ExtraHop

TAXONOMY OF

SolarWinds SUNBURST DNS Abuse Tactics

EXECUTIVE SUMMARY

This report details a critical technique used in the SUNBURST attack to evade detection: hiding command-and-control traffic by taking advantage of known weaknesses with enterprise domain name systems (DNS). DNS is a popular attack vector both because of its ubiquity and its noisiness. The sheer volume of DNS queries make it extremely difficult to monitor and secure.

The SUNBURST malware got its communications in and out of the target's infrastructure via DNS, using a rarely-seen domain generation algorithm (DGA). This traffic wasn't caught by logging and other traditional security measures, leaving a troubling question for how to detect this kind of activity in the future.

DNS is prone to hijacking, cache poisoning and redirection attacks, along with DNS tunneling and other methods of abuse that can obfuscate to where DNS traffic resolves. The challenges in securing DNS make understanding its function and flaws of critical importance, so security teams can establish the best practices and tools necessary to prevent it from being used in future attacks.

TABLE OF CONTENTS

INTRODUCTION

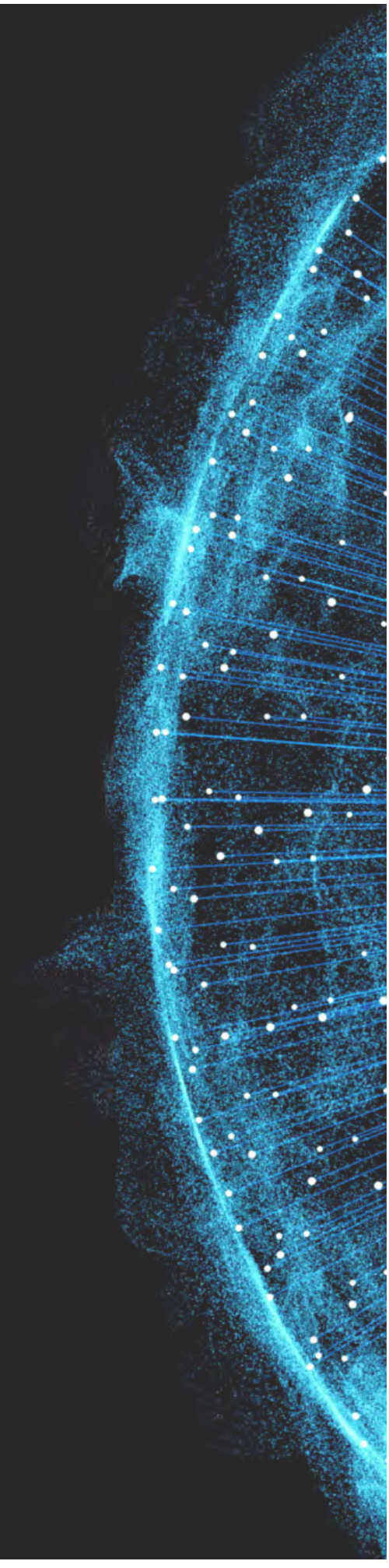
The SolarWinds SUNBURST attack reveals how a sophisticated and well-funded cyberattack can spread through the supply chain and linger for months without notice. Nation-state attackers broke into SolarWinds and inserted weaponized Trojans into the update build servers of SolarWinds Orion. The malware then infected the networks of customers as they installed the updates. Once loaded into downstream servers, SUNBURST opened backdoors and stayed hidden for at least eight months—possibly longer—until intelligence vendors alerted SolarWinds and then released news of the breaches in December 2020.

> *"This attack was a very sophisticated supply chain attack, which refers to a disruption in a standard process resulting in a compromised result with a goal of being able to attack subsequent users of the software. In this case, it appears that the code was intended to be used in a targeted way as its exploitation requires manual intervention."*
> *SolarWinds Security Advisory, January 29, 2021*

As of this writing, projected insurance losses from SolarWinds SUNBURST breaches are expected to reach $90 million USD. Around 18,000 companies received infected updates. At least 100 US companies were breached downstream from the updates.

