Turtles All the Way Down: Security at Every Level

Charles Wilson
Principal Engineer, Cybersecurity Development Lifecycle Practice

8/18/21 1:05:00 PM

Category: security-supply-chain

Tags: security, cybersecurity, autonomous vehicles, supply chain, AVCDL, OpenSSL, Heartbleed, open-source software

This is the first in a series on supply chain cybersecurity.

In the world of security, we like to believe that if we can secure the perimeter, we can protect the castle. The problem is that this belief isn't true, particularly when it comes to cybersecurity.

No Man Is an Island

There was a time when you could build yourself a castle with one entrance, surround it with a moat, and call it a day. Here's one such example, Beaumaris Castle [2] in Wales.

Over time, most of the top of the castle has been eaten away, and half the moat is gone. You can now simply walk up to the outer wall. That's just the way of things, both in the mundane world and that of technology.

For a very long time we didn't give a second thought to the security of most devices because they were isolated. Those same devices are now being connected to the internet at a furious pace. Their security moats have effectively been filled in.

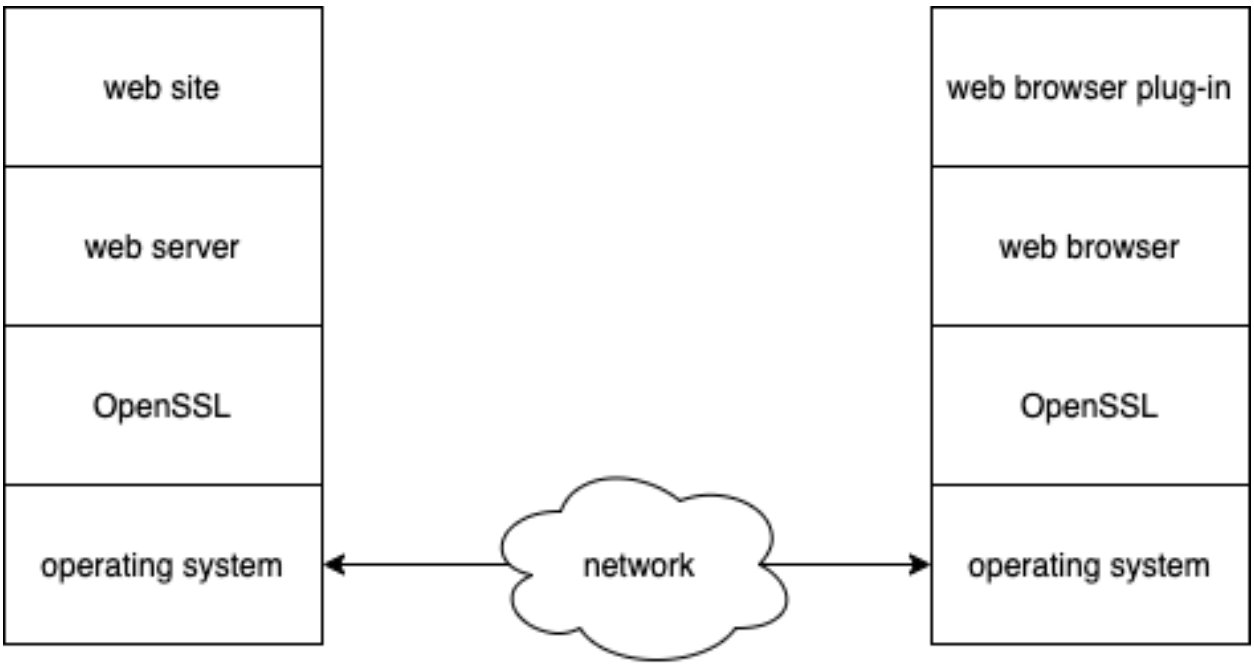
Since recognizing that a problem exists is the first step toward solving it, the industry is taking steps to ensure that we're secure, right? Not really.

Not It

Cybersecurity is really hard to do well. To make matters worse, when cybersecurity is done properly, it's invisible. Because of this, you can't use traditional metrics to show an ROI for cybersecurity. There's also a pervasive belief that cybersecurity is a system-level problem.

Combine these issues with the desire to get to market quickly and you create a situation which effectively disincentivizes security at every stage of the supply chain. Suppliers often defer security to their customers higher up on the supply chain. Manufacturers can err by focusing exclusively on their own security, assuming what they received from suppliers is secure. And then the end-user or consumer presumes they are purchasing and using a safe and secure product. At the end of the day, the last security team to the table gets saddled with the task of assuring the consumer that the product is secure.
Let's look at a concrete example.

OpenSSL [7] is open-source software used to secure communications. It is available on nearly every operating environment (Linux, Windows, MacOS, DOS, VMS, etc.). If you access a web site, OpenSSL is probably being used at one or both ends of the connection. Let's consider the following supply chain use case:



On the left is a simplified web server and, on the right, a simplified web browser. As the creator of a web site, you wouldn't think twice about how your communications are being secured. That's handled by the web hosting service.

