CYBERSECURITY

What Sauron Tells Us About What NSA's Up To, and What It Should Do Next

By Nicholas Weaver Monday, August 15, 2016, 2:52 PM

DayZero: Cybersecurity Law and Policy

Last week, both Symantec and Kaspersky released a series of reports on a nation-state malware attacker dubbed alternately "Strider" or "Sauron." Although neither formally attributes the attack to any particular country, reading between the lines it is pretty clear: The U.S. was Here.

Attributing malcode is always questionable, and there is some possibility this is not the (rather excellent) work of the NSA but instead some the efforts of a FVEY ally. But—given the quality, the modular nature, target selection, and lots of other details—the smart money is on this being from our friends at Fort Meade. If we assume that it is, what does this mean?

This Code Has Legacy

The code reveals an interesting legacy. Sauron appears to be the product of a long-lived, continually evolving, large-scale development. While the bulk of the code may be new, there are indications that a number of components—notably those protocols using RC5 and RC6—arose prior to 2001 while other pieces kept receiving enhancements.

The centerpiece of the Sauron codebase is a Lua framework similar to the one in Flame (another piece of malcode widely attributed to the NSA) but enhanced with features not present in the general Lua language. Lua is a moderately obscure scripting language developed in the 1990s, although the version used in Sauron is newer than 2000 because it contains features which were only added in Lua 4.0 (released in 2000). Lua is relatively small, with interpreters only occupying 100 kB. Although slightly obscure, it is a good choice for a malcode framework, or at least it was a good choice several years ago—today, one might prefer Python which is larger but substantially more powerful and flexible.

Also telling is a note buried in one of the Lua functions. The custom Lua modules contained documentation strings, small helpful notes for those using the Sauron framework to develop their attacks. One function, *sinfo*, specifically warns to "use this option with care since it causes cdrom-spinup, little network traffic and floppy-noise." When was the last time you saw a floppy disk?

Other components speak to an even older history. Various encryption libraries include the RC6 and RC5 encryption algorithms, which are sensible but still strangely obsolete choices. RC5 dates back to 1994 and RC6 succeeded it in 1998. There is no real reason to use RC5 after RC6's definition. Furthermore, RC6 was a candidate for the AES standard, a federal standard for cryptography, and there was no sense using RC6 after 2001 when the AES standard group decided on Rijndael. Another component uses the Salsa20 stream cypher developed by Daniel Bernstein in 2005. Again, this is a probably good cypher, but overcome by a superior successor Bernstein developed in 2007, ChaCha20. Taken together, these aspects suggest older protocols. Even if you write new code, if you need it to interact with an older system, using an older cryptographic protocol, your new code will use the old algorithm.

Other components are substantially newer. The *xkgate* funciton installs a driver portion of avast! antivirus and then exploits a vulnerability in that driver to take control over the system. Since the version deployed is 9.0.2006.159, released in October 2013, this component clearly represents a far more recent development.

So, presuming this is NSA, we can glean that there is little desire to reinvent the wheel at Fort Meade. Instead, the agency is, at least in some cases, using components for long-periods while adding updated enhancements over time.

Your Tax Dollars Are Well Spent

Beyond the well-developed and long history, reports suggest that this code is also very well done. On most systems it only exists in memory, making detecting and analyzing an infection very difficult. To gain persistence, it generally seeks to take over control of the systems used to manage systems, enabling the attacker to quickly re-infect everything in an institution without leaving traces.

Other aspect point to a huge amount of engineering for stealth. It contains multiple mechanisms to exfiltrate data, including DNS requests (the portion of the Internet that turns names like "www.lawfareblog.com" into numbers), HTTP, and email. All these methods for data exfiltration took great care to look like normal traffic in order to resist detection.

The DNS exfiltration mechanism is particularly interesting. Medium volume (over 4kB or, to speak in terms Lawfare readers understand, a little over one "billable page" in a PACER docket report) of data exfiltration over DNS is readily detectable. But small requests, such as just reporting system information or checking to see if there are commands to execute, is effectively impossible to detect today. The Sauron authors appeared aware of the state of the art, as the framework is careful to ensure that the DNS queries are effectively undetectable.

