon.

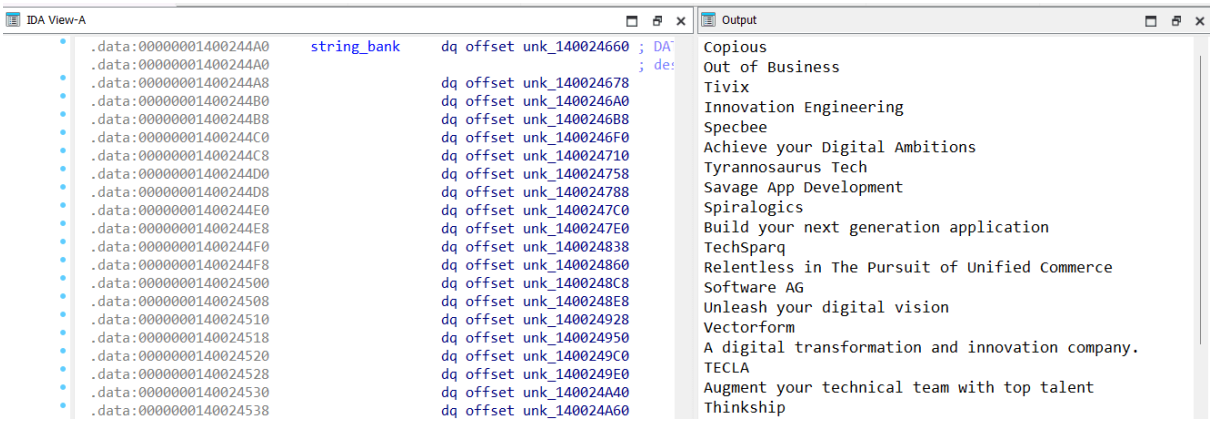


PE execution handler via Procmon

String bank

Another change observed was the adoption of using a list of legitimate companies for the folder paths and scheduled task names for WARMCOOKIE (referred to as a "string bank"). This is done for defense evasion purposes, allowing

the malware to relocate to more legitimate-looking directories. This approach uses a more dynamic method (a list of companies to use as folder paths, assigned at malware runtime) as opposed to hardcoding the path into a static location, as we observed with previous variants (`C:\ProgramData\RtlUpd\RtlUpd.dll`).



WARMCOOKIE string bank

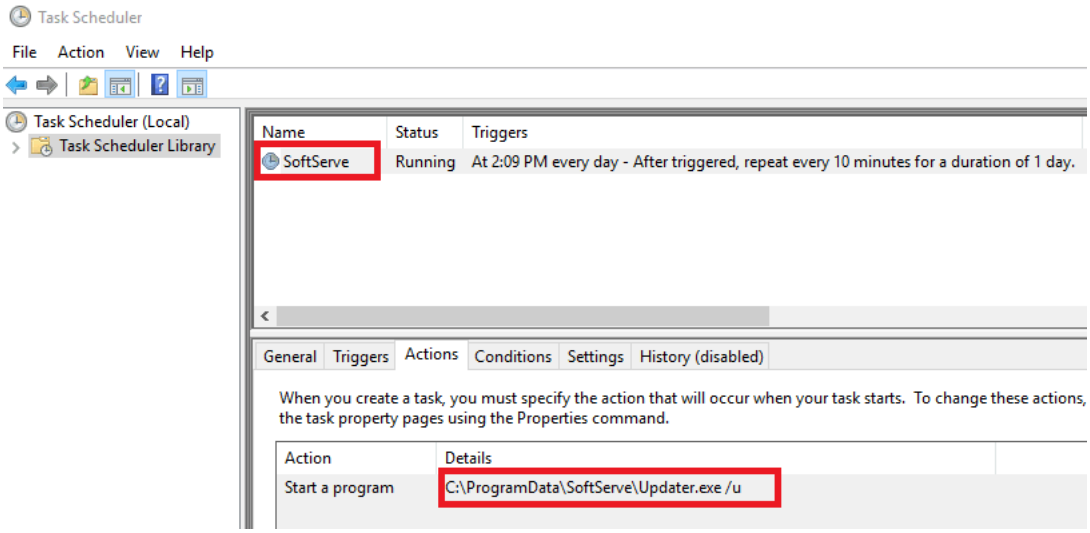
The malware uses `GetTickCount` as a seed for the `srand` function to randomly select a string from the string bank.

```
__int64 __fastcall des::GetRandomIntegerFromRange(int min_value, int max_value)
{
  unsigned int TickCount; // eax

  TickCount = GetTickCount();
  srand(TickCount);
  return rand() % (max_value - min_value + 1) + min_value;
}
```

