Cookie stealing: the new perimeter bypass

news.sophos.com/en-us/2022/08/18/cookie-stealing-the-new-perimeter-bypass

Sean Gallagher

August 18, 2022

Credential-stealing malware is an integral part of the toolkit used by a wide variety of cybercriminals and other adversaries. While user account names and passwords are the most obvious targets of credential-stealing activities, the increased use of multi-factor authentication (MFA) to protect web-based services has reduced the effectiveness of that approach. Attackers are increasingly turning to stealing the “cookies” associated with credentials to clone active or recent web sessions—bypassing MFA in the process.

The latest version of the Emotet botnet is just one of the many malware families that target cookies and other credentials stored by browsers, such as stored logins and (in some cases) payment card data. Google’s Chrome browser uses the same encryption method to store both multi-factor authentication cookies and credit card data—both targets of Emotet.

The range of criminals targeting cookies is broad. At the bottom end of the cybercrime range, information-stealing malware such as the Raccoon Stealer malware-as-a-service and the RedLine Stealer keylogger / information stealer—both of which can be purchased through underground forums—are often used by entry-level criminals to collect cookies and other credentials in bulk for sale to criminal marketplaces.

One such marketplace, Genesis , was the apparent source for a cookie belonging to an employee of the game developer Electronic Arts. Members of the Lapsus$ extortion group claimed to have purchased a stolen session cookie from the marketplace, giving them access to EA's Slack instance; that allowed them to spoof an existing login of an EA employee and deceive a member of EA's IT team into providing them network access. This allowed Lapsus$ to grab 780 gigabytes of data, including game and graphics engine source code, which the group then used to attempt to extort EA.

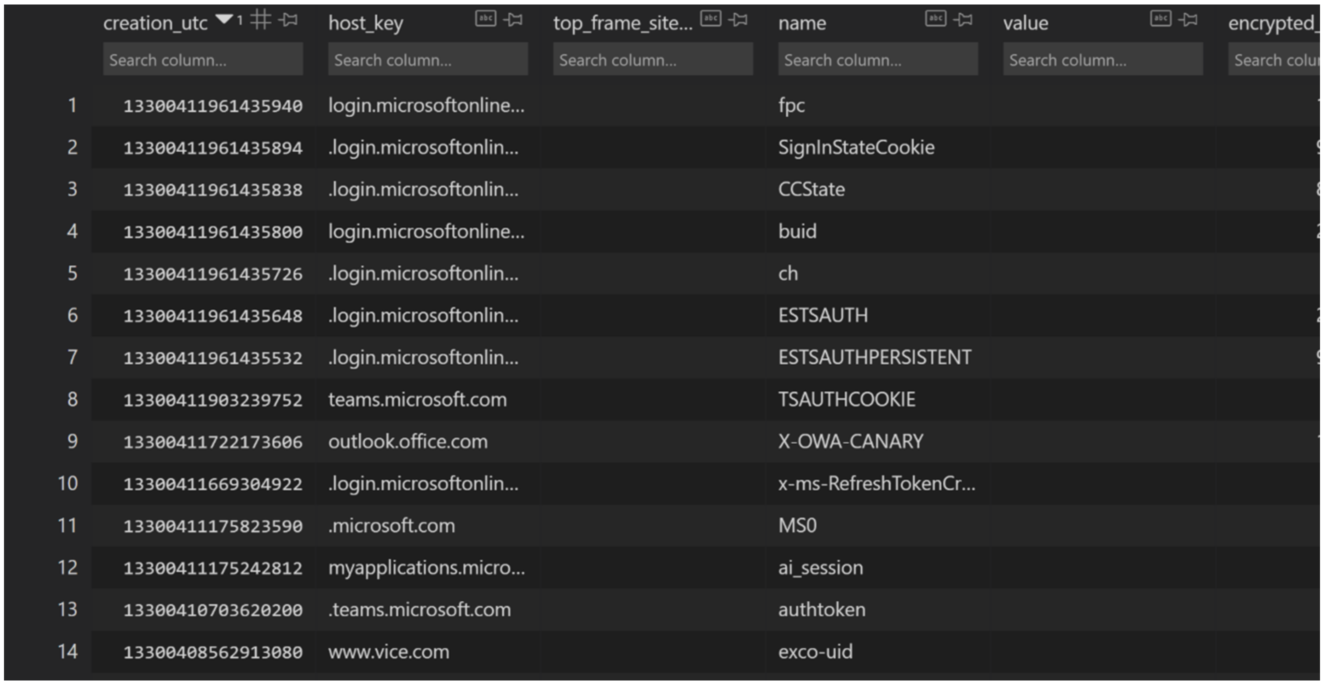
At the higher end of the criminal sophistication spectrum, we have observed active adversaries harvest cookies in a variety of ways. In some cases, we've seen evidence of ransomware operators using the same info stealer malware as less sophisticated attackers. But we've also often seen hands-on attacks abusing legitimate offensive-security tools such as Mimikatz, Metasploit Meterpreter and Cobalt Strike to execute cookie harvesting malware or run scripts that grab cookies from browsers' caches.