Other effective techniques include creating hidden filesystems on USB devices to cross air-gaps and payload specifically engineered to attack an undisclosed virtual private networking setup belonging to a target. The Sauron users also demonstrate excellent operational security with every instance appearing different in nearly every possible way, with different domain names, control systems, exfiltration channels, and file hashes.

There is only one minor bug disclosed in the Kaspersky report, a failure to cleanup some temporary files when exfiltrating data. Overall the attention to detail and care speaks well of the developers and users. You can object to the NSA's activities, but no one can fairly claim it is doing anything less than a spectacular job.

Your Civil Liberties Are Safe

Another striking aspect of the malcode is its rarity. This isn't something everyone—or even every FSB member—might expect to encounter on their system. Kasperski reports only a few high-value targets infected in Russia, Iran, Rwanda, and possibly an Italian speaking country, while Symantec also adds China, an embassy in Brussels, and an unstated organization in Sweden. Symantec only discovered 36 infections in a total of 7 organizations

Whoever is responsible for target selection took great care—this is an example of the kind of spying that should be encouraged over mass collection: targeted espionage against a few, high-value targets.

US Diplomats Are Going To Be Annoyed

The fact that everyone spies, does not make getting caught any less embarrassing. The identified institutions are now aware that the NSA was in their systems—with only the thinnest of plausible deniability—and their reactions still remain open questions.

Attributing Sauron here does not just apply to the current version but to future variants as well. NSA could easily make a new variant, Sauron 2.0, that would bypass the detectors which will now be constructed to find Sauron. But that leaves NSA with a predicament: While Sauron 2.0 might evade current detection, if it ever is discovered it will be attributable to the NSA.

Starting anew would will not necessarily help. Let us suppose that the NSA decides to completely replace the infrastructure: eliminating the old RC5/RC6 protocols, switching from Lua to python, and otherwise starting over. The effort would cost millions of dollars and result in a framework no stealthier than a hypothetical Sauron 2.0. And even this new framework, if detected, could probably also be attributed to the NSA. The complete rebuild eliminate some of the clearer relationships to captured NSA malcode, but the workmanship, combined with the language of the many developers and inevitable bits of metadata leaked out during the development process, would still scream FVEY. The Five Eyes in general, and NSA in particular, is just too good to not recognize their A game when detected, it can’t plausible come from anywhere else.

Instead of deploying massive funds to replace Sauron with a Sauron 2.0, the NSA could take an alternative approach. The kind of stealthy and advanced engineering deployed in Sauron, results in high-quality malcode that is hard to detect but easy to attribute (by virtue of its quality). Commercial, off-the-shelf malcode, used by criminals and nation-states alike, is less stealthy but much harder to attribute to any given actor. And attribution to NSA would be near impossible when paired with the excellent operational security evidence in the Sauron campaigns.

Recognizing this, instead of replacing Sauron, NSA could consider distinguishing targets into two classes: those which maximum stealth and those for which plausible deniability is needed. Where stealth is paramount, the existing Sauron infrastructure, with suitable teaks, will accomplish the purpose. But where deniability is required, NSA should switch gears entirely. Instead of using bespoke malcode, the U.S. should simply combine the NSA's OPSEC with criminal-COTS malcode such as njRat or others, just like many other actors already do, which both saves a fortune and renders effective attribution impossible.

There are no hard-and-fast rules for where stealth versus deniability should be preferred. But policies should account for an important reality: The NSA can deploy highly stealthy attacks. The NSA can deploy nearly impossible to attribute attacks. But it probably cannot do both simultaneously.

Topics: Cybersecurity

Tags: National Security Agency (NSA)

Nicholas Weaver is a senior staff researcher focusing on computer security at the International Computer Science Institute in Berkeley, California. All opinions are his own.

@ncweaver

MORE ARTICLES ›

RELATED ARTICLES

A List of Female Technology Policy Experts

Susan Hennessey Tue, Aug 23, 2016, 1:45 PM

A DOJ Cybercrime Round Up

Matthew Weybrecht Fri, Aug 19, 2016, 1:47 PM

Everything You Know About the Vulnerability Equities Process Is Wrong

Dave Aitel, Matt Tait Thu, Aug 18, 2016, 2:46 PM

NSA and the No Good, Very Bad Monday

Nicholas Weaver Tue, Aug 16, 2016, 10:34 AM

More on Securing the Election

Carrie Cordero Mon, Aug 15, 2016, 1:40 PM