

Trainingsszenario - Prozessinjektion

Gruppenname

Trainingsdaten - Prozessinjektion (Hive)

Szenario

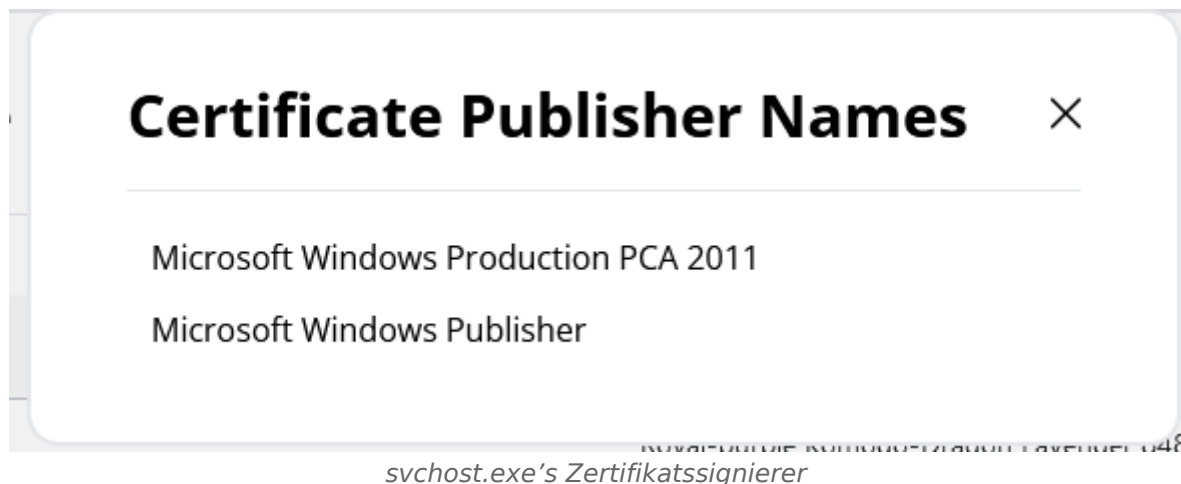
In diesem Trainingsszenario führen wir eine Ransomware-Payload mittels Prozessinjektion in einen bereits laufenden Svchost.exe-Prozess aus. Der betreffende Svchost ist ein normaler Systemvorgang, läuft seit längerem, ist signiert und wird unter dem LocalSystem-Benutzerkonto ausgeführt.

Da der Svchost „echt“ ist, wird er oft übersehen. Administratoren wissen, dass viele Instanzen von Svchost ausgeführt werden, und solange die Argumente des Programms korrekt aussehen, wird er als „Black Box“ behandelt und in Ruhe gelassen. Dies ist eine offensichtliche und häufig ausgenutzte Schwachstelle, die Hacker sich zunutze machen. In Kombination mit einer Zero-Day-Payload ist dieser Ansatz sehr erfolgreich darin, Ransomware-Angriffe zu verschleiern.

Erkennung

Number of Incidents	Machine Name	Program Path	Program Arguments
▼ 1	\\CC-NOAH-TEST	C:\Windows\System32\svchost.exe	-k UnistackSvcGroup

Created	Response Name	Untrusted Remote Threads	Modified Memory	No Trusted Cert
October 18, 2022 at 9:16:41 PM EDT	 Royal-purple Komc	True	False	False



Die Situation rund um diese automatisierte Reaktion ist auf den ersten Blick nicht direkt erkennbar. Die Kommandozeilenargumente sehen für Svchost gültig aus, und die ausführbare Datei ist von Microsoft signiert.

Die Tatsache, dass ein nicht vertrauenswürdiger Remote-Thread vorliegt, ist verdächtig, erfordert jedoch weitere Untersuchung. Viele Remote-Threads treten auf einem System auf, aber die meisten davon bleiben „vertrauenswürdig“, da sie für normale Interprozesskommunikation, Datenaustausch usw. verwendet werden.