There are also legitimate applications and processes that interact with browser cookie files. We found anti-malware software, auditing tools, and operating system helpers among cookie-snooping detections in Sophos telemetry: Bing's wallpaper updater, for example, accesses cookies to retrieve new desktop backgrounds. But with these benign sources screened out, we saw thousands of attempts to access browser cookies per day that fall outside the realm of benign software behavior. Occasionally, these detections spike dramatically as specific campaigns are launched. Additionally, some legitimate applications that use cookies may leak them, exposing tokens to attackers.

Hands in the Cookie Jar

Browsers store cookies in a file; for Mozilla Firefox, Google Chrome, and Microsoft Edge, the file is a SQLite database in the user profile folder. (Similar SQLite files store browser history, website logins and autofill information on these browsers). Other applications that connect to remote services have their own cookie repositories, or in some cases access to those of web browsers.

The content of each cookie in the database is a list of parameters and values—a key-value store that identifies the browser session to the remote website, including in some cases a token passed by the site to the browser after user authentication. One of these key-value pairs specifies the expiration of the cookie—how long it is valid for before it must be renewed.



	creation_utc	host_key	top_frame_site...	name	value	encrypted_
1	13300411961435940	login.microsoftonline...		fpc		
2	13300411961435894	.login.microsoftonlin...		SignInStateCookie		
3	13300411961435838	.login.microsoftonlin...		CCState		
4	13300411961435800	login.microsoftonline...		buid		
5	13300411961435726	.login.microsoftonlin...		ch		
6	13300411961435648	.login.microsoftonlin...		ESTSAUTH		
7	13300411961435532	.login.microsoftonlin...		ESTSAUTHPERSISTENT		
8	13300411903239752	teams.microsoft.com		TSAUTHCOOKIE		
9	13300411722173606	outlook.office.com		X-OWA-CANARY		
10	13300411669304922	.login.microsoftonlin...		x-ms-RefreshTokenCr...		
11	13300411175823590	.microsoft.com		MS0		
12	13300411175242812	myapplications.micro...		ai_session		
13	13300410703620200	.teams.microsoft.com		authtoken		
14	13300408562913080	www.vice.com		exco-uid		

Figure 1: Some of the cookies in a cookies.sqlite file

The reason for cookie theft is straightforward: Cookies associated with authentication to web services can be used by attackers in “pass the cookie” attacks, attempting to masquerade as the legitimate user to whom the cookie was originally issued and gain access to web services without a login challenge. This is similar to “pass the hash” attacks, which use locally stored authentication hashes to gain access to network resources without having to crack the passwords.