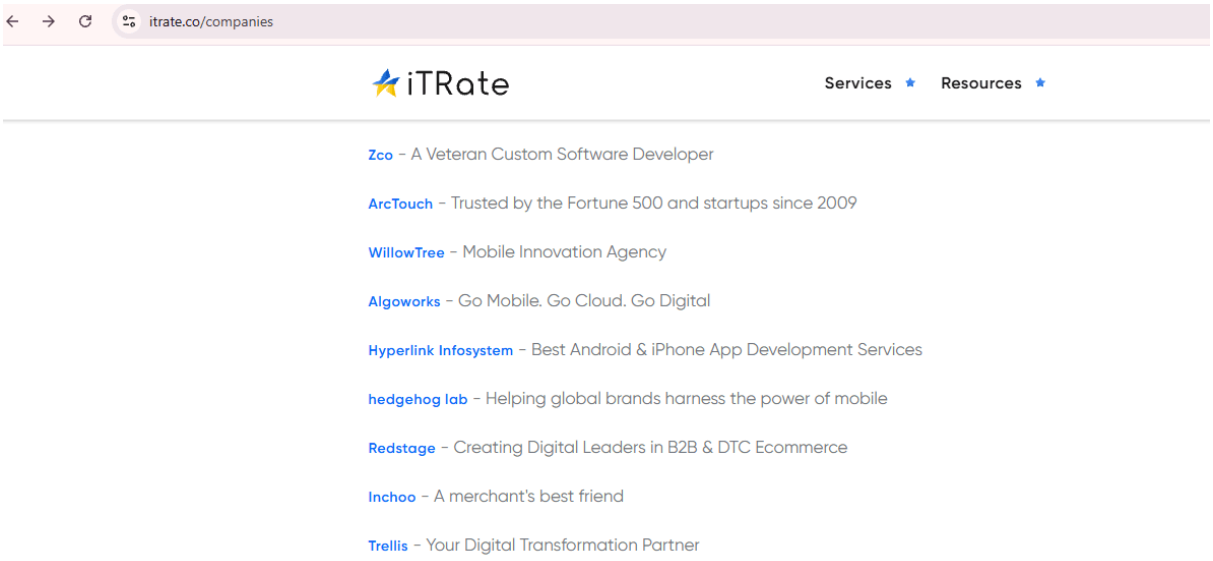
Function used for selecting strings from the string bank

The following depicts an example of a scheduled task showing the task name and folder location:



Scheduled task using string bank

By searching a few of these names and descriptions, our team found that this string bank is sourced from a website used to rate and find reputable IT/Software companies.