Once installed, the SUNBURST Trojan hid its command-and-control (C2) activities by taking advantage of known weaknesses with enterprise domain name systems (DNS). Attackers know that, with millions of DNS requests and queries in a given day, DNS traffic and queries are difficult to log, and log management can't scale. So, they hid their DNS activities in all this noise and carefully timed queries and traffic to fly under the radar. The SUNBURST Trojan also manipulated DNS queries to identify prized systems to copy out of the organization, exploit DNS resolution issues and data link libraries (DLLs), and route outbound traffic from infected systems through seemingly trustworthy registrars and domains.

In this white paper, we break down the SUNBURST DNS tactics used to obfuscate what should be detectable network activities and hide C2 communications and sensitive data transfer.



SUNBURST Methodology

Identity and access management (IAM) should have been a top priority in the DevOps environment.

It's unclear how SolarWinds was initially breached. Early reports indicated that the breach started in September 2019 with an Office 365 account belonging to a SolarWinds employee. However, Microsoft debunked that theory in February 2021, saying it could find no evidence of Microsoft products used to breach the SolarWinds network as attempts didn't succeed.

A SolarWinds advisory says that SUNBURST originated from multiple servers in the US and mimicked legitimate network traffic that antivirus tools and firewall services didn't detect. Once in SolarWinds, the attackers got into the Orion build servers through an easy-to-guess shared password. The advisory also said the attackers were able to further "circumvent threat detection techniques employed by both SolarWinds, other private companies, and the federal government." Security teams failed in their responsibility to secure the SolarWinds DevOps environment, where secure identity and access management (IAM) should have been a top priority.

Low, Slow, Patient

If we're to believe the breach started in September 2019, the attackers took five months inside SolarWinds to test and compile the malicious code and insert it in the Orion build updates (specifically the Orion platform builds for versions 2019.4 HF 5 and 202.2 and 202.2 HF, according to the SolarWinds advisory). It then waited another month until the updates were made available to SolarWinds Orion clients. From March until June of 2020, SolarWinds served up the malware through the Orion updates. In June, attackers and the malware retreated from the Orion VM builds. By then the SUNBURST Trojan had infected at least 18,000 customers who deployed the update in their environments. See the timeline:

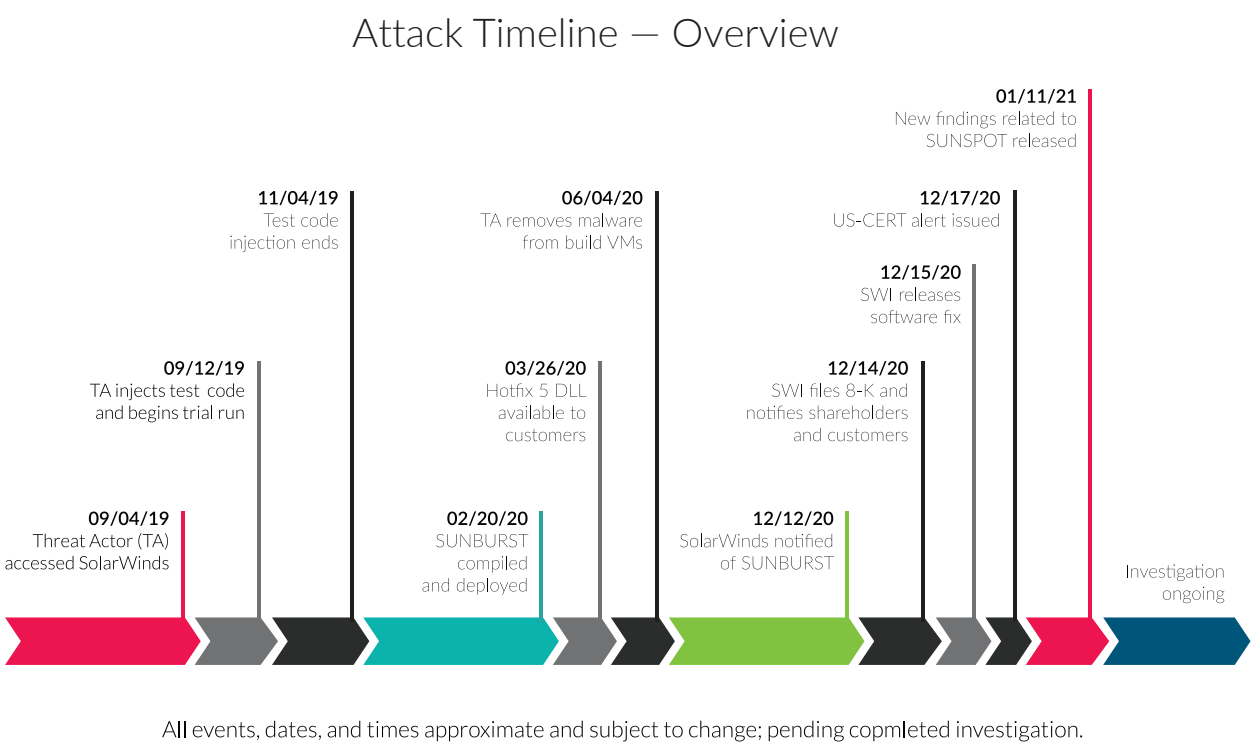


Figure 1. Timeline and Details of SolarWinds SUNBURST Supply Chain Attack (source: Channele2e)

Knows How to Hide

As seen from the timeline figure above, SolarWinds was notified of the SUNBURST downloads eight months after SolarWinds began delivering the infected updates. A patch for infected customers was created, and on December 17, 2020 the US-CERT issued a notification. So, the attackers who infiltrated networks along this supply chain had six to eight months inside their victim organizations before the news went public.

There are a number of reasons this attack held on so long without detection, including:

1. **Too Much Trust in the Patch and Update Process.** Most organizations trust their patches to be free of malware because they come directly from their software suppliers.
2. **Circumvents Pre-Deployment Testing.** The backdoor encoded into the update stayed dormant for at least two weeks, so it wouldn't be detected during testing and deployment.
3. **Innate Security-Awareness:** According to FireEye's initial analysis, the SUNBURST malware uses "multiple obfuscated blocklists to identify processes, services, and drivers associated with forensic and anti-virus tools" to avoid these detection tools or to turn them off.
4. **Sneakily Sets up C2 Pathways:** The attackers took advantage of DNS services and blind spots of DNS services and blind spots to establish itself, spread to internal IP addresses, and send out data through C2 channels to rogue servers.

Subverts DNS

Using DNS to enable and hide C2 communications is the most interesting aspect of the SUNBURST malware payload. SUNBURST DNS tactics began as soon as devices were infected, and they started carefully trying to reach external C2 servers. In a rarely used attack method, the SUNBURST backdoor uses a DGA to hide C2 traffic inside DNS. According to a January blog from Symantec, this subversion of DGA enabled attackers to identify each infected computer sending information to the C2 servers by using a XOR cipher to encode system identifiers in the first 14 characters of the DNS queries.

When they received the encoded user IDs, they decrypted them with the cipher and read the characters to determine which systems they wanted to launch secondary attacks against. Most of those secondary attacks were against critical infrastructure and security software vendors that could perpetuate the attack and cause the most damage.

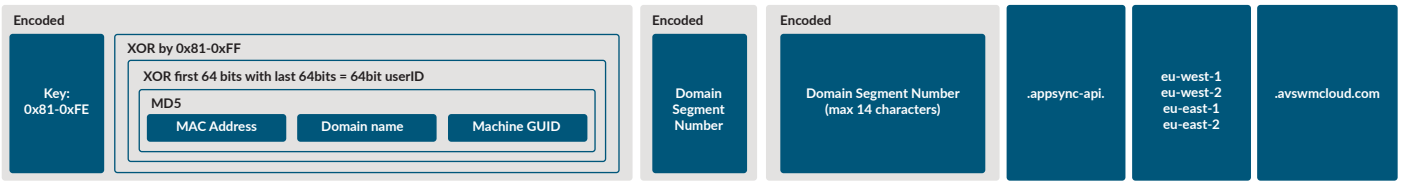


Figure 2. Structure of SUNBURST DNS lookups (source: Symantec)

Despite the sophisticated use of DGA to hide illicit SUNBURST traffic, on December 11, 2020, threat researchers from FireEye announced that, as part of their own breach investigation, they had uncovered an exploit within the SolarWinds Orion platform. On December 13, they confirmed that this exploit was being used by a nation-state adversary to perpetrate wide-spread attacks. They coined the name "SUNBURST" to describe the threat. SolarWinds subsequently confirmed the corruption and weaponization of its Orion software.