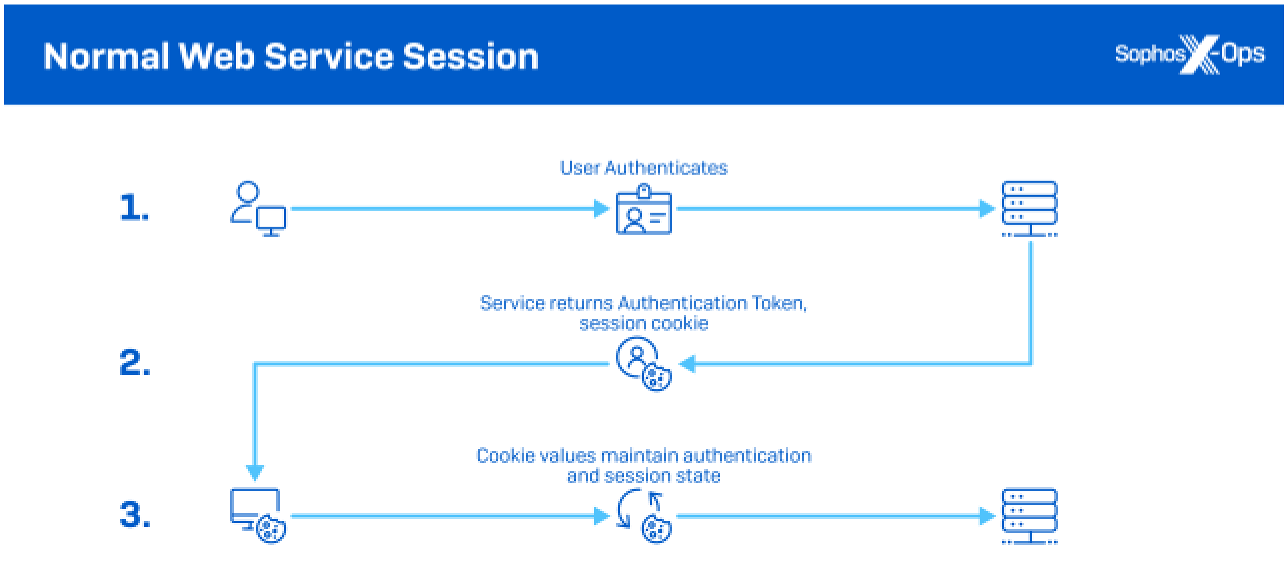
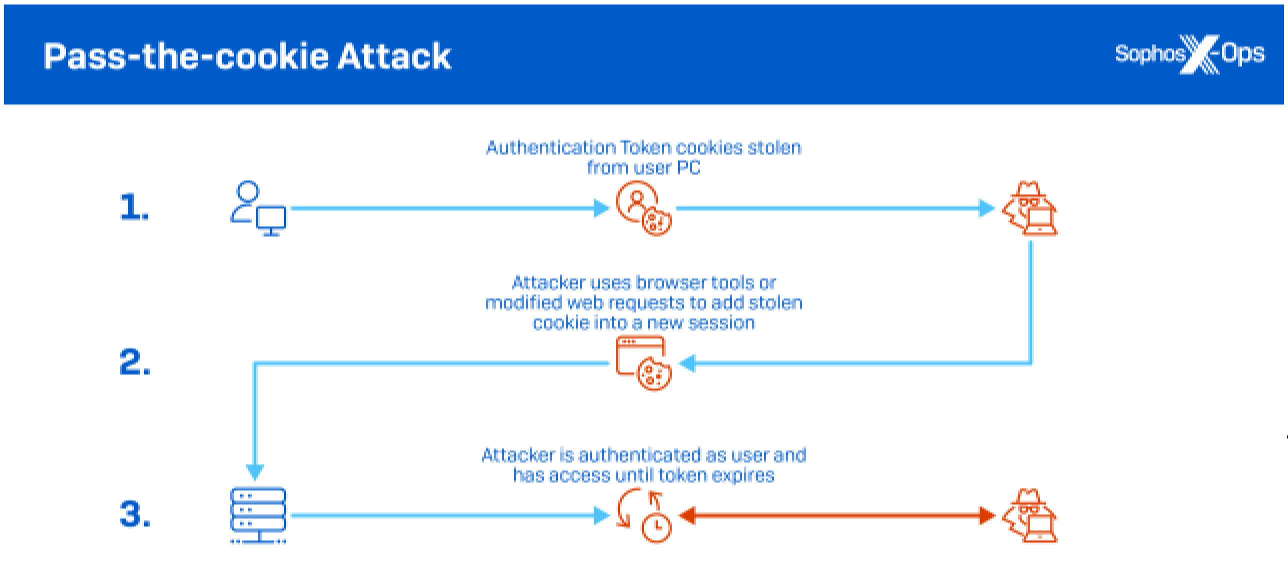