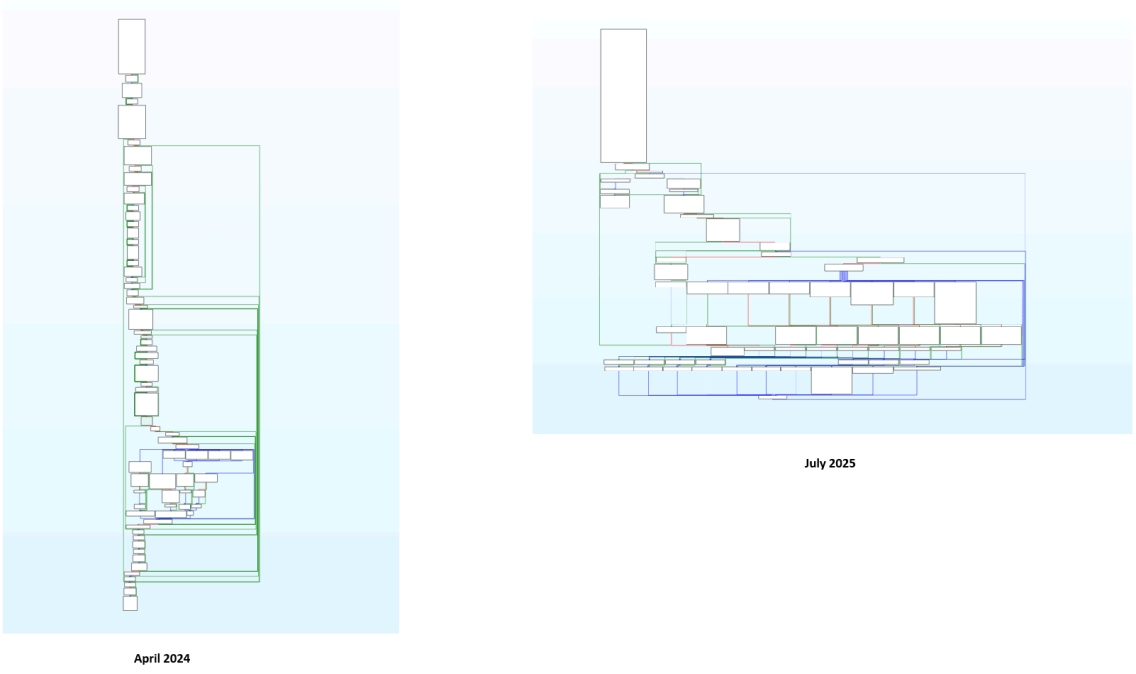
IT rating website used to populate the string bank

Smaller changes

In our last write-up, WARMCOOKIE passed a command-line parameter using `/p` to determine if a scheduled task needs to be created; this parameter has been changed to `/u`. This appears to be a small, but additional change to break away from previous reporting.

In this new variant, WARMCOOKIE now embeds 2 separate GUID-like mutexes; these are used in combination to better control initialization and synchronization. Previous versions only used one mutex.

Another noticeable improvement in the more recent versions of WARMCOOKE is code optimization. The implementation seen below is now cleaner with less inline logic which makes the program optimized for readability, performance, and maintainability.



Code optimization comparison

Clustering configs

Since our initial publication in July 2024, WARMCOOKIE samples have included a campaign ID field. This field is used by operators as a tag or marker providing context to the operators around the infection, such as the distribution method. Below is an example of a sample with a campaign ID of `traffic2`.

```
checksum[0] = VolumeSerialNumber ^ checksum_mutex;
computer_name_checksum = des::CalculateCRC32Checksum(computer_name, 2 * size_computer_name, -1);
checksum[1] = computer_name_checksum ^ des::CalculateCRC32Checksum(username, 2 * size_username_4, -1);
des::RetrieveOSInfo(p_os_info);
key = des::StringDecrypt2(dword_14001B580);   // 416590bdc875e4474a4d
campaign_string = des::StringDecrypt(dword_14001B620);// traffic2
```

Campaign ID within WARMCOOKIE

Based on the extracted configurations of samples in the last year, we hypothesize that the embedded RC4 key can be used to distinguish between operators using WARMCOOKIE. While unproven, we observed from various samples that some patterns started to emerge based on clustering the RC4 key.

RC4 Key	Campaign ID	Campaign ID (Decoded)	Sample Count
81ea45461471	Carrier		1
	jsrh		1
	PrivateDLL		1
	ptk2		1
	qwa2		7
	smb64		2
	traf		1
83ddc084e21a244c	aws		2
	bing2		2
	bing3		1
	YmluZw==	bing	1
fd1285af2130	capo		1
	WTJGd2J3PT0=	capo	3
	Y2Fwbw==	capo	3
4b42ab8d536f	adobe.com		1
	B00FCA1005FFF		3
	E2A09F		1
416590bdc875e4474a4d	traffic1		1
	traffic2		1
ac180d12b62a	bG9kMmxvZA==	lod2lod	1

RC4 key distribution with campaign IDs

By using the RC4 key, we can see overlap in campaign themes over time, such as the build using RC4 key `83ddc084e21a244c`, which leverages keywords such as `bing`, `bing2`, `bing3`,and `aws` for campaign mapping. An interesting note, as it relates to these build artifacts, is that some builds contain different command handlers/functionality. For example, the build using the RC4 key `83ddc084e21a244c` is the only variant we have observed that has PowerShell script execution capabilities, while most recent builds contain the DLL/EXE handlers.