Once the initial exploit was uncovered, numerous security vendors and agencies went to work identifying indicators of compromise (IOCs) using techniques like DNS log analysis and static/dynamic analysis. Within days, Microsoft and a coalition of security vendors located the attacker's primary domain, **avsvmcloud[.]com**. Working with GoDaddy, Microsoft and FireEye configured a new A record for **avsmcloud[.]com** to resolve to an IP that deactivates the malware on the victim system. They accomplished this by turning the smart DGA program used by SUNBURST against itself. Specifically, they found IP blocks that the Trojan would automatically kill connections with, including a large block of IPs belonging to Microsoft. They reconfigured the A records on the C2 server to kill all connections by inserting a Microsoft IP and resolving all connection requests to that IP. Working with the FBI and the Infrastructure Security Agency, Microsoft and FireEye are also helping identify and inform victims that their IPs are infected with the SUNBURST Trojan so they can find and erase the malware from their systems.

Time Is of the Essence

After taking over the SUNBURST **avsvmcloud** domain, Microsoft locked it down to stop it from infecting new systems and communicating with victim systems. Once the primary domain had been taken over and locked, scanning the patch updates made it possible to detect and block the Trojans from loading into victim systems—an important lesson in the event that future attacks use similar mechanisms. If the attack isn't detected at that stage, then IT teams need to stop it before it communicates and sends home sensitive data over C2 servers. Figure 3 shows the kill chain progression from initial infection to C2 communications and data removal.

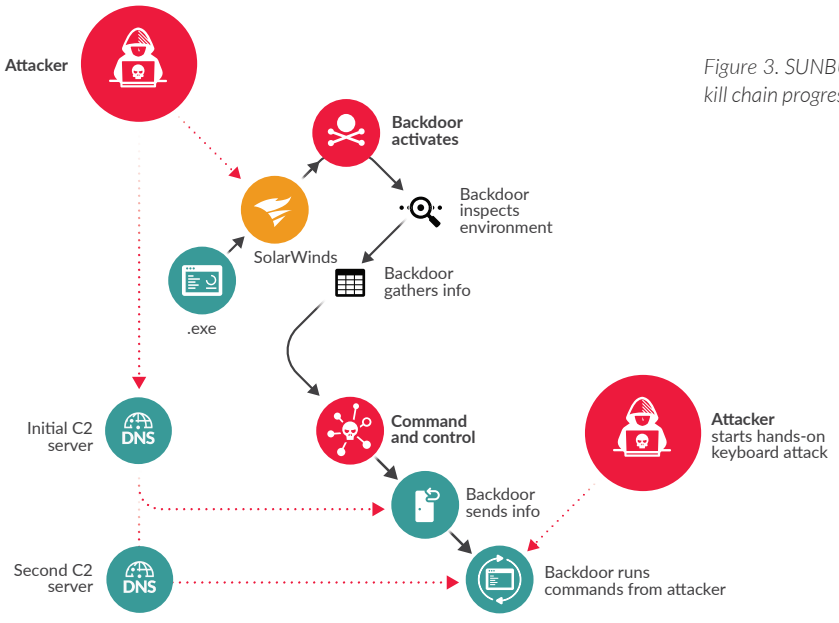


Figure 3. SUNBURST kill chain progression

Stopping the SUNBURST Trojan and others like them from establishing C2 communications isn't easy given that this particular attack is smart enough to know when it's being watched and goes dormant at the first sign of being observed. Manually comparing DNS deviations against historic traffic in passive DNS lookups is helpful and won't usually tip off the malware that it's being observed; however, organizations still need to know what indicators to look for to make the proper comparisons.

Dissecting SUNBURST DNS Exploitation

DNS is a vital enterprise service that acts as a gateway between connection requests from the internet and company IP addresses, which DNS keeps private while passing along requests. As embedded infrastructure, DNS is often considered a "set it and leave it," technology. Unfortunately, this can render it overlooked and underprotected.