Normal Web Service Session
Sophos X-Ops
1.
User Authenticates
2.
Service returns Authentication Token, session cookie
3.
Cookie values maintain authentication and session state

Figure 2: Legitimate web-service activity…


Pass-the-cookie Attack
Sophos X-Ops
1.
Authentication Token cookies stolen from user PC
2.
Attacker uses browser tools or modified web requests to add stolen cookie into a new session
3.
Attacker is authenticated as user and has access until token expires

…

and how pass-the-cookie attacks subvert that.

This can lead to exploitation of web services (such as AWS or Azure), software-as-a-service and collaboration services for further data exposure or lateral movement, such as business email compromise, access to cloud data stores, or using a hijacked Slack session to lure additional victims into downloading malware or exposing other data that can be leveraged for access.

Many web-based applications perform additional checks to prevent session spoofing, such as checking the IP address of the request against where the session was initiated. But if the cookies are used by a hands-on-keyboard attacker from within the same network, those sorts of measures may not be enough to stop exploitation. And applications that are built for a combination of desktop and mobile use may not use geolocation as consistently.

Some cookie theft attacks may be sprung completely remotely from within the target’s own browser. HTML injection attacks can use code inserted into a vulnerable web page to exploit cookies for other services—giving access to the target’s profile information on those services and allowing password and email changes.

Cheaper by the Dozen

Often, malware operators will use paid download services and other untargeted approaches to gather as many victims’ cookies and other credentials as possible at low cost and with little effort. This type of stealer deployment is very similar to the ones used in Raccoon Stealer and other malware campaigns we saw being distributed through droppers as a service.

Malware packages in ISOs or ZIP files are advertised through malicious websites boosted by search engine optimization as installers for pirated or “cracked” commercial software packages. ISO-based delivery packages are also widely used in place of malicious documents in malware spam email campaigns, mostly because of Microsoft’s recent blocking of macros in Office files from the Internet.

In one download-as-a-service case we saw on a college network, stealer malware arrived wrapped up in a fake software installer downloaded from a website, most likely one advertising pirated commercial software. The installer was delivered in a 300-megabyte ISO file downloaded by the user; large ISO files are often used in an attempt to jam up file scans by malware detection software.

The ISO contained BLENDERINSTALLER3.0.EXE, a repurposed software installation utility from another software package. This dropper installs several files, using a PowerShell command and an executable created with AutoIT (a legitimate tool frequently abused by malware operators) to extract malware from the .ISO and download additional malware files from Discord’s content delivery network. The malware package then injects a series of commands through a .NET process (using jsc.exe from the .NET framework) to grab both cookies and login data from Chrome.