The hosting service doesn't worry about communication security because that capability was provided along with the operating system. The operating system vendor doesn't give it much thought because OpenSSL has been around since 1998 and, since OpenSSL is open source, the community is responsible for reviewing it.

The right-hand side is basically a mirror of the left. At the end of the day, no one person, company, or entity is accountable for the security of OpenSSL.

In 2011 an error was introduced into OpenSSL. This error provided hackers with an opening to extract security-related information from the server side of a communications session secured by OpenSSL. The issue was identified in 2014 and given the name Heartbleed [8]. There was a lot of finger-pointing, and many articles written about how quality software costs money [9]; how it was the US government's fault [10]; and why this can happen again [11].

It should be noted that at the time OpenSSL was receiving about US$2,000 / year in donations. Per year. This lack of investment meant that the project could not fund a developer to ensure that this critical software was properly maintained.

This is a very public example of the impact that a failure within the supply chain can have on those higher up the chain. The problems were lack of security controls at the point of creation, and lack of assumption of responsibility for the verification of security-relevant aspects by the consumer (in this case the OS and web server vendors).

In a more recent incident, a casino was hacked through an internet-connected fish tank thermometer [12]. This shows that just because a component is doing something trivial doesn't mean that it doesn't deserve scrutiny.

Turtles All the Way Down

The phrase "turtles all the way down [1]" is typically used within computing to refer to infinite recursion [13]. I'd like to use it as a metaphor for how we need to view supply chain security.

Every component of a system at every level of integration needs to be secure. Rather than thinking we can get by with a moat, we need to consider the security of the elements supporting every participant in the supply chain and how our own behaviors can impact the security of those elements.

Consider a human pyramid [3].

If any person fails to provide support, the entire structure is at risk. Conversely, the stronger the levels are, the more structure they can support.

The technology supply chain for things like medical devices, aircraft, and vehicles (especially autonomous ones) is really deep. At every level there are numerous suppliers, each of whom may be using security-relevant technology (both hardware- and software-based).

The further away you get from the point of introduction of a technology, the less visibility you have into it. By the time you're two levels away, you may not even be aware of its presence. This is a major problem; so much so that U.S. presidents have issued Executive Orders [4, 5] trying to address the problem.

All or Nothing

It's one thing for the head of a government to mandate alignment within all departments; it's quite another to deal with the participants in your company's supply chain pyramid. So, what are we to do?

The only way forward is to:

1. Honestly assess the state of security throughout the totality of your supply chain
2. Acknowledge the level of security-related technical debt [6] present

3. Create a plan to manage the debt
4. Execute on that plan

This won't be painless or pretty or cheap, but it is necessary if we want to ensure that our ever-increasingly complex systems are reasonably secure.

But How?

The above four steps are fairly generic. What's needed are tangible solutions that can scale. In upcoming posts we'll cover three instruments to aid in ensuring the security of the supply chain.

- Cybersecurity Manufacturer Disclosure Statement
- Vendor Self-reported Cybersecurity Maturity Assessment
- Cybersecurity Interface Agreement

References

1. **Turtles all the way down**
   https://en.wikipedia.org/wiki/Turtles_all_the_way_down
2. **Beaumaris Castle**
   http://orapweb.rcahms.gov.uk/coflein/C/CD2003_612_028.jpg
3. **Human pyramid built by a "Falcons" team in Catalonia** (Pere López, **CC BY-SA 3.0**)
   https://en.wikipedia.org/wiki/Human_pyramid#/media/File:Escala_de_9_Falcons_Vilafranca.jpg
4. **Executive Order 13806 - Assessing and Strengthening the Manufacturing and Defense Industrial Base and Supply Chain Resiliency of the United States**
   https://www.federalregister.gov/documents/2017/07/26/2017-15860/assessing-and-strengthening-the-manufacturing-and-defense-industrial-base-and-supply-chain
5. **Executive Order 14017 - America's Supply Chains**
   https://www.federalregister.gov/documents/2021/03/01/2021-04280/americas-supply-chains
6. **Technical debt**
   https://en.wikipedia.org/wiki/Technical_debt
7. **OpenSSL**
   https://www.openssl.org
8. **The Heartbleed Bug**
   https://heartbleed.com
9. **Quality Software Costs Money – Heartbleed Was Free**
   https://queue.acm.org/detail.cfm?id=2636165
10. **The U.S. Government: Paying to Undermine Internet Security, Not to Fix It**
    https://www.propublica.org/article/the-u.s.-government-paying-to-undermine-internet-security-not-to-fix-it
11. **How Heartbleed Broke the Internet – And Why It Can Happen Again**
    https://www.wired.com/2014/04/heartbleedslesson/
12. **Criminals Hacked A Fish Tank To Steal Data From A Casino**
    https://www.forbes.com/sites/leemathews/2017/07/27/criminals-hacked-a-fish-tank-to-steal-data-from-a-casino
13. **Recursion**
    https://en.wikipedia.org/wiki/Recursion_(computer_science)