> *"DNS is a hierarchical naming system for domains and other Internet resources. DNS can be viewed as an address book for the Internet; a primary function of DNS is mapping domain names to host IP addresses. DNS is maintained as a distributed database system that employs a client/server model. Resolvers (client programs) query the database for information. Name servers (server programs) respond with information obtained from locally stored resource records." –Google domains help site*

Soft Targets

According to VeriSign, DNS root servers negotiated 84 billion queries a day in 2020 on average. For enterprises, that number is usually in the millions, making DNS traffic almost impossible to log and easier for SUNBURST to hide within.

By the time suspect domains hit the intelligence feeds, they've already engaged in malfeasance

DNS is prone to hijacking, cache poisoning and redirection attacks, along with DNS tunneling and other methods of abuse that can obfuscate to where DNS traffic resolves. DNS systems do not notify administrators of traffic anomalies and deviations, making this attack useful for malicious actors to hide their traffic between infected IPs and domains outside the organization.

DNS-related intelligence is of little help. These intelligence feeds provide insight into known DNS threat activity, meaning that by the time a domain lands on one of the lists, they have likely already engaged in malfeasance. This problem was well illustrated by SUNBURST. The suspect domain was benign and sat dormant for several months before being activated. The first time it appeared on any DNS intelligence feeds was December, after SolarWinds was alerted to the exploit. By the time it was included in any feeds, much of the damage had already been done.

SUNBURST Knows its DNS

In the SUNBURST attack, queries and domains resolved into seemingly legitimate traffic from seemingly legitimate domains. Most of these domains were hosted at secondary providers that also looked legitimate. This enabled the SUNBURST malware to breach and take root in victim organizations without notice.

The following is a step by step walkthrough of the SUNBURST DNS exploits used to set up C2, communicate between infected devices, and identify IPs of interest for further inspection offline:

- Orion unknowingly distributed a standard Windows Installer patch file that included compressed resources associated with the Orion update that included the hidden Trojan component, **SolarWinds.Orion.Core.BusinessLayer.dll.**

- Once it downloaded into victim systems, the SUNBURST Trojan remained dormant for 10 to 14 days before beginning to establish communications with its malicious domain, **avsvmcloud[.]com.** *(Note how easy it is for the human eye to swap out AVS for AWS.)*

- Part of the malware included a malicious DLL that was loaded by the legitimate **SolarWinds.BusinessLayerHost.exe** or **SolarWinds.BusinessLayerHostx64.exe** depending on system configuration.

- If the initial ping to the domain was successful, SOLARWINDS sent queries and programs low and slow, in 30-minute increments to set up the redirects and communicate without notice.

- DNS was then leveraged to establish the domain redirects in several steps:

  - The queries attempted to resolve to a subdomain of **avsvmcloud[.]com.** But the recursive DNS server is not authorized to resolve **avsvmcloud[.]com,** so it forwarded the request.
  - Then an attacker-controlled authoritative DNS server resolved the request with a wildcard A record.
  - The attacker checked the victim's domain name, then added a CNAME record for the victim's domain name.
  - The DNS response returned a CNAME record that pointed to the C2 domain.

What's in a (C) NAME?

DNSimple
From the DNSimple record editor administrators can add, remove, and update CNAME records.

CNAME
CNAME records in Windows DNS servers point to another domain name (example.com), but never directly to an IP address number. These are often used as secondary domains such as email servers.

A Record
An A record always points to the primary server IP address.

- Using DGA, the SUNBURST malware also created unique identifiers for the infected systems so that attackers could pick systems of value to which they wanted to return.

- They used a XOR cipher to encode these system identifiers, along with the status of security products on that system, into the first 14 characters of the DNS queries to avoid detection.

- At the C2 server, the attackers decrypted the identifiers with the VOX cipher, then scanned for the type of systems they wanted to launch secondary attacks against (primarily government, infrastructure, and software companies).

- Then they opened a secondary secure HTTPS communication channel to the primary C2 server.

- Sensitive data and intellectual property was sent encrypted to servers hosted at small, seemingly benign cloud providers for further analysis by the attackers.

See Figure 4, below, for a visual map of how these DNS redirects are established.