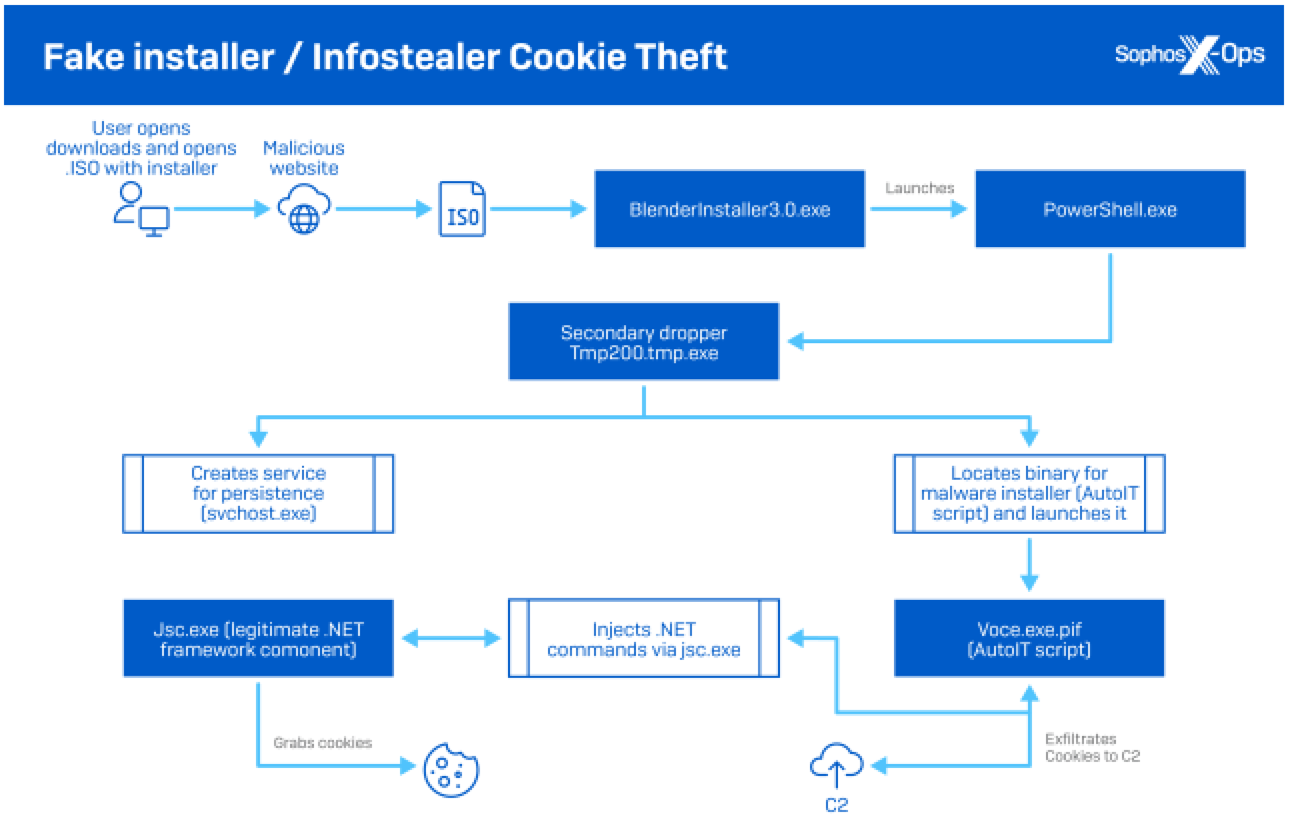

Fake installer / Infostealer Cookie Theft
Sophos X-Ops
User opens downloads and opens .ISO with installer
Malicious website
ISO
BlenderInstaller3.0.exe
Launches
PowerShell.exe
Secondary dropper Tmp200.tmp.exe
Creates service for persistence (svchost.exe)
Locates binary for malware installer (AutoIT script) and launches it
Jsc.exe (legitimate .NET framework comonent)
Injects .NET commands via jsc.exe
Voce.exe.pif (AutoIT script)
Grabs cookies
Exfiltrates Cookies to C2
C2

Figure 3: A fake installer / info stealer cookie theft

Crafted attacks

Malicious spam is also used with other disguised attachments, often targeting organizations in specific industries or locations. In October 2021, a Turkish computer user received an email with the attachment, an XZ archive file. This contains a disguised executable, “ürün örnekleri resmi pdf.exe” (which translates to “product samples image pdf.exe”). The executable was a self-extracting malware dropper built with the Delphi programming language (known as “BobSoft Mini Delphi”).

