SolarWinds: Insights into Attacker Command and Control Process

symantec-enterprise-blogs.security.com/blogs/threat-intelligence/solarwinds-sunburst-command-control

Threat Hunter TeamSymantec

In the third of a series of follow-up analysis on the SolarWinds attacks, we investigate how the attackers controlled the Sunburst malware.

In our most recent blog on the SolarWinds attacks, we examined the domain generation algorithm (DGA) used to initiate contact with the attackers' command and control (C&C) servers. The control flow, what happens after that contact is made, is also noteworthy.

The control flow of Sunburst varies depending on commands received from the attacker. However, the general control flow can be reconstructed in order to understand how communications would have progressed on machines that were of interest to the attackers.

As described in our previous blog, two types of DNS requests are used for initial communications, and both receive DNS replies. The attackers use two fields in the DNS replies: "A records" for control flow and CNAME to hold data on a secondary C&C server.

IP addresses as commands

Normally, when querying DNS, a hostname string is provided to be translated into a numeric IP address, e.g., google.com may translate into 142.250.72.238. The IP address is held in the A record of the response. Sunburst parses the A record for IP addresses, but they are not used as IP addresses at all, but instead are actually triggers for different malware behavior. Instead of the attackers selecting random IP addresses to trigger different behaviors, they have selected IP address ranges belonging to Google, Amazon, and Microsoft. These are possibly chosen in order to reduce the chances of detection. Again, these IPs are not used as IP addresses in any way and the actual computer systems with these IP addresses are not contacted by the malware.

The IP address value received in the DNS reply represents one of five behaviors depending on the current state:

- Continue sending additional Windows domain name chunks
- Send product security status information
- Launch a secondary C&C channel
- Clean up and exit
- Reset state as if executing for the first time

For a typical infection, Sunburst will first send to the C&C server the first 14 characters of the Windows domain name. This will continue for each 14 character chunk, until the entire Windows Domain name is sent.

Next, the attackers will instruct Sunburst to send the current security product statuses and then send information enabling Sunburst to launch a more robust secondary HTTP-based C&C.