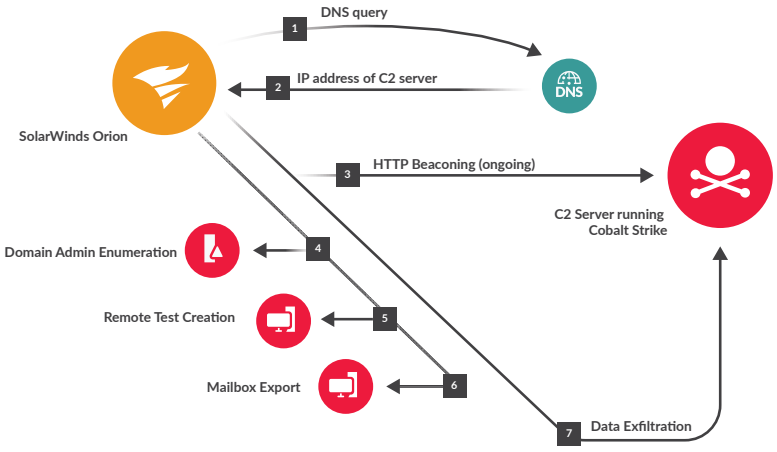


Figure 4. Topography of SUNBURST DNS redirect techniques

Hiding Behind Domains and Proxies

To reduce exposure, the goal is to detect the malicious DNS-related activity before the C2 servers are communicating with impacted IPs within the network. But detecting and understanding the intention of connecting domains is difficult even under the best of circumstances. There are several reasons IT security professionals and tools cannot see these rogue addresses. For example:

- They use clean domains that are not on blacklists. They specifically look for non-threatening domains with names related to the transactions.
- C2 domains resolve to seemingly benign, but questionable domain resellers and small cloud service providers, such as **chunkhost.com, liquidtelecom.com,** or **mivocloud.com;** as well as to external email providers and proxy services.
- Proxy services hide the identity of domains that the infected IPs resolve to, such as **insorg[.]org, safe-inet[.]net,** and **safe-inet[.]com.**

Prevent, Detect, Respond

In supply chain attacks, the kill chain starts with the supplier. If the attack on SolarWinds was detected early enough, SUNBURST operators and their malware programs would not have succeeded in accessing the SolarWinds build servers and uploading malware to SolarWinds customers through a trusted update. A supply chain kill chain would stop it there, but if not then, at the time the Orion patch update tried to download. If not there, then at testing, which should include a sandboxing component to detonate anything in a mock operational environment. Beyond that, a supply chain kill chain would block C2 activity before it establishes and starts sending out sensitive data.

See Figure 6, below for the supply chain kill chain.

Compromise Supplier	Lateral Movement	Compromise Customer	Lateral Movement	Command & Control	Exfiltration
In this step, the attacker uses techniques from the standard killchain to break into a supplier's system.	A normal attacker seeks valuable data. A supply chain attacker behaves differently—looking to compromise build servers or other channels into downstream environments.	Using software update paths or build servers, attacker achieves initial compromise of the supplier's customer.	This component is likely at least partially automated, as in the Raindrop malware dropper in the SUNBURST attack.	Standard C2 tactics.	Normal exfiltration and impact tactics.

This is enabled through standard lateral movement and command-and-control tactics, but the objective is different. Rather than targeting databases of valuable information, the attacker seeks systems within the supplier that can get the attacker access to downstream customer environments.

As of this writing, SolarWinds still doesn't know how the attack originated in their systems before it spread to the Orion update build servers. If the attackers managed to move laterally inside SolarWinds, they should have stopped them before they accessed the company's sensitive development servers. Unfortunately, that did not happen. The attackers infiltrated the SolarWinds build environment and SUNBURST Trojans were hidden in Orion updates signed by legitimate SolarWinds keys. In the supply chain attack kill chain, it's important to stop this type of attack from spreading (through a trusted patch or update to other devices in the network) and communicating to the external C2 server.

Another emerging practice is to encrypt DNS traffic—often referred to as DNS over HTTPS (DoH). This practice, in the enterprise, will inhibit threat hunters and incident responders from being able to hold systems to account. While an end user may want to hide what they are doing at home from their ISP, governments et al, creating opacity around DNS records may serve to provide bad actors a place where they can work in the dark, unless security tools are able to decrypt.