Other campaign IDs appear to use terms such as `lod2lod`, `capo,` or `PrivateDLL`. For the first time, we saw the use of embedded domains versus numeric IP addresses in WARMCOOKIE from a sample in July 2025.

WARMCOOKIE infrastructure overview

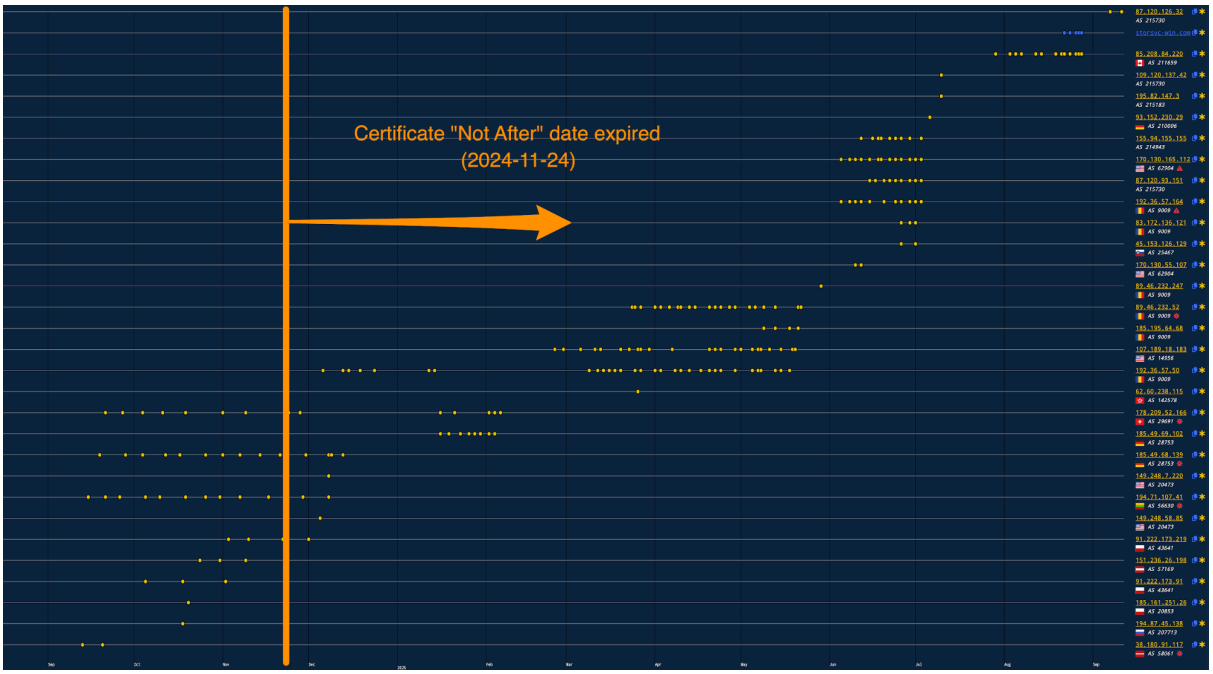
After extracting the infrastructure from these configurations, one SSL certificate stands out. Our hypothesis is that the certificate below is possibly a default certificate used for the WARMCOOKIE back-end.

```
Issuer
    C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
Not Before
    2023-11-25T02:46:19Z
Not After
    2024-11-24T02:46:19Z
Fingerprint (SHA1)
    e88727d4f95f0a366c2b3b4a742950a14eff04a4
```

```
Fingerprint (SHA256)
    8c5522c6f2ca22af8db14d404dbf5647a1eba13f2b0f73b0a06d8e304bd89cc0
```

Certificate details

Note the “Not After” date above shows that this certificate is expired. However, new (and reused) infrastructure continues to be initialized using this expired certificate. This is not entirely new infrastructure, but rather a reconfiguration of redirectors to breathe new life into existing infrastructure. This could indicate that the campaign owners are not concerned with the C2 being discovered.



Certificate reuse screenshot, September 2024 to September 2025

Conclusion

Elastic Security Labs continues to observe WARMCOOKIE infections and the deployment of new infrastructure for this family. Over the last year, the developer has continued to make updates and changes, suggesting it will be around for some time to come. Based on its selective usage, it continues to remain under the radar. We hope that by sharing this information, organizations will be better equipped to protect themselves from this threat.