The dropper, in turn, installed several executables. The first was a legitimate Microsoft Visual Studio component (msbuild.exe). MSBuild is normally used to compile and execute coding projects; it can be passed project files or XML files containing scripts on the command line and launch them Since the file is a trusted Microsoft binary, it can be packed into a dropper to mask the malicious nature of the malware.

The second executable was retrieved from the Discord content delivery network and decrypted: It was the Phoenix keylogger, an information stealer. Also dropped at some point was QuasarRat, a remote access tool written in C#.

Over the next week, the attacker used the installed QuasarRAT to launch the Phoenix info stealer and execute commands through MSBuild. The commands built and executed by MSBuild accessed the cookie files on the targeted machine.

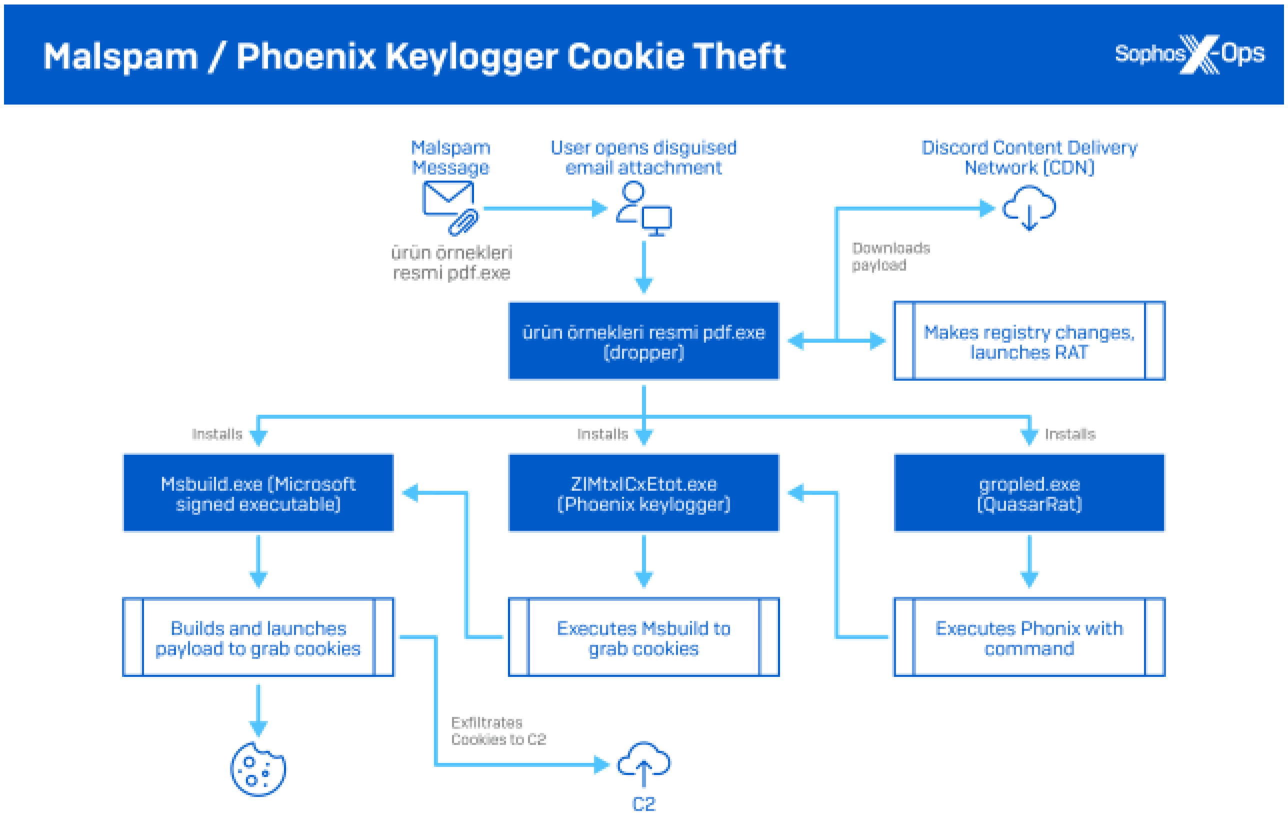

Malspam / Phoenix Keylogger Cookie Theft
Sophos X-Ops
Malspam Message
ürün örnekleri resmi pdf.exe
User opens disguised email attachment
Discord Content Delivery Network (CDN)
Downloads payload
ürün örnekleri resmi pdf.exe (dropper)
Makes registry changes, launches RAT
Installs
Installs
Installs
Msbuild.exe (Microsoft signed executable)
ZIMtxICxEtot.exe (Phoenix keylogger)
gropled.exe (QuasarRat)
Builds and launches payload to grab cookies
Executes Msbuild to grab cookies
Executes Phonix with command
Exfiltrates Cookies to C2
C2

Figure 4: Portrait of a Malspam / Phoenix theft

Targeted exploitation

Stealing cookies is not just an automated activity. In some cases, it's also part of efforts by active adversaries seeking ways to deepen their penetration of a targeted network. In these cases, the attackers use a foothold on the network to deploy exploitation tools and use those tools to spread their access. As more data of value has moved off the network and into cloud services, these attackers have added lateral movement to those services through cookie stealing and the scraping of web login data to their list of things to do.

We discovered an extended intrusion of this sort active in June 2022, in which cookie stealing was part of ongoing Cobalt Strike and Meterpreter activity stretching back months. The attackers specifically targeted cookies in the Microsoft Edge browser. First, they were able to use the Impacket exploit kit to spread from the initial point of entry via Windows SMB file transfer, dropping Cobalt Strike and Meterpreter on targeted computers within the network.

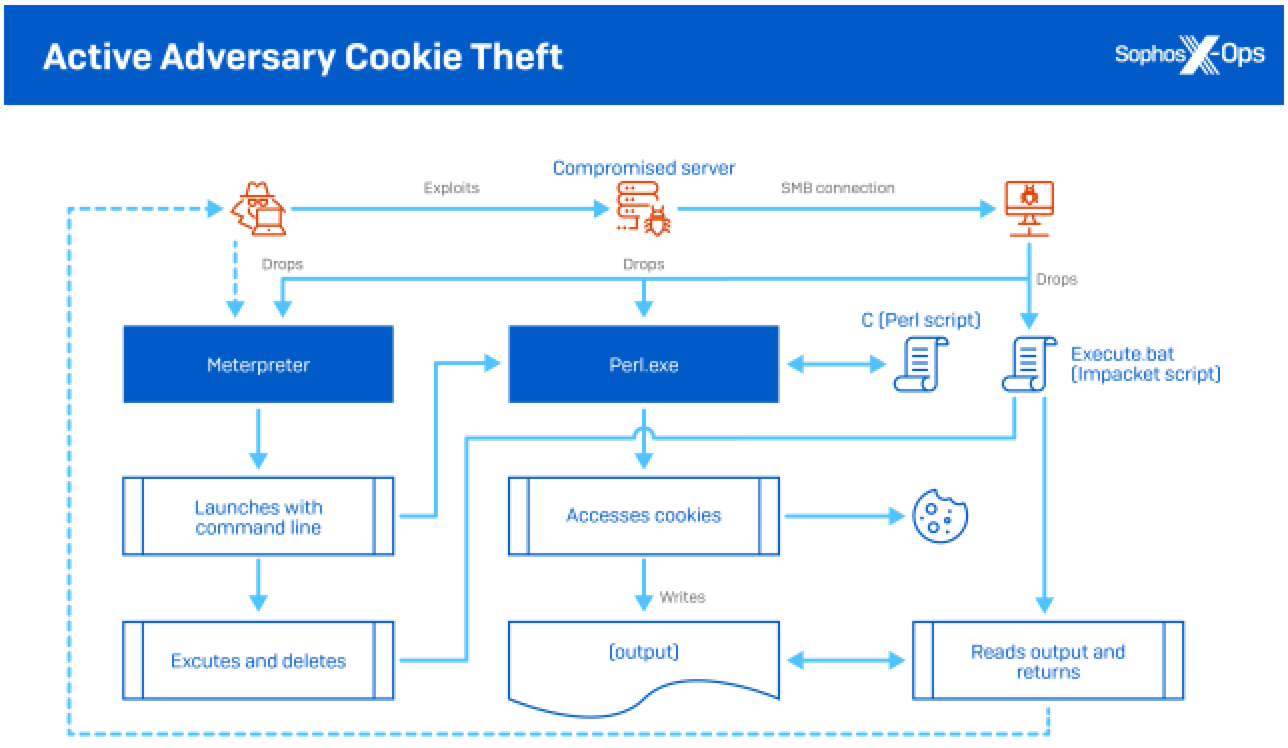


Figure 5: A hands-on cookie theft sequence

Next, the attackers dropped a copy of a legitimate Perl script interpreter on the targeted system, along with a Perl script file (named c) and a batch file (execute.exe) seen in previous Impacket-based intrusions. They then used Meterpreter to pass the following command string:

```
c:\\windows\\vss\\perl.exe -type c ^> \\\\127.0.0.1\\C$\\__output 2^>^&1 >
C:\\WINDOWS\\TEMP\\execute.bat & C:\\WINDOWS\\system32\\cmd.exe
/Q /c C:\\WINDOWS\\TEMP\\execute.bat & del C:\\WINDOWS\\TEMP\\execute.bat
```

The Perl script accessed the cookie files on the target machine and output the contents to a temporary file named _output. The batch file, among other things, transferred the contents of _output back to the attacker and deleted the Perl script. The remaining shell commands turned off screen output, deleted the batch file, and terminated the command shell.

Keeping a lid on cookies

These three examples represent just part of the cookie-stealing cybercrime spectrum. Information-stealing malware increasingly include cookie theft as part of their functionality, and the success of marketplaces that make the sale of stolen cookies a viable business. But more focused attackers are targeting cookies as well, and their activities may not be detected by simple anti-malware defenses because of their abuse of legitimate executables, both already present and brought along as tools.

Cookie theft would not be nearly as much of a threat if long-lived access cookies weren't used by so many applications. Slack, for example, uses a combination of persistent and session-specific cookies to check for users' identity and authentication. While session cookies are cleared when a browser is closed, some of these applications (such as Slack) remain open indefinitely in some environments. These cookies may not expire fast enough to prevent someone from exploiting them if they're stolen. Single-sign-on tokens associated with some multifactor authentication can pose the same potential threat if users don't close sessions.

Regularly clearing cookies and other authentication information for browsers reduces the potential attack surface provided by browser profile files, and organizations can use administrative tools for some web-based platforms to shorten the allowable timeframe that cookies remain valid.

But strengthening cookie policies comes with trade-offs. Shortening the lives of cookies means more re-authentication by users. And some web-based applications that leverage clients based on Electron or similar development platforms may have their own cookie-handling issues; for instance, they may have their own cookie stores that can be specifically targeted by attackers outside of the context of web browser stores.

Sophos uses several behavioral rules to prevent abuse of cookies by scripts and untrusted programs and detects information-stealing malware with a number of memory and behavior detections.

Indicators of compromise for the attacks mentioned in this report are available on the SophosLabs Github page.

Sophos X-Ops thanks Felix Weyne of SophosLabs for his contributions to this report.