Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims With SUNBURST Backdoor

fireeye.com/blog/threat-research/2020/12/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor.html

Threat Research

December 13, 2020 | by FireEye
FireEye Evasion supply chain

Executive Summary

- We have discovered a global intrusion campaign. We are tracking the actors behind this campaign as UNC2452.
- FireEye discovered a supply chain attack trojanizing SolarWinds Orion business software updates in order to distribute malware we call SUNBURST.
- The attacker's post compromise activity leverages multiple techniques to evade detection and obscure their activity, but these efforts also offer some opportunities for detection.
- The campaign is widespread, affecting public and private organizations around the world.
- FireEye is releasing signatures to detect this threat actor and supply chain attack in the wild. These are found on our public GitHub page. FireEye products and services can help customers detect and block this attack.

Summary

FireEye has uncovered a widespread campaign, that we are tracking as UNC2452. The actors behind this campaign gained access to numerous public and private organizations around the world. They gained access to victims via trojanized updates to SolarWind's Orion IT monitoring and management software. This campaign may have begun as early as Spring 2020 and is currently ongoing. Post compromise activity following this supply chain compromise has included lateral movement and data theft. The campaign is the work of a highly skilled actor and the operation was conducted with significant operational security.

SUNBURST Backdoor

SolarWinds.Orion.Core.BusinessLayer.dll is a SolarWinds digitally-signed component of the Orion software framework that contains a backdoor that communicates via HTTP to third party servers. We are tracking the trojanized version of this SolarWinds Orion plug-in as SUNBURST.

After an initial dormant period of up to two weeks, it retrieves and executes commands, called "Jobs", that include the ability to transfer files, execute files, profile the system, reboot the machine, and disable system services. The malware masquerades its network traffic as the Orion Improvement Program (OIP) protocol and stores reconnaissance results within legitimate plugin configuration files allowing it to blend in with legitimate SolarWinds activity. The backdoor uses multiple obfuscated blocklists to identify forensic and anti-virus tools running as processes, services, and drivers.

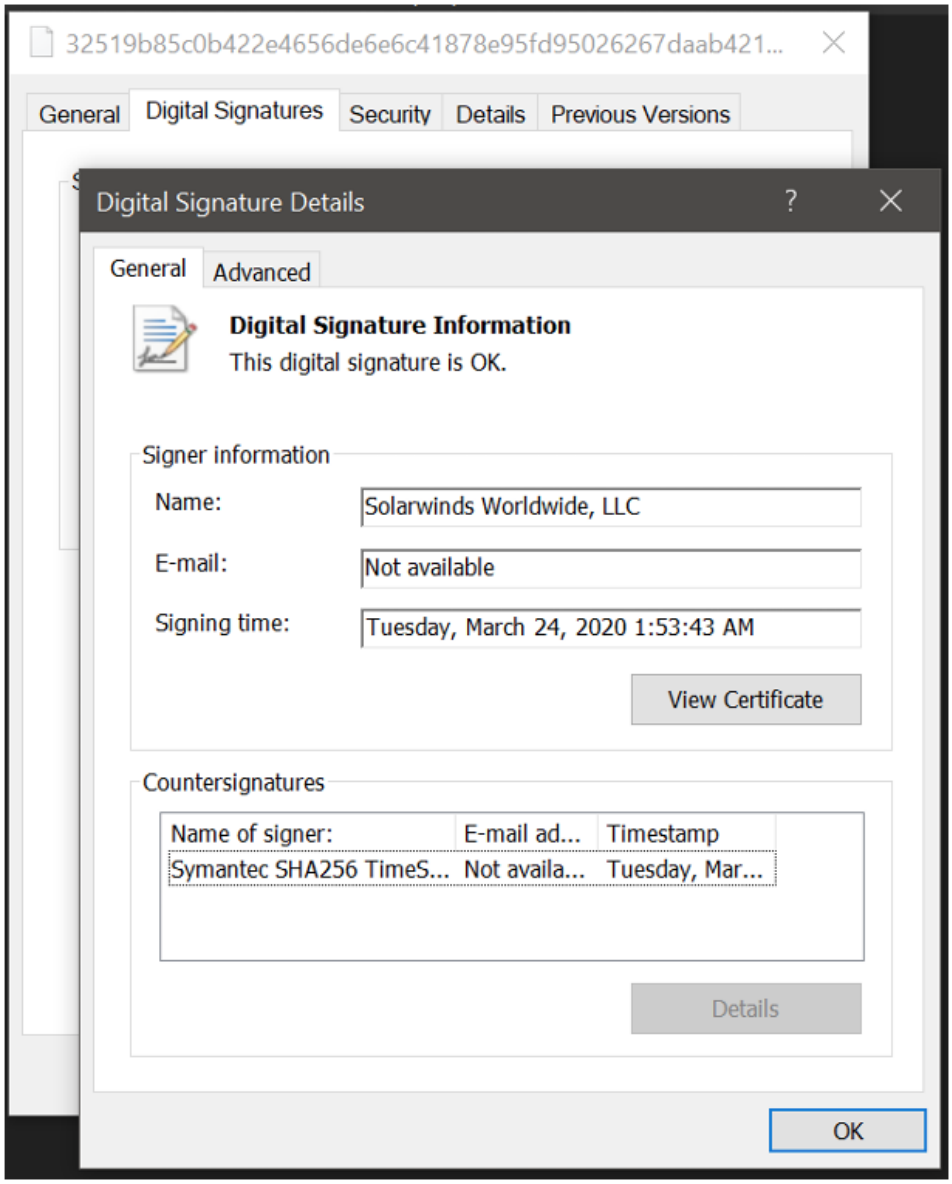

Figure 1: SolarWinds digital signature on software with backdoor

Multiple trojanzied updates were digitally signed from March - May 2020 and posted to the SolarWinds updates website, including:

> hxxps://downloads.solarwinds[.]com/solarwinds/CatalogResources/Core/2019.4/2019.4.5220.20574/SolarWinds-Core-v2019.4.5220-Hotfix5.msp

The trojanized update file is a standard Windows Installer Patch file that includes compressed resources associated with the update, including the trojanized SolarWinds.Orion.Core.BusinessLayer.dll component. Once the update is installed, the malicious DLL will be loaded by the legitimate SolarWinds.BusinessLayerHost.exe or SolarWinds.BusinessLayerHostx64.exe (depending on system configuration). After a dormant period of up to two weeks, the malware will attempt to resolve a subdomain of avsvmcloud[.]com. The DNS response will return a CNAME record that points to a Command and Control (C2) domain. The C2 traffic to the malicious domains is designed to mimic normal SolarWinds API communications. The list of known malicious infrastructure is available on FireEye's GitHub page.

Worldwide Victims Across Multiple Verticals

FireEye has detected this activity at multiple entities worldwide. The victims have included government, consulting, technology, telecom and extractive entities in North America, Europe, Asia and the Middle East. We anticipate there are additional victims in other countries and verticals. FireEye has notified all entities we are aware of being affected.

Post Compromise Activity and Detection Opportunities

We are currently tracking the software supply chain compromise and related post intrusion activity as UNC2452. After gaining initial access, this group uses a variety of techniques to disguise their operations while they move laterally. This actor prefers to maintain a light malware footprint, instead preferring legitimate credentials and remote access for access into a victim's environment. This section will detail a few of the notable techniques and outline potential opportunities for detection.

TEARDROP and BEACON Malware Used

Multiple SUNBURST samples have been recovered, delivering different payloads. In at least one instance the attackers deployed a previously unseen memory-only dropper we've dubbed TEARDROP to deploy Cobalt Strike BEACON.

TEARDROP is a memory only dropper that runs as a service, spawns a thread and reads from the file "gracious_truth.jpg", which likely has a fake JPG header. Next it checks that HKU\SOFTWARE\Microsoft\CTF exists, decodes an embedded payload using a custom rolling XOR algorithm and manually loads into memory an embedded payload using a custom PE-like file format. TEARDROP does not have code overlap with any previously seen malware. We believe that this was used to execute a customized Cobalt Strike BEACON.

Mitigation: FireEye has provided two Yara rules to detect TEARDROP available on our GitHub. Defenders should look for the following alerts from FireEye HX: MalwareGuard and WindowsDefender:

Process Information

file_operation_closed
file-path*: "c:\\windows\\syswow64\\netsetupsvc.dll
actor-process:
pid: 17900

Window's defender Exploit Guard log entries: (Microsoft-Windows-Security-Mitigations/KernelMode event ID 12)

Process"\Device\HarddiskVolume2\Windows\System32\svchost.exe" (PID XXXXX) would have been blocked from loading the non-Microsoft-signed binary
'\Windows\SysWOW64\NetSetupSvc.dll'

Attacker Hostnames Match Victim Environment

The actor sets the hostnames on their command and control infrastructure to match a legitimate hostname found within the victim's environment. This allows the adversary to blend into the environment, avoid suspicion, and evade detection.

Detection Opportunity

The attacker infrastructure leaks its configured hostname in RDP SSL certificates, which is identifiable in internet-wide scan data. This presents a detection opportunity for defenders -- querying internet-wide scan data sources for an organization's hostnames can uncover malicious IP addresses that may be masquerading as the organization. (Note: IP Scan history often shows IPs switching between default (WIN-*) hostnames and victim's hostnames) Cross-referencing the list of IPs identified in internet scan data with remote access logs may identify evidence of this actor in an environment. There is likely to be a single account per IP address.

IP Addresses located in Victim's Country

The attacker's choice of IP addresses was also optimized to evade detection. The attacker primarily used only IP addresses originating from the same country as the victim, leveraging Virtual Private Servers.

Detection Opportunity

This also presents some detection opportunities, as geolocating IP addresses used for remote access may show an impossible rate of travel if a compromised account is being used by the legitimate user and the attacker from disparate IP addresses. The attacker used multiple IP addresses per VPS provider, so once a malicious login from an unusual ASN is identified, looking at all logins from that ASN can help detect additional malicious activity. This can be done alongside baselining and normalization of ASN's used for legitimate remote access to help identify suspicious activity.

Lateral Movement Using Different Credentials

Once the attacker gained access to the network with compromised credentials, they moved laterally using multiple different credentials. The credentials used for lateral movement were always different from those used for remote access.

Detection Opportunity

Organizations can use HX's LogonTracker module to graph all logon activity and analyze systems displaying a one-to-many relationship between source systems and accounts. This will uncover any single system authenticating to multiple systems with multiple accounts, a relatively uncommon occurrence during normal business operations.

Temporary File Replacement and Temporary Task Modification

The attacker used a temporary file replacement technique to remotely execute utilities: they replaced a legitimate utility with theirs, executed their payload, and then restored the legitimate original file. They similarly manipulated scheduled tasks by updating an existing legitimate task to execute their tools and then returning the scheduled task to its original configuration. They routinely removed their tools, including removing backdoors once legitimate remote access was achieved.

Detection Opportunity

Defenders can examine logs for SMB sessions that show access to legitimate directories and follow a delete-create-execute-delete-create pattern in a short amount of time. Additionally, defenders can monitor existing scheduled tasks for temporary updates, using frequency analysis to identify anomalous modification of tasks. Tasks can also be monitored to watch for legitimate Windows tasks executing new or unknown binaries.

This campaign's post compromise activity was conducted with a high regard for operational security, in many cases leveraging dedicated infrastructure per intrusion. This is some of the best operational security that FireEye has observed in a cyber attack, focusing on evasion and leveraging inherent trust. However, it *can* be detected through persistent defense.

In-Depth Malware Analysis

SolarWinds.Orion.Core.BusinessLayer.dll (b91ce2fa41029f6955bff20079468448) is a SolarWinds-signed plugin component of the Orion software framework that contains an obfuscated backdoor which communicates via HTTP to third party servers. After an initial dormant period of up to two weeks, it retrieves and executes commands, called "Jobs", that include the ability to transfer and execute files, profile the system, and disable system services. The backdoor's behavior and network protocol blend in with legitimate SolarWinds activity, such as by masquerading as the Orion Improvement Program (OIP) protocol and storing reconnaissance results within plugin configuration files. The backdoor uses multiple blocklists to identify forensic and anti-virus tools via processes, services, and drivers.

Unique Capabilities

- Subdomain DomainName Generation Algorithm (DGA) is performed to vary DNS requests
  - CNAME responses point to the C2 domain for the malware to connect to.
  - The IP block of A record responses controls malware behavior
- Command and control traffic masquerades as the legitimate Orion Improvement Program
- Code hides in plain site by using fake variable names and tying into legitimate components

Delivery and Installation

Authorized system administrators fetch and install updates to SolarWinds Orion via packages distributed by SolarWinds's website. The update package CORE-2019.4.5220.20574-SolarWinds-Core-v2019.4.5220-Hotfix5.msp (02af7cec58b9a5da1c542b5a32151ba1) contains the SolarWinds.Orion.Core.BusinessLayer.dll described in this report. After installation, the Orion software framework executes the .NET program SolarWinds.BusinessLayerHost.exe to load plugins, including SolarWinds.Orion.Core.BusinessLayer.dll. This plugin contains many legitimate namespaces, classes, and routines that implement functionality within the Orion framework. Hidden in plain sight, the class SolarWinds.Orion.Core.BusinessLayer.OrionImprovementBusinessLayer implements an HTTP-based backdoor. Code within the logically unrelated routine SolarWinds.Orion.Core.BusinessLayer.BackgroundInventory.InventoryManager.RefreshInternal invokes the backdoor code when the Inventory Manager plugin is loaded.

SolarWinds.Orion.Core.BusinessLayer.dll is signed by SolarWinds, using the certificate with serial number 0f:e9:73:75:20:22:a6:06:ad:f2:a3:6e:34:5d:c0:ed. The file was signed on March 24, 2020.

Initialization

On execution of the malicious SolarWinds.Orion.Core.BusinessLayer.OrionImprovementBusinessLayer.Initialize method the sample verifies that its lower case process name hashes to the value 17291806236368054941. This hash value is calculated as the standard FNV-1A 64-bit hash with an additional XOR by 6605813339339102567 after computing the FNV-1A. This hash matches a process named businesslayerhost.exe.

The sample only executes if the filesystem write time of the assembly is at least 12 to 14 days prior to the current time; the exact threshold is selected randomly from an interval. The sample continues to check this time threshold as it is run by a legitimate recurring background task. Once the threshold is met, the sample creates the named pipe 583da945-62af-10e8-4902-a8f205c72b2e to act as a guard that only one instance is running before reading SolarWinds.Orion.Core.BusinessLayer.dll.config from disk and retrieving the XML field appSettings. The appSettings fields' keys are legitimate values that the malicious logic re-purposes as a persistent configuration. The key ReportWatcherRetry must be any value other than 3 for the sample to continue execution.

The sample checks that the machine is domain joined and retrieves the domain name before execution continues. A userID is generated by computing the MD5 of all network interface MAC addresses that are up and not loopback devices, the domain name, and the registry value HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\MachineGuid. The userID is encoded via a custom XOR scheme after the MD5 is calculated. The ReportWatcherPostpone key of appSettings is then read from SolarWinds.Orion.Core.BusinessLayer.dll.config to retrieve the initial, legitimate value. This operation is performed as the sample later bit packs flags into this field and the initial value must be known in order to read out the bit flags. The sample then invokes the method Update which is the core event loop of the sample.

DGA and Blocklists

The backdoor determines its C2 server using a Domain Generation Algorithm (DGA) to construct and resolve a subdomain of avsvmcloud[.]com. The Update method is responsible for initializing cryptographic helpers for the generation of these random C2 subdomains. These subdomains are concatenated with one of the following to create the hostname to resolve:

- .appsync-api.eu-west-1[.]avsvmcloud[.]com
- .appsync-api.us-west-2[.]avsvmcloud[.]com
- .appsync-api.us-east-1[.]avsvmcloud[.]com
- .appsync-api.us-east-2[.]avsvmcloud[.]com

Process name, service name, and driver path listings are obtained, and each value is hashed via the FNV-1a + XOR algorithm as described previously and checked against hardcoded blocklists. Some of these hashes have been brute force reversed as part of this analysis, showing that these routines are scanning for analysis tools and antivirus engine components. If a blocklisted process is found the Update routine exits and the sample will continue to try executing the routine until the blocklist passes. Blocklisted services are stopped by setting their HKLM\SYSTEM\CurrentControlSet\services\<service_name>\Start registry entries to value 4 for disabled. Some entries in the service list if found on the system may affect the DGA algorithms behavior in terms of the values generated. The list of stopped services is then bit-packed into the ReportWatcherPostpone key of the appSettings entry for the samples' config file. If any service was transitioned to disabled the Update method exits and retries later. The sample retrieves a driver listing via the WMI query Select * From Win32_SystemDriver. If any blocklisted driver is seen the Update method exits and retries. If all blocklist tests pass, the sample tries to resolve api.solarwinds.com to test the network for connectivity.

Network Command and Control (C2)

If all blocklist and connectivity checks pass, the sample starts generating domains in a while loop via its DGA. The sample will delay for random intervals between the generation of domains; this interval may be any random value from the ranges 1 to 3 minutes, 30 to 120 minutes, or on error conditions up to 420 to 540 minutes (9 hours). The DNS A record of generated domains is checked against a hardcoded list of IP address blocks which control the malware's behavior. Records within the following ranges will terminate the malware and update the configuration key ReportWatcherRetry to a value that prevents further execution:

- 10.0.0.0/8
- 172.16.0.0/12
- 192.168.0.0/16
- 224.0.0.0/3

- fc00:: - fe00::
- fec0:: - ffc0::
- ff00:: - ff00::
- 20.140.0.0/15
- 96.31.172.0/24
- 131.228.12.0/22
- 144.86.226.0/24

Once a domain has been successfully retrieved in a CNAME DNS response the sample will spawn a new thread of execution invoking the method HttpHelper.Initialize which is responsible for all C2 communications and dispatching. The HTTP thread begins by delaying for a configurable amount of time that is controlled by the SetTime command. The HTTP thread will delay for a minimum of 1 minute between callouts. The malware uses HTTP GET or HEAD requests when data is requested and HTTP PUT or HTTP POST requests when C2 output data is being sent to the server. The PUT method is used when the payload is smaller than 10000 bytes; otherwise the POST method is used. The If-None-Match HTTP header holds an XOR encoded representation of the userID calculated earlier, with a random array of bytes appended that is of the same length.

A JSON payload is present for all HTTP POST and PUT requests and contains the keys “userId”, “sessionId”, and “steps”. The “steps” field contains a list of objects with the following keys: “Timestamp”, “Index”, “EventType”, “EventName”, “DurationMs”, “Succeeded”, and “Message”. The JSON key “EventType” is hardcoded to the value “Orion”, and the “EventName” is hardcoded to “EventManager”. Malware response messages to send to the server are DEFLATE compressed and single-byte-XOR encoded, then split among the “Message” fields in the “steps” array. Each “Message” value is Base64 encoded separately. Not all objects in the “steps” array contribute to the malware message – the integer in the “Timestamp” field must have the 0x2 bit set to indicate that the contents of the “Message” field are used in the malware message. Step objects whose bit 0x2 is clear in the Timestamp field contain random data and are discarded when assembling the malware response.

Steganography

In observed traffic these HTTP response bodies attempt to appear like benign XML related to .NET assemblies, but command data is actually spread across the many GUID and HEX strings present. Commands are extracted from HTTP response bodies by searching for HEX strings using the following regular expression: "\{[0-9a-f-]{36}\}"|"[0-9a-f]{32}"|"[0-9a-f]{16}". Command data is spread across multiple strings that are disguised as GUID and HEX strings. All matched substrings in the response are filtered for non HEX characters, joined together, and HEX-decoded. The first DWORD value shows the actual size of the message, followed immediately with the message, with optional additional junk bytes following. The extracted message is single-byte XOR decoded using the first byte of the message, and this is then DEFLATE decompressed. The first character is an ASCII integer that maps to the JobEngine enum, with optional additional command arguments delimited by space characters.

Commands are then dispatched to a JobExecutionEngine based upon the command value as described next.

Supported Commands

Command	Value	Operation
Idle	0	No operation
Exit	1	Terminate the current thread.
SetTime	2	Sets the delay time between main event loop executions Delay is in seconds, and varies random between [.9 * <delay>, 1.1 * <delay>]. If the delay is < 300 it is doubled on the next execution through the loop, this means it should settle onto an interval of around [5, 10] minutes. There is a second, unrelated delay routine that delays for a random interval between [16hrs, 83hrs]
CollectSystemDescription	3	Profile the local system including hostname, username, OS version, MAC addresses, IP address, DHCP configuration, and domain information.

UploadSystemDescription	4	Perform a HTTP request to the specified URL, parse the results and compare components against unknown hashed values. Format a report and send to the C2 server.
RunTask	5	Starts a new process with the given file path and arguments
GetProcessByDescription	6	Returns a process listing. If no arguments are provided returns just the PID and process name. If an argument is provided it also returns the parent PID and username and domain for the process owner.
KillTask	7	Terminate the given process, by PID.
GetFileSystemEntries	8	Given a path and an optional match pattern recursively list files and directories
WriteFile	9	Given a file path and a Base64 encoded string write the contents of the Base64 decoded string to the given file path. Write using append mode. Delay for [1s, 2s] after writing is done.
FileExists	10	Tests whether the given file path exists.
DeleteFile	11	Deletes the specified file path.
GetFileHash	12	Compute the MD5 of a file at a given path and return result as a HEX string. If an argument is provided, it is the expected MD5 hash of the file and returns an error if the calculated MD5 differs.
ReadRegistryValue	13	Arbitrary registry read from one of the supported hives
SetRegistryValue	14	Arbitrary registry write from one of the supported hives.
DeleteRegistryValue	15	Arbitrary registry delete from one of the supported hives
GetRegistrySubKeyAndValueNames	16	Returns listing of subkeys and value names beneath the given registry path
Reboot	17	Attempts to immediately trigger a system reboot.

Indicators and Detections to Help the Community

To empower the community to detect this supply chain backdoor, we are publishing indicators and detections to help organizations identify this backdoor and this threat actor. The signatures are a mix of Yara, IOC, and Snort formats.

A list of the detections and signatures are available on the FireEye GitHub repository found here. We are releasing detections and will continue to update the public repository with overlapping detections for host and network-based indicators as we develop new or refine existing ones. We have found multiple hashes with this backdoor and we will post updates of those hashes.

MITRE ATT&CK Techniques Observed

ID	Description

T1012	Query Registry
T1027	Obfuscated Files or Information
T1057	Process Discovery
T1070.004	File Deletion
T1071.001	Web Protocols
T1071.004	Application Layer Protocol: DNS
T1083	File and Directory Discovery
T1105	Ingress Tool Transfer
T1132.001	Standard Encoding
T1195.002	Compromise Software Supply Chain
T1518	Software Discovery
T1518.001	Security Software Discovery
T1543.003	Windows Service
T1553.002	Code Signing
T1568.002	Domain Generation Algorithms
T1569.002	Service Execution
T1584	Compromise Infrastructure

Immediate Mitigation Recommendations

SolarWinds recommends all customers immediately upgrade to Orion Platform release 2020.2.1 HF 1, which is currently available via the SolarWinds Customer Portal. In addition, SolarWinds has released additional mitigation and hardening instructions here.

In the event you are unable to follow SolarWinds' recommendations, the following are immediate mitigation techniques that could be deployed as first steps to address the risk of trojanized SolarWinds software in an environment. If attacker activity is discovered in an environment, we recommend conducting a comprehensive investigation and designing and executing a remediation strategy driven by the investigative findings and details of the impacted environment.

- Ensure that SolarWinds servers are isolated / contained until a further review and investigation is conducted. This should include blocking all Internet egress from SolarWinds servers.
- If SolarWinds infrastructure is not isolated, consider taking the following steps:
    - Restrict scope of connectivity to endpoints from SolarWinds servers, especially those that would be considered Tier 0 / crown jewel assets
    - Restrict the scope of accounts that have local administrator privileged on SolarWinds servers.
    - Block Internet egress from servers or other endpoints with SolarWinds software.

- Consider (at a minimum) changing passwords for accounts that have access to SolarWinds servers / infrastructure. Based upon further review / investigation, additional remediation measures may be required.
- If SolarWinds is used to managed networking infrastructure, consider conducting a review of network device configurations for unexpected / unauthorized modifications. Note, this is a proactive measure due to the scope of SolarWinds functionality, not based on investigative findings.

Acknowledgements

This blog post was the combined effort of numerous personnel and teams across FireEye coming together. Special thanks to:

Andrew Archer, Doug Bienstock, Chris DiGiamo, Glenn Edwards, Nick Hornick, Alex Pennino, Andrew Rector, Scott Runnels, Eric Scales, Nalani Fraser, Sarah Jones, John Hultquist, Ben Read, Jon Leathery, Fred House, Dileep Jallepalli, Michael Sikorski, Stephen Eckels, William Ballenthin, Jay Smith, Alex Berry, Nick Richard, Isif Ibrahima, Dan Perez, Marcin Siedlarz, Ben Withnell, Barry Vengerik, Nicole Oppenheim, Ian Ahl, Andrew Thompson, Matt Dunwoody, Evan Reese, Steve Miller, Alyssa Rahman, John Gorman, Lennard Galang, Steve Stone, Nick Bennett, Matthew McWhirt, Mike Burns, Omer Baig.

Also special thanks to Nick Carr, Christopher Glyer, and Ramin Nafisi from Microsoft.