Show DNS Some Love

For a long time, DNS was hand curated, and manually edited ("named.conf" and zone files to ensure proper DNS resolution and PTR records, for example). Over the last few years, point-and-click DNS tools have made this administration easier but at the cost of proper DNS hygiene. Visibility into these misconfigurations is extremely difficult to accomplish with logs and windows events. As stated earlier, this difficulty in monitoring and the resulting lack of visibility has made DNS a preferred avenue for C2 communications as well as exfiltration. In the case of SUNBURST, programmatic decisions were made based on the returned values.

For this reason DNS needs proactive administration and protection. Shore up DNS servers by keeping them up to date, patched, and supported with security technologies like DNSSec (to prevent DNS reflection attacks) and DNS monitoring. For tighter control, restrict zone transfers, and turn off DNS recursion (enabled by default on most BIND servers on all major Linux distributions) to prevent poisoning attacks, among other best practices. Another emerging practice is to encrypt DNS traffic (often referred to as DNS over HTTPS (DoH)), but this is a two-edged sword because it blocks visibility to attackers and the host organization.

Network detection and response (NDR), unlike events or syslogs, helps identify and locate specific conditions that violate DNS hygiene. Examples of this include the aforementioned, unapproved DNS servers, DGA-like names, suspicious TLDs, and mismatched DNS answers (domain hijacking).

See What Changed

Know your typical DNS traffic and keep historic files for future lookup and comparison. With passive DNS, IT staff can flag changes to a domain's A record, for example, without alerting attackers because it's not happening live on their infected systems. DNS activity (traffic, queries, domains, and IPs) should be inspected for changes and anomalous behaviors in order to detect sophisticated C2 activity.

In the SUNBURST attack methodology, anomalies to look for include:

- A sensitive server like the Orion network management server contacting an external host for the first time.
- Connections from internal systems to external domains happening after hours.
- Unusual connections between internal systems (for example devops doesn't need access to the accounting system).
- Connections to external countries and locations outside routine DNS traffic and queries.
- Abnormal file reads.
- Increased SSH traffic with remote execution commands.

Decode DNS Traffic and Compare

If the traffic and queries seem suspect for any of the above reasons (or others not on the list), then it's time to pull the data aside and compare it to historic DNS data for deeper inspection. It should be decoded to metadata for keywords and suspicious domains as well as data leakage.

Once C2 channels are established and the secondary connection is made over HTTPS, it's difficult to determine what packets contain DNS requests or responses and what domains and IP addresses were requested. This means a network visibility solution must be able to decrypt and see into suspect traffic.

Using passive queries to search for changes, looking into the metadata to find keywords indicative of C2 communications, and comparing the metadata in previous snapshots of time are all processes that should be automated to improve detection and response.

Detect Domain Name Trickeries

This includes looking for misspelled and inaccurate domain names. For example, the SUNBURST attackers used a common technique called typosquatting to make domains seem legitimate and route victims through to the primary C2 server. Things to look out for include:

- Domain extensions outside the domain's normal geographic area, such as .au or .io that shouldn't be in the region they represent.
- Domains that look legitimate but use one or more letters out of place or add characters, such as in the SUNBURST case using **avsvmcloud[.]com** rather than **awscloud.com.**

- Appearing to sync to benign registrars or platforms but upon closer inspection are scammy and questionable.
- Subdomains point to generic sync interfaces, such as **appsynch.api** and appear as REST-based mobile app domains.

Correlations between network traffic, historic metadata, DNS connections and behaviors, along with drill-down capabilities into suspect traffic helps investigators know what to block and helps them build to the short list, focusing further investigation. From there, they can block the C2 channels and more easily locate impacted devices and make repairs. (See how ExtraHop tracked SUNBURST through metadata and other advanced monitoring techniques in the addendum provided by ExtraHop.)

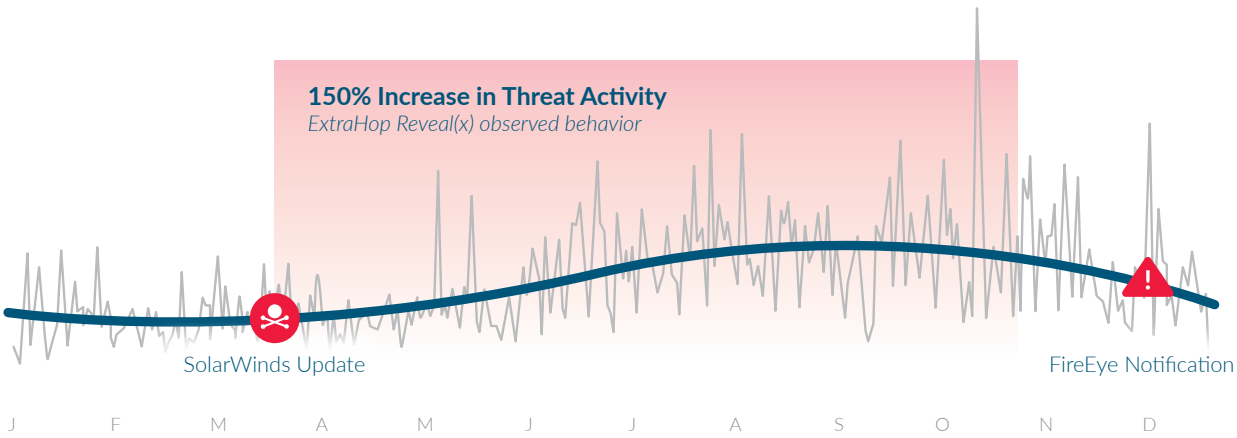
A Network View of SUNBURST

A note from ExtraHop on how Reveal(x) detected and addressed SUNBURST

When the SUNBURST attack was first revealed on December 13, organizations raced to identify and shut down affected SolarWinds binaries within their environments. Once the access points had been shut down, the investigations began in earnest as security teams tried to determine whether, and to what extent, their systems and data had been compromised. For many, the duration of the attack made this nearly impossible. Storing logs for months on end is cost prohibitive, and in any case, many of the devices––such as DNS, whose logs would have revealed malicious behavior—didn't have logging enabled to begin with.

But evidence of SUNBURST was evident in network metadata.

The following chart shows the threat activity detected—with anonymized, aggregate data from the many environments ExtraHop secures—between January 1, 2020 and December 19, 2020. Between late March and early October, detections increased by approximately 150 percent. The privacy protections ExtraHop maintains prevents this data from including destinations—it wasn't known that the increase in traffic was largely going to the same place.



The data shows that there was a significant and suspicious change in behavior on the network, including that of DNS. The magnitude of the increase in detections in the timeline aligns with the SUNBURST post-compromise activity at its height. It also demonstrates that the behavior of sophisticated attackers was—and is—visible on the network.

Conclusion

The SUNBURST attack that spread from SolarWinds to at least 18,000 downstream customers revealed it had more than 1,000 developer fingerprints once the malware was reverse-engineered. Now that it's in the wild and being reverse engineered, SUNBURST's methodologies will inevitably be packaged and used in different and more sophisticated ways.

The best security practices and regular risk assessments at the vendor or manufacturer pushing out the digital product are critical mitigations. However, as this attack proves, organizations down the supply chain need to be prepared for this type of attack to happen again. That means raising awareness about the risks in the software supply chain and its impact on software delivery, then establishing tools and best practices to detect these types of attacks as early in the supply chain kill chain as possible.

Deb Radcliff

About the Author:

Deb Radcliff has more than 25 years of experience in the cybersecurity industry, starting out as the first investigative journalist to make cybercrime a beat where she followed the FBI, DoD, Secret Service, CIA, local and state law enforcement as they were building their own cyber units. Her articles are cited in numerous research papers and college textbooks. She spoke at West Point, won two Neal Awards, and was runner up for a third. In 2005, Radcliff stood up a new Analyst Program for the SANS Institute and oversaw focused whitepaper and webcast content developed by SANS experts for 15 years. She still writes articles and blogs about cybersecurity, and, in April 2021, she published Breaking Backbones, a cyber thriller.