Wenn ein Remote-Thread auf ungewöhnliche Weise in einen Prozess erstellt und als nicht vertrauenswürdig markiert wird, ist genauere Prüfung erforderlich.

Injector Pid	Injector Path	Injectee Pid	Injectee Path ↑
3992	C:\Windows\System32\csrss.exe	8844	C:\Windows\System32\cmd.exe
4228	C:\Windows\System32\dwm.exe	3992	C:\Windows\System32\csrss.exe
3268	C:\Windows\System32\dwm.exe	5972	C:\Windows\System32\csrss.exe
4228	C:\Windows\System32\dwm.exe	3992	C:\Windows\System32\csrss.exe

Harmlose Prozessinjektionen, die normalerweise auftreten

Um die vielen Prozessinjektionen zu sortieren, die zu einem bestimmten Zeitpunkt auf einem System auftreten, können wir uns sowohl auf den Zeitpunkt des Vorfalls als auch auf die PID des markierten Prozesses beziehen. Normalerweise beginnen wir mit einigen PIDs verdächtiger Reaktionen und arbeiten uns die Kette der Injektionen und Erstellungen hinauf, um die vollständige Geschichte zu erfassen.

Response Details



Description	Value
Response Name	Royal-purple Komodo-Dragon Lavender 848
Machine Name	\\CC-NOAH-TEST
Program Path	C:\Windows\System32\svchost.exe
Program Arguments	-k UnistackSvcGroup
Agent Name	
Ram usage	0
File access count	0
Number of threads	0
Pid	6224

[View Certificate Publisher Names](#)

Injectee Pid ▼

6224

≡

▼

☰

Equals ▼

6224

Reset

Apply

Herausfiltern irrelevanter Injektionen

Injector Pid	Injector Path	Injectee Pid ▼	Injectee Path ↑
9168	C:\svchost_injector.exe	6224	C:\Windows\System32\svchost.exe

Der eindeutige Beweis!

Wir wissen bereits, dass niemand auf diese Weise in Svchost injizieren sollte, aber ganz sicher nicht dieser Prozess! Von hier an können wir diesen Prozess so behandeln, als wäre es „malware.exe“, und herausfinden, wer ihn ausgeführt hat, da es jemand gewesen sein muss.

Created ↑	Child Pid ▾	Parent Path	Child Path
October 18, 2022 at 9:11:09 PM EDT	9168	C:\Windows\System32\services.exe	C:\Program Files (x86)\Dropbox\Update\DropboxUpdate.exe
October 18, 2022 at 9:16:04 PM EDT	9168	C:\Windows\explorer.exe	C:\svchost_injector.exe

Prozesse, die mit PID 9168 gestartet wurden

Da es zwei Prozesserstellungen mit PID 9168 gab und diese sehr unterschiedlich sind, können wir dies anhand des Zeitstempels und/oder des untergeordneten Pfads eingrenzen und feststellen, dass er von einem explorer.exe erstellt wurde. Das bedeutet, dass jemand den Injektorprozess manuell ausgeführt hat, und es kann kein Zufall eines fälschlicherweise markierten Svchost-Verhaltens sein.

Relevante Dokumentation

- **Prozessinjektion**

- [Mitre T1559](#) - Interprozesskommunikation kann dem Injektor nach Abschluss der Injektion Kontrolle über den Zielprozess ermöglichen.
- [Mitre T1106](#) - Native APIs bieten oft den direktesten Zugriff auf Betriebssystemfunktionen zum Zugriff auf Dateien, laufende Prozesse und mehr.
- [Mitre T1569](#) - Die Injektion in einen Systemdienst wie einen bestehenden Svchost kann böartigen Code so tarnen, dass er als von einem bekannten und vertrauenswürdigen Prozess ausgeführt gemeldet wird.
- [Mitre T1055](#) - Prozessinjektion, üblicherweise verwendet, um böartigen Code in einem Zielprozess auszuführen, während der ursprüngliche Prozess weiterlaufen kann.

Revision #1

Created 2026-07-24 03:56:06 UTC by Dennis Underwood

Updated 2026-07-24 03:56:06 UTC by Dennis Underwood