Malware and MITRE ATT&CK

Elastic uses the MITRE ATT&CK framework to document common tactics, techniques, and procedures that advanced persistent threats use against enterprise networks.

Tactics

Tactics represent the why of a technique or sub-technique. It is the adversary’s tactical goal: the reason for performing an action.

- Initial Access
- Execution
- Defense Evasion
- Discovery
- Command and Control
- Exfiltration

Techniques

Techniques represent how an adversary achieves a tactical goal by performing an action.

Detecting malware

Prevention

YARA

Elastic Security has created the following YARA rules to identify this activity.

- Windows.Trojan.WarmCookie

Observations

The following observables were discussed in this research.

Observable	Type	Name	Reference
87.120.126.32	ipv4-addr		WARMCOOKIE C2 Server
storsvc-win[.]com	domain		WARMCOOKIE C2 Server
85.208.84.220	ipv4-addr		WARMCOOKIE C2 Server
109.120.137.42	ipv4-addr		WARMCOOKIE C2 Server
195.82.147.3	ipv4-addr		WARMCOOKIE C2 Server
93.152.230.29	ipv4-addr		WARMCOOKIE C2 Server
155.94.155.155	ipv4-addr		WARMCOOKIE C2 Server
87.120.93.151	ipv4-addr		WARMCOOKIE C2 Server
170.130.165.112	ipv4-addr		WARMCOOKIE C2 Server
192.36.57.164	ipv4-addr		WARMCOOKIE C2 Server
83.172.136.121	ipv4-addr		WARMCOOKIE C2 Server
45.153.126.129	ipv4-addr		WARMCOOKIE C2 Server
170.130.55.107	ipv4-addr		WARMCOOKIE C2 Server
89.46.232.247	ipv4-addr		WARMCOOKIE C2 Server
89.46.232.52	ipv4-addr		WARMCOOKIE C2 Server
185.195.64.68	ipv4-addr		WARMCOOKIE C2 Server
107.189.18.183	ipv4-addr		WARMCOOKIE C2 Server
192.36.57.50	ipv4-addr		WARMCOOKIE C2 Server
62.60.238.115	ipv4-addr		WARMCOOKIE C2 Server
178.209.52.166	ipv4-addr		WARMCOOKIE C2 Server
185.49.69.102	ipv4-addr		WARMCOOKIE C2 Server
185.49.68.139	ipv4-addr		WARMCOOKIE C2 Server
149.248.7.220	ipv4-addr		WARMCOOKIE C2 Server
194.71.107.41	ipv4-addr		WARMCOOKIE C2 Server
149.248.58.85	ipv4-addr		WARMCOOKIE C2 Server
91.222.173.219	ipv4-addr		WARMCOOKIE C2 Server
151.236.26.198	ipv4-addr		WARMCOOKIE C2 Server
91.222.173.91	ipv4-addr		WARMCOOKIE C2 Server
185.161.251.26	ipv4-addr		WARMCOOKIE C2 Server

Observable	Type	Name	Reference
194.87.45.138	ipv4-addr		WARMCOOKIE C2 Server
38.180.91.117	ipv4-addr		WARMCOOKIE C2 Server
c7bb97341d2f0b2a8cd327e688acb65eaefc1e01c61faaeba2bc1e4e5f0e6f6e	SHA-256		WARMCOOKIE
9d143e0be6e08534bb84f6c478b95be26867bef2985b1fe55f45a378fc3ccf2b	SHA-256		WARMCOOKIE
f4d2c9470b322af29b9188a3a590cbe85bacb9cc8fcd7c2e94d82271ded3f659	SHA-256		WARMCOOKIE
5bca7f1942e07e8c12ecd9c802ecdb96570dfaaa1f44a6753ebb9ffda0604cb4	SHA-256		WARMCOOKIE
b7aec5f73d2a6bbd8cd920edb4760e2edadc98c3a45bf4fa994d47ca9cbd02f6	SHA-256		WARMCOOKIE
e0de5a2549749aca818b94472e827e697dac5796f45edd85bc0ff6ef298c5555	SHA-256		WARMCOOKIE
169c30e06f12e33c12dc92b909b7b69ce77bcbfc2aca91c5c096dc0f1938fe76	SHA-256		WARMCOOKIE

References

The following were referenced throughout the above research: