Editor: Sean Peisert, sppeisert@lbl.gov

Perspectives on the SolarWinds Incident

Sean Peisert | Lawrence Berkeley National Laboratory and University of California, Davis
Bruce Schneier | Harvard University
Hamed Okhravi | MIT Lincoln Laboratory
Fabio Massacci | University of Trento and Vrije Universiteit Amsterdam
Terry Benzel | USC Information Sciences Institute
Carl Landwehr | University of Michigan
Mohammad Mannan | Concordia University
Jelena Mirkovic | University of Southern California Information Sciences Institute
Atul Prakash | University of Michigan
James Bret Michael | Naval Postgraduate School

Editor's Note

A significant cybersecurity event has recently been discovered in which malicious actors gained access to the source code for the Orion monitoring and management software made by the company SolarWinds and inserted malware into that source code. This article contains brief perspectives from a few members of the *IEEE Security & Privacy* editorial board regarding that incident.

A serious cybersecurity event was recently revealed: malicious actors had gained access to the source code for the SolarWinds Orion monitoring and management software. They inserted malware into that source code so that, when the software was distributed to and deployed by SolarWinds customers as part of an update, the malicious software could be used to surveil customers who unknowingly installed the malware and gain potentially arbitrary control over the systems managed by Orion. Of course, such a level of control has given attackers opportunities for further exploitation as well.

At the time of this writing, it is reported by SolarWinds that the update containing the malware was installed by thousands of customers, including numerous U.S. federal agencies and businesses around the world. The cybersecurity company FireEye was among the first reported to be actually compromised by the malware.

Numerous technical details on this incident appear online and in a variety of other publications as well as from SolarWinds itself.[1] At the same time, there remains a great deal to say about how we might best respond to and recover from the incident as well as how we might avoid similar such events in the future.

The news of this event broke at a time that made it infeasible for *IEEE Security & Privacy* to write a full, detailed piece by the production deadline for this issue. However, due to the magnitude of the

Digital Object Identifier 10.1109/MSEC.2021.3051235
Date of current version: 15 March 2021

incident, the Editorial Board still wanted to address it in some way without waiting two months for the next issue. Therefore, this issue contains two pieces related to the SolarWinds incident: This first article contains brief perspectives from some members of the *IEEE Security & Privacy* Editorial Board, including numerous questions asked by Editorial Board members and also some suggested solutions. The second is a companion "Point–Counterpoint" column article by Fabio Massacci and Trent Jaeger that digs specifically into the quandaries and questions of software patching that relate to the SolarWinds incident. Additional details will undoubtedly continue to surface, and *IEEE Security & Privacy* expects to cover this occurrence further and in greater detail in future issues as we continue to learn about this compromise and its effects.

— Sean Peisert

Editorial Board Members' Perspectives

SolarWinds and Market Incentives

Bruce Schneier

The penetration of government and corporate networks worldwide is the result of inadequate cyberdefenses across the board. The lessons are many, but I want to focus on one important one we've learned: the software that's managing our critical networks isn't secure, and that's because the market doesn't reward that security.

SolarWinds is a perfect example. The company was the initial infection vector for much of the operation. Its trusted position inside so many critical networks made it a perfect target for a supply-chain attack, and its shoddy security practices made it an easy target.

Why did SolarWinds have such bad security? The answer is because it was more profitable. The company is owned by Thoma Bravo partners, a private-equity firm known for radical cost-cutting in the name of short-term profit. Under CEO Kevin Thompson, the company underspent on security even as it outsourced software development. *The New York Times* reports that the company's cybersecurity advisor quit after his "basic recommendations were ignored." In a very real sense, SolarWinds profited because it secretly shifted a whole bunch of risk to its customers: the U.S. government, IT companies, and others.

This problem isn't new, and, while it's exacerbated by the private-equity funding model, it's not unique to it. In general, the market doesn't reward safety and security—especially when the effects of ignoring those things are long term and diffuse. The market rewards short-term profits at the expense of safety and security. (Watch and see whether SolarWinds suffers any long-term effects from this hack, or whether Thoma Bravo's bet that it could profit by selling an insecure product was a good one.)

The solution here is twofold. The first is to improve government software procurement. Software is now critical to national security. Any system of procuring that software needs to evaluate the security of the software and the security practices of the company, in detail, to ensure that they are sufficient to meet the security needs of the network they're being installed in. If these evaluations are made public, along with the list of companies that meet them, all network buyers can benefit from them. It's a win for everybody.

But that isn't enough; we need a second part. The only way to force companies to provide safety and security features for customers is through regulation. This is true whether we want seatbelts in our cars, basic food safety at our restaurants, pajamas that don't catch on fire, or home routers that aren't vulnerable to cyberattack. The government needs to set minimum security standards for software that's used in critical network applications, just as it sets software standards for avionics.

Without these two measures, it's just too easy for companies to act like SolarWinds: save money by skimping on safety and security and hope for the best in the long term. That's the rational thing for companies to do in an unregulated market, and the only way to change that is to change the economic incentives.

Hamed Okhravi

The SolarWinds hacks are not novel or unique. Software Trojans and supply-chain attacks have been understood in the community for many decades; as a case in point, even the logo of the IEEE Symposium on Security and Privacy, one of the top-tier venues in computer security, is a Trojan horse! However, the SolarWinds hack highlights some of the troubling trends and important lessons that we, as a community, should pay attention to and work on resolving with better technologies as well as policies.

First, for decades, one of the main paradigms in security has been "additive" security. We "add" tools, antiviruses, intrusion detection/prevention systems, monitoring and management tools, and so on to make a system more secure. Every new tool, while it might reduce parts of the attack surface, also adds a new attack surface: vulnerabilities in the tool itself become a new possible vector of compromise for the system. Numerous recent vulnerabilities discovered in security software further highlight this tradeoff.[2]

This is not an easy problem to resolve. While some security software programs close many more holes than they open, others, by the virtue of their sheer size and complexity, can make the system more vulnerable. We, as a community, do not have a proper, quantitative understanding of this tradeoff, and, thus, decisions for deploying or not deploying a new tool are made without proper understanding of their implications. This also highlights the need for another paradigm that requires more attention in the community: "reductive" security, or making a system more secure by removing its unnecessary services, libraries, features, and so on, thereby shrinking its attack surface.

Second, software supply-chain attacks are as important—if not more important—than hardware supply-chain attacks. Because of the vast complexity of modern software and the comparatively low cost of introducing malicious logic into the code, this threat vector can be expected to become more prominent over the next few years. The traditional security mechanisms that are developed to ward off malicious "outside" code, such as code signing, digital certificates, secure update delivery mechanisms, transport security, and so on, have little impact on this threat vector, as evidenced by the SolarWinds hack. Deeper analysis techniques, code provenance tracking, and liability mechanisms as well as novel technologies and refined policies are necessary to mitigate such attacks.

Finally, cobbling together software code and libraries from various and sometimes unknown sources may be a good way of developing tools quickly and cheaply; it is certainly not an effective way to develop tools securely. Large software projects often incorporate code and libraries from many sources[3] with unknown provenance. At best, these code pieces can be buggy or faulty, and, at worst, they can contain malicious, implanted logic. Rapid and secure software development remains as tantalizing a goal as ever.

Fabio Massacci

SolarWinds offers network and database monitoring and logging services. SolarWinds, like all modern IT suppliers, grew out of mergers and acquisitions. A simple investigation on the company's FAQ page[1] shows that the "vulnerable platform" is, in reality, a patchwork of services: 18 (sub)products are, indeed, affected by the vulnerabilities (from Access Management, to Web Performance Monitor, to Log Analyzer—which was recently acquired); another 40+ (sub)products are not affected.

So, for example, Log Analyzer is affected, while Kiwi Log Analyzer, Log & Event Manager, Log and Event Manager Workstation Edition, and Loggly are not. The Database Performance Analyzer Integration Module is affected, but the Database Performance Analyzer is not—or, more precisely, "we do not believe is affected."[1]

You have no chance as an end administrator to know what you are getting onto your systems. Figure 1 illustrates the stark reality for modern administrators.

Of course, as all system-monitoring tools, SolarWinds systems are properly integrated with various mechanisms of authentication (such as Microsoft Active Directories), possibly with single sign-on (SSO) (which is a good security practice, isn't it? But I will return to that) at the final user sites. If one of these system tools from the patchwork is compromised, the end user administrators are likely compromised. This is indeed quite clear from the instructions in the Emergency Act 21-01.[4]

What lessons do we learn from this story?

- This is not about third-party software; it is about fourth- or eighth- or 16th-party software. We don't know what we are installing, and even the people who sell it to us have no clear idea.
- All security procedures are designed for first-party software. (Just think about threat analysis and STRIDE, or quality gates, and so on.) We can claim that suppliers should stick to security procedures and so forth, but are we willing to pay the price? If nobody is liable for anything. . . .
- Several security mechanisms (like SSO) that we conceived to be used with first-party products (which are secured as first-party products) can be damning in the new world order, as you SSO into something you have no idea about.

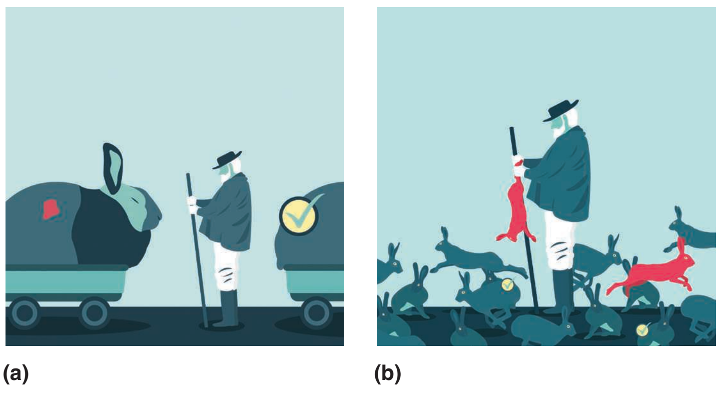

Figure 1. (a) How we think the world is with good security gates and processes. (b) How the world really is. In the framework of a week, SolarWinds fetched a vulnerability (SUNBURST) to find out that another one was there (SUPERNOVA). (Source: Anna Formilan for the European Union AssureMOSS project. Used with permission.)

Terry Benzel

There has been a lot of important technical discussion to date, but I would also like to see some conversation about what the threat is, what the motivations are for exploiting the vulnerabilities, and how those fit into the larger social–political landscape. Given the extent of access gained, what is it being used for? Is it "simply" exporting data and information? Are the accesses being used to change the information/platforms/integrity of data and processes?

The wide variety of organizations that have been attacked makes it difficult to attribute a single motivation. How do we close all of the doors that have been opened and understand what was done during the long time that access was available?

Finally, we should note that this is not unique and not specific to adversaries. This is just a highly visible move in a grand game that has been ongoing for a very long time. Understandably, digging very deeply in this area will quickly become sensitive, but I do think that we owe the readers a bit of discussion about the bigger picture.

Carl Landwehr

We all accept digitally signed software updates. Solar Winds evidently distributed validly signed updates that included malware, so customers who trusted them and installed the sabotaged updates became vulnerable. Perhaps the first time this kind of attack was written about was in Ken Thompson's Turing Award lecture "Reflections Trusting Trust" in 1984. What can be done about this problem from a technical standpoint? One possibility might be to develop tools that would help consumers of updates assess their security more deeply than merely checking that a digital signature is valid. The signature provides provenance assurance and nothing more, and if the signing key has been compromised, even that is lost.

What if the consumer could run checks on the signed update to check it for vulnerabilities? This is the approach taken by the Securely Taking on New Executable Software of Uncertain Provenance (STONESOUP) program at Intelligence Advanced Research Projects Activity in 2009. Of course, this approach will never ensure there are no vulnerabilities, but it could raise the bar substantially. Considerable technology has been developed by STONESOUP and subsequent programs, including DARPA's Cyber Grand Challenge, that demonstrates the ability to perform significant automated analyses on both source code and binaries to expose potential vulnerabilities (and, in some cases, even to remove them). This technology is commercially available.

Even better would be to put the responsibility on developers, who are in the best position to apply serious effort to ensuring that their updates don't contain malware. Why not establish best practices for the release of updates and make those developers who can't show they followed these practices liable for the malware they distribute? If we want to improve the resilience of our software systems, why don't we start by establishing liability for companies who distribute sabotaged updates? Nation-state attackers may still penetrate defenses, and relief from liability should be available to companies that can show they followed the best practices. Establishing liability requires policy action, but perhaps the severity of the Solar Winds compromise can provide sufficient political will to help the software industry begin to assume responsibility for its products.

Mohammad Mannan

Security products, like other software applications, come with numerous vulnerabilities and design flaws, and are exploited in cyber attacks every now and then. As of January 26, 2021, a quick search at cve.mitre.org on SolarWinds/McAfee/Symantec returns hundreds of Common Vulnerabilities and Exposures (CVE) entries. SolarWinds is making the headline now, but will possibly be forgotten sometime soon (recall the RSA hack from 2011). I would argue that security products should be a distinct software category—perhaps be made to follow some standards. Other software categories provide direct and useful functionalities. In contrast, security products don't enable any new functionality—designed only to secure other systems and applications, and as such, must not create new security nightmares.

Some suggestions along this line seem apropos:

- Security products should do no harm. They should be designed with failure in mind—i.e., they must fail gracefully, without reducing the security level of the protected system compared to the original state (without the product itself). This will require them to work with less privilege. To some extent, these products should be treated like medicine/vaccine—the bottom line should be to avoid any harm.
- The operating system (OS) itself should take over more responsibilities gradually for tasks requiring system-level access. (The OS is the necessary evil that we need to keep secure anyway.)
- Security products need to accept liabilities if they downgrade the target system's security in any way—this can force them to become a security company (i.e., adopt the best security practices, including in development/deployment/management phases) as opposed to yet another software company that happens to make security products Instead of requiring an abuse to happen to

trigger penalties, another avenue could be to incur a certain amount of fine for each new CVE, and after a certain threshold, the product or company may lose their "security" label.

Of course, these suggestions here are incomplete and need more serious consideration. However, if no concrete measures are taken soon, we should not be surprised when (not if) the next SolarWinds like incident happens.

Jelena Mirkovic

It is important to note that this was a well-planned, well-executed attack. For example, the changes in code were done at the source code level over months and possibly by an insider.[5] The changes were subtle, using coding styles and naming conventions that were already in the code. From that point on, the code was tested, signed, and released through the SolarWinds platform. After infection, the code was dormant for two weeks. After that, it mimicked another known protocol to exfiltrate credentials. These credentials were used to access Security Assertion Markup Language token authority and craft new tokens. These tokens were then used to access confidential data.

In my opinion, there were two weak links that the attack exploited: insiders in the companies producing software/hardware and installing monitoring software that precedes encryption (otherwise, credentials could not be stolen). The first is impossible to eliminate. The second—perhaps, but there will always be tension between the desire to monitor everything and protecting privacy (in this case, the privacy of credentials that were later used to compromise the security of data).

I think the better question is how one could detect when such high-level compromises occur. Once stolen, data have to be exfiltrated somehow. In this specific case, new Internet destinations were contacted by compromised machines, and data were exfiltrated. This could potentially be detected, although command and control servers were hosted on public clouds (Microsoft and Amazon), which may not have looked suspicious. Therefore, the third weak link is our reliance on clouds and allowing organizations with high confidentiality requirements to communicate with public clouds.

Atul Prakash

The issues surrounding the SolarWinds incident raise important questions. First and foremost, this is clearly a supply-chain attack. I think that a number of my questions are around potential privileges, both in this incident specifically and in software supply chains more generally. In this incident, were the compromised SolarWinds servers more overprivileged than necessary to function properly? Toward answering this, what capability did SolarWinds provide, and how is it typically deployed? Why did its compromise then lead to the compromise of internal email accounts, even assuming that the SolarWinds server was fully compromised and gave root access? Does the server act as a man in the middle, or is it also installed at end servers, such as mail or authentication servers? If it is a man in the middle, why did end-to-end authentication fail to protect mail accounts? How many other such third-party services that could allow deeper compromise of other services are out there?

James Bret Michael

Damage assessment will likely take months to accomplish, given the reach of SolarWinds (i.e., the number of customers and, through transitivity, other parties affected) and the large number of organizations (e.g., the U.S. Federal Bureau of Investigation, the United States Cyber Command, and security firms) involved in the investigations of SolarWinds.

Cyber hygiene still seems to be a stumbling block to thwarting the successful use of low-hanging exploitable vulnerabilities. Of the 300,000 customers (granted, not all use the Orion software or were still licensed to do so), a relatively small number of customers actually uploaded the updates. Preventing another SolarWinds debacle is a technical (e.g., eliminating unnecessary system complexity to make it feasible to assess the trustworthiness of the software/firmware updates and the organizations providing those updates) and human problem (e.g., getting the bureaucracies in organizations to address supply-chain issues).

There doesn't seem like there is much we can do to solve the supply-chain problems. Software, hardware, and co-design are global. Lots of systems and apps comprise components sourced from multiple suppliers. Governments do not control the supply chain, just as they do not control innovation or how products and services are used. Governments have been somewhat ineffectual in handling supply-chain issues. They complain a lot but don't seem to have workable solutions or good leadership in the supply-chain arena.

SolarWinds appears to have involved one or more insiders and external actors, making it challenging to perform a full damage assessment and nearly impossible to detect adulterated software components. Like I commented before, I don't see anything new here in SolarWinds. It is just that a lot of people are embarrassed (including top cybersecurity firms), and the media is having a field day. Why weren't the more sophisticated users of the software monitoring their own systems more closely?

It will be interesting to see the forthcoming attempts to apply artificial intelligence to detect the types of reconnaissance, command and control, and other aspects of penetration that were employed by SolarWinds. However, the first step should be to use a bit of common sense to think through the problem before trying to apply AI.

SolarWinds takes advantage of vulnerabilities in widely used platforms (.net) and protocols (e.g., Domain Name System). Can they be fixed? What does it take to get them fixed or replaced?

Initial findings indicate that advanced tactics, techniques, and procedures (TTPs) were employed by SolarWinds. The cyberdefense teams are always trying to catch up with the latest TTPs and new implementations of well-known TTPs used by the attackers. Addressing TTPs requires better communications between the offensive and defensive sides of the house.

Overreaction by governments to SolarWinds could have a profoundly negative effect on global innovation. Then again, governments are not in the driver's seat here. The private sector will continue to innovate.

The way forward may be to take a fundamentally new approach to security in the near- (e.g., cloud hypervisors pushing security rather than individual customers patching—or not), mid- (e.g., zero-trust, or something like it), and long-term (e.g., quantum-based approaches). The other part of the equation that needs to be addressed is motivation—getting the market to include security as a requirement. ■

Disclaimer

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the authors' employers.

References

1. "SolarWinds Security Advisory." SolarWinds. https://www.solarwinds.com/securityadvisory (accessed Dec. 24, 2020).
2. "How to compromise the enterprise endpoint." Google Project Zero. June 28, 2016. https://googleprojectzero.blogspot.com/2016/06/how-to-compromise-enterprise-endpoint.html (accessed Jan. 8, 2021).
3. A. Nappa, R. Johnson, L. Bilge, J. Caballero, and T. Dumitras, "The attack of the clones: A study of the impact of shared code on vulnerability patching," in *Proc. 2015 IEEE Symp. Security Privacy*, pp. 692–708. doi: 10.1109/SP.2015.48.
4. "Emergency directive 21-01." Cybersecurity and Infrastructure Security Agency, U.S. Department of Homeland Security. Dec. 13, 2020. https://cyber.dhs.gov/ed/21-01 (accessed Dec. 13, 2020).
5. T. Pericin. "SunBurst: The next level of stealth." Dec. 16, 2020. https://blog.reversinglabs.com/blog/sunburst-the-next-level-of-stealth (accessed Jan. 8, 2021).

Sean Peisert leads computer security R&D at Lawrence Berkeley National Laboratory, Berkeley, California, 94720, USA, and is an adjunct associate professor at the University of California, Davis (UC), Davis, California, 95616, USA. He is the editor in chief of *IEEE Security & Privacy*. Peisert received a Ph.D. in computer science from UC San Diego. He is a Senior Member of IEEE and the Association of Computing Machinery. Contact him at sppeisert@lbl.gov.

Bruce Schneier is a security technologist, fellow, and lecturer at the Harvard Kennedy School, Cambridge, Massachusetts, 02138, USA, and the chief of security architecture of Inrupt, Inc. Boston, Massachusetts, 02210, USA. He blogs at www.schneier.com. Contact him at schneier@schneier.com.

Hamed Okhravi is a senior staff member at the Massachusetts Institute of Technology (MIT) Lincoln Laboratory, Lexington, Massachusetts, 02421, USA. Okharavi received a Ph.D. in electrical and computer engineering from the University of Illinois at Urbana-Champaign. He is also the recipient of two R&D 100 Awards, MIT Lincoln Laboratory's Best Invention and Early Career Technical Achievement Awards, and NSA's Best Scientific Cybersecurity Paper Award. He is a Member of IEEE. Contact him at hamed.okhravi@ll.mit.edu.

Fabio Massacci is a professor at the University of Trento, Trento, 38123, Italy, and Vrije Universiteit, Amsterdam, 1081 HV, The Netherlands. Massacci received a Ph.D. in computing from the University of Rome "La Sapienza." He received the IEEE Requirements Engineering Conference Ten Year Most Influential Paper Award on security in sociotechnical systems. He participates in the FIRST special interest group on the Common Vulnerability Scoring System and the European pilot CyberSec4Europe on the governance of cybersecurity. He coordinates the European AssureMOSS project. Contact him at fabio.massacci@ieee.org.

Terry Benzel is the director of networking and cybersecurity research at the Information Sciences Institute of the University of Southern California, Los Angeles, California, 90292, USA. Benzel received an M.A. from Boston University and an executive M.B.A. from the University of California, Los Angeles. She is a senior member of the IEEE Computer Society, an associate editor in chief of *IEEE Security & Privacy*, and a member of the Board of Governors of the IEEE Computer Society. Contact her at tbenzel@isi.edu.

Carl Landwehr is a visiting professor at the University of Michigan, USA, where he received a Ph.D. He is an IEEE Fellow, a member of the inaugural class of the National Cyber Security Hall of Fame, and served as editor in chief of *IEEE Security & Privacy* for four years. He is a member of the board of the Center for Democracy and Technology. Contact him at celand@umich.edu.

Mohammad Mannan is an associate professor at the Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Quebec, H3G 1M8, Canada. Mannan received a Ph.D. in computer science in the area of Internet authentication and usable security from Carleton University. Contact him at m.mannan@concordia.ca.

Jelena Mirkovic is a project leader at the Information Sciences Institute of the University of Southern California, Los Angeles, California, 90292, USA. Mirkovic received a Ph.D. in computer science from the University of California, Los Angeles. Contact her at mirkovic@isi.edu.

Atul Prakash is a professor of computer science at the University of Michigan, Ann Arbor, Michigan, 48109, USA. Prakash received a Ph.D. in computer science from the University of California, Berkeley. Contact him at aprakash@umich.edu.

James Bret Michael is a professor in the Departments of Computer Science and Electrical and Computer Engineering, Naval Postgraduate School, Monterey, California, 93943, USA. Michael received a Ph.D. from George Mason University. He is an associate editor in chief of *Computer* and *IEEE Security & Privacy*. Contact him at bmichael@nps.edu.

Digital Object Identifier 10.1109/MSEC.2021.3060556