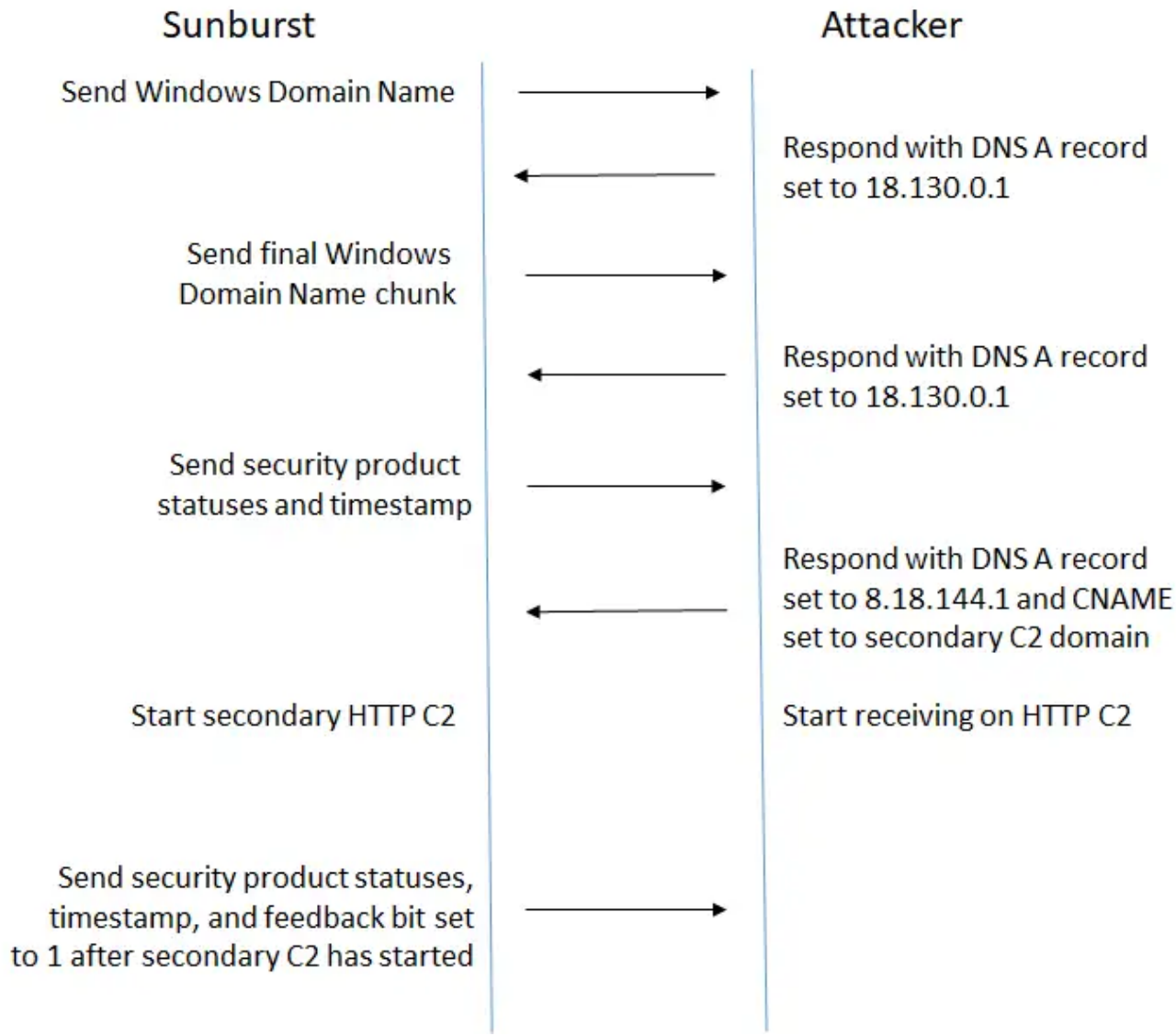

Figure 1. Example of C&C control flow

Within any DNS reply, the attackers can also instead instruct Sunburst to restore security product service registry keys that were previously disabled back to their believed default settings and quit or reset the internal state as if it is a fresh infection.

IP address range	Behavior
• 8.18.144.0/23 • 71.152.53.0/24 • 87.238.80.0/21 • 199.201.117.0/24	If not all Windows domain name segment chunks have been sent, continue sending Windows domain name segments Otherwise, send security product statuses and optionally launch a secondary C&C, if the C&C information has been provided
• 18.130.0.0/16 • 99.79.0.0/16 • 184.72.0.0/15	If not all Windows domain name segment chunks have been sent, continue sending Windows domain name segments Otherwise, send product security statuses and feedback bit immediately after launching secondary C&C or, if that fails, only after sleeping for 1-3 minutes
• 20.140.0.0/15 • 96.31.172.0/24 • 131.228.12.0/22 • 144.86.226.0/24	Clean up infection by restoring disabled security product service registry keys, if needed, and quit
• 41.84.159.0/24 • 74.114.24.0/21 • 154.118.140.0/24 • 217.163.7./24	Reset state as if it is a new infection

Figure 2. IP address ranges and the behaviors they cause

In addition, to facilitate lab environment and sandbox avoidance, if the DNS requests result in a private range IP address, including 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, 224.0.0.0/4, fc00::/7, fec0::/10, ff00::/8, Sunburst will also clean up and exit.

Second stage C&C

Victims that were of interest to the attackers would have reached the secondary C&C stage. In this case, the DNS reply would contain a hostname in the CNAME record to be used as a HTTP C&C channel. The CNAME value is checked to see if it starts with http:// or https:// and is prepended with https:// otherwise.

Next, Sunburst will perform repeated HTTP GET requests to receive commands, sleeping a variable amount of time between requests. The path portion of the URL always starts with /swip/upd/ followed by randomly selected URL patterns, in the following order.

Optionally, one or more of the following:

- SolarWinds
- .CortexPlugin

- .Orion

Optionally, followed by one of the following:

- .Wireless
- .UI
- .Widgets
- .NPM
- .Apollo
- .CloudMonitoring

If no previous string has been selected, one of the following or optionally, one of the following if a previous string has been selected:

- .Nodes
- .Volumes
- .Interfaces
- .Components

Optionally, followed by:

> -<number>.<number>.<error or success code of previous request>

Followed by:

> .xml

Also, if the first character in the path is a dot (.), the dot is removed. For example, the following are example URLs that could be used:

- https[:]//infinitysoftwares.com/swip/upd/SolarWinds.CortexPlugin.Nodes-5.2.1.xml
- https[:]//infinitysoftwares.com/swip/upd/Orion.Wireless.UI-3.1.0.xml
- https[:]//infinitysoftwares.com/swip/upd/Nodes-1.2.0.xml

The final digit before the .xml extension is not a random character, but represents the return code for previous command requests – typically 0 means success and 1 or other values mean error. For the first 16 requests, a different randomly generated path is used each time, but after that it may repeat random generations that it used previously. If data is being sent to the attacker as a result of a command (e.g. sending system information), the path construction will be different and will be covered in a future blog.

Sunburst also adds the custom header “If-None-Match:” set to the previously described 8-byte userid XOR’d by a randomly generated 8-byte sequence that is appended and then converted into lowercase ASCII hex.

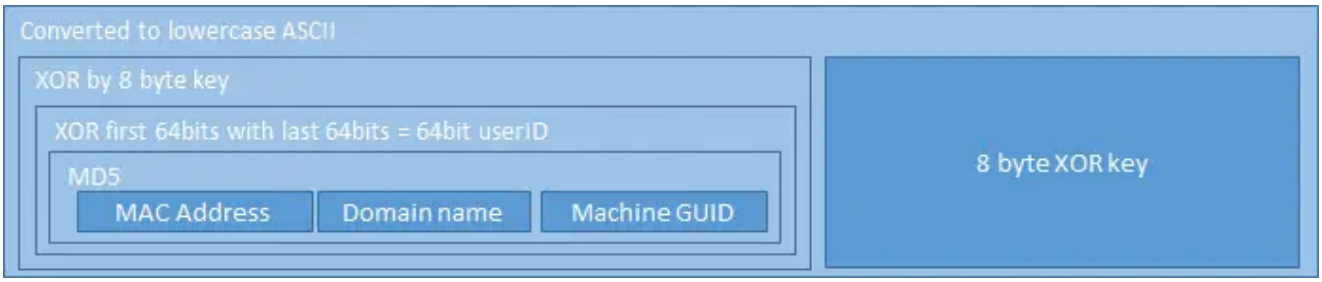

Figure 3. Sunburst custom header

After the HTTP GET request is made, Sunburst checks for a valid response from the attacker. The response uses steganography and is a faux, but convincing XML page of data, where only certain fields are utilized by Sunburst.

The faux XML data is searched for a sequence of:

- 36 hexadecimal characters including dashes
- 32 hexadecimal characters
- 16 hexadecimal characters

These sequences are concatenated together with non-hexadecimal characters removed. The first DWORD (4 bytes) of the sequence represents the size of the remaining data, which is validated and then the next byte is an XOR key. The rest of the data is then XOR decrypted by the key byte and then decompressed. After being decompressed, the first character is a number that specifies the action to perform followed by the number of included arguments in the following data for the command:

```
<?xml version="1.0" encoding="utf-8"?>
<assembly Name="Orion.Wireless.UI" Key="{8f53440a-5ff0-4532-a77d-0004148b5ec7}"
Version="3.1">
     <dependencies>
          <assemblyIdentity Name="Microsoft.Application.Insights"
Keys="{62c5ca68-51f5-4195-88da-6d759326f1dd}" Version="1.1.3.1"
Culture="neutral" PublicKeyToken="b77a5c561934e089"
Hash="cc7b13ffcd2ddd51e53429feb434a9b5"/>
          <assemblyIdentity Name="SolarWinds.Wireless.Heatmaps.Collector"
Keys="{c87e2be4-7b61-408c-9121-ef28f5710ba2}" Version="2.5.0.212"
Culture="neutral" PublicKeyToken="cd35c564ae573ef3"
Hash="cee3422fbb63aa97dec1b430f3d2ddd5"/>
          <assemblyIdentity Name="SolarWinds.Cortex.Node" Keys="{64f861df-2c02
-4f44-ab69-6adf4395db5e}" Version="1.8.1.322" Culture="neutral"
PublicKeyToken="4be67b7d35f5aeb3" Hash="134b043f2bdb534ab78e7de43721fbb5"/>
```

Figure 4. A contrived example of data received by Sunburst that is decoded to extract the command

The command number can instruct Sunburst to perform the following behaviors:

Command number	Command

0	Do nothing
1	Stop HTTP(S) command and control
2	Set sleep time before next check in
3	Gather and send system information: Domain SID of administrator account Hostname Username Operating system version Path of system directory Days elapsed since the system started Information on network adapters, including: Description MACAddress DHCPEnabled DHCPServer DNSHostName DNSDomainSuffixSearchOrder DNSServerSearchOrder IPAddress IPSubnet DefaultIPGateway
4	Proxies HTTP(S) communications
5	Executes a new process/command
6	Sends the currently running process list including PID, process name, and optionally parent PID, username, and domain of process owner
7	Terminates a currently running process
8	Searches for a provided filename pattern and returns the full file paths for the found

8	Searches for a provided filename pattern and returns the full file paths for the found filename pattern
9	Downloads or appends to a file on the system
10	Checks if a file exists
11	Deletes a file
12	Generates the MD5 of a file and if provided an MD5, checks if it matches or returns the value of the MD5 of the file
13	Sends the specified value of the registry
14	Sets a registry key or value
15	Deletes a registry key or value
16	Sends registry subkeys and values
17	Initiates a reboot with the reason code set to a planned application installation (0x800400002)

Figure 5. Command numbers and corresponding behaviors

If no data needs to be sent to the attacker, the above GET request with the return code encoded in the URL filename is sent.

If data needs to be sent to the attacker as a result of these commands (e.g. sending system information), the data will be included in the next regular HTTP(S) communication as a POST request or a HEAD request if some error occurred where the data is missing. Details on each of these behaviors and how data is uploaded to the attacker will be covered in an upcoming blog.

Protection/Mitigation

Tools associated with these attacks will be detected and blocked on machines running Symantec Endpoint products.

File-based protection:

- Backdoor.Sunburst
- Backdoor.Sunburst!gen1
- Backdoor.SuperNova
- Backdoor.Teardrop

Network-based protection:

System Infected: Sunburst Malware Activity

For the latest protection updates, please visit the Symantec Protection Bulletin.

About the Author

Threat Hunter Team

Symantec

The Threat Hunter Team is a group of security experts within Symantec whose mission is to investigate targeted attacks, drive enhanced protection in Symantec products, and offer analysis that helps customers respond to attacks.

Want to comment on this post?