Microsoft HTML Application (HTA) Abuse, Part Deux

By Keith McCammon, Chief Security Officer

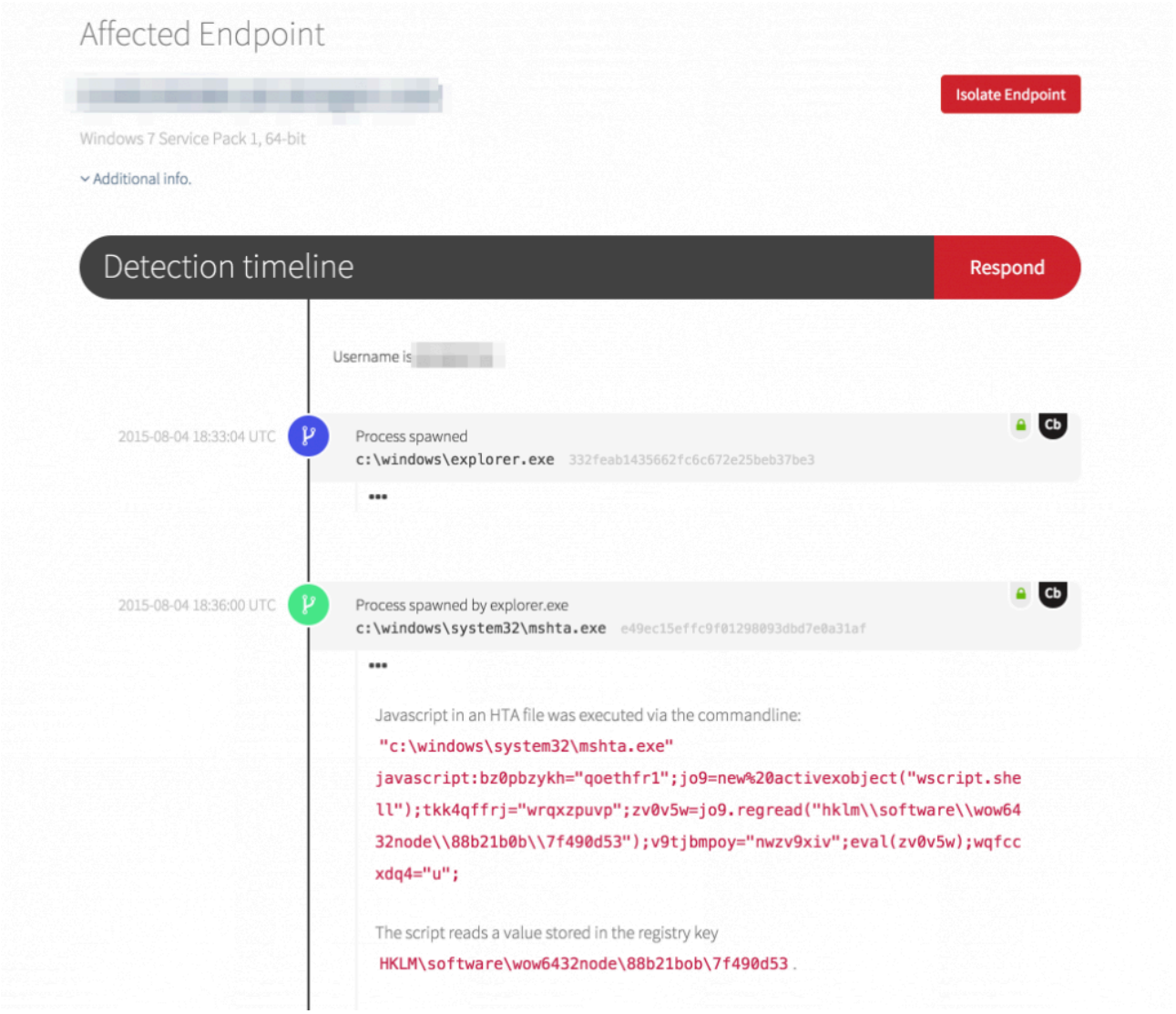
Published: 2015-08-14 · Archived: 2026-04-06 01:51:00 UTC

In our most recent Detection Profile, we looked at a red team's post-exploitation activity as detected by Red Canary. The tool was identified through open sources as PoshRat, a PowerShell-based remote access tool that takes advantage of a security policy bypass in Microsoft HTML Applications (HTA) to establish a reverse shell.

Unfortunately, HTA abuse is widespread and not limited to use by red teams. The Red Canary SOC continues to observe HTA abuse in the wild as well. Here we'll look at one such detection, wherein a malicious HTA file is leveraged to complete an exploitation chain without dropping any binary files to disk.

First, we see mshta.exe called with a tell-tale command line containing both overt calls and some encoded values:

```
"c:\windows\system32\mshta.exe" javascript:bz0pbzykh="qoethfr1";jo9=new%20activexobject("wscript.she
```



Note the call to regread(), wherein the routine picks up a binary object from registry key hklm\software\wow6432node\88b21b0b\7f490d53. This entry would have been dropped by a preceding process, likely rundll32.exe.

The product of this encoded routine is a Powershell child process used to execute a script using iex, an alias for the Invoke-Expression cmdlet. The iex cmdlet takes the next bit of input, a command that is stored in an obscure environment variable, and executes it. What we see coming from the Carbon Black sensor is:

```
"c:\windows\syswow64\windowspowershell\v1.0\powershell.exe" iex $env:veckxeg
```

The entire sequence of events is repeated a second time almost immediately, though the second iteration uses runonce.exe as the triggering process.

The specific malware family observed is subject to some interpretation. The techniques described above are indicative of known variants of Poweliks as well Win32/Xswkit (a.k.a. Gookit). Both of these malware families are notable because they reside in the registry or in memory, and do not rely on typical exploitation chains that utilize overt dropper binaries before migrating into system processes. They also tend to evolve rapidly, and thus antivirus or even anti-exploitation tools cannot be relied upon for prevention.

Another win for a solid visibility tool coupled with broad detection, particularly observation of trusted system processes.
