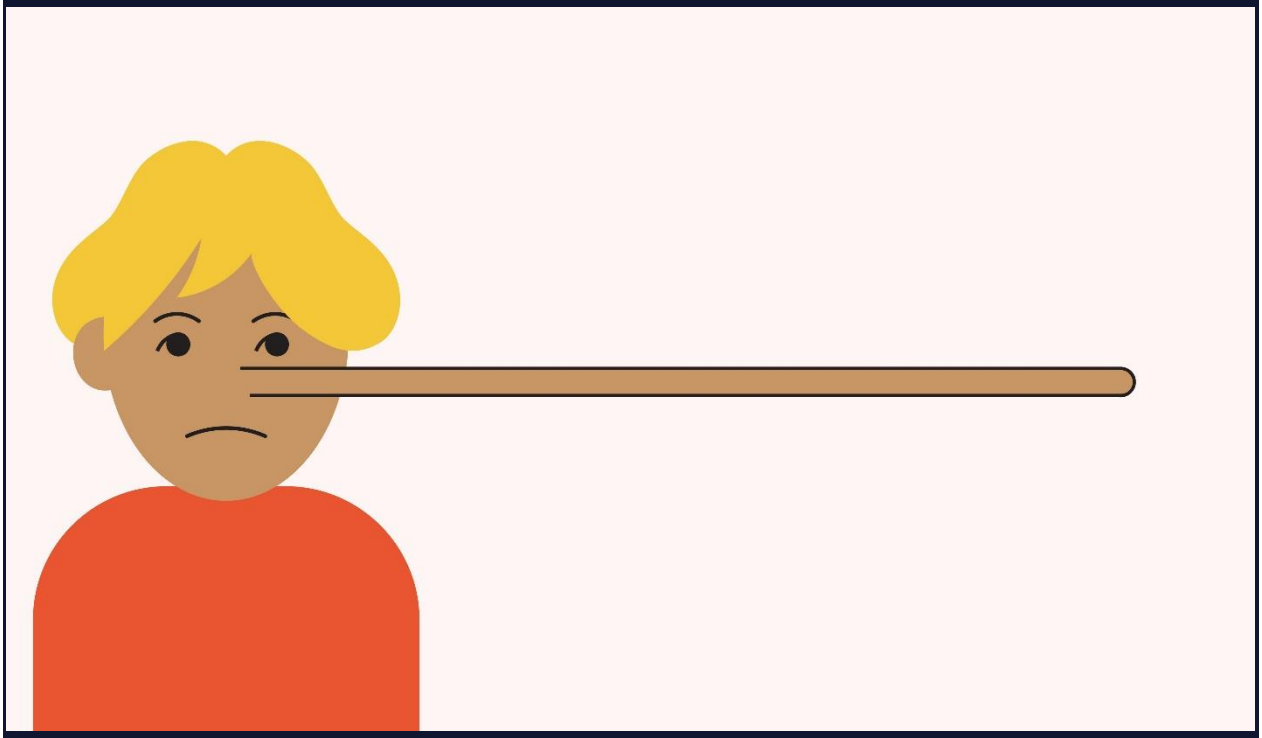
The World of Software and Data Integrity Failures

This article will discuss Software and Data Integrity Failures: what it is, how it happens, and why it's dangerous.



What Are Software and Data Integrity Failures?

Software and Data Integrity Failures (Integrity Failures, for brevity) are a class of vulnerability where insufficient measures are taken to ensure data and software integrity. In particular, it covers cases where data containing software or software components is acquired and used without verifying its integrity. Usually, but not always, this data is downloaded from the internet.

There are plenty of legitimate reasons why a program might want to download code or software: dependency management, updates, deserialization, etc… But, just because there's a good reason doesn't mean it can be done recklessly.

Many types of vulnerability, including Integrity Failures, involve data being treated as code. One thing that sets Integrity Failures apart from these other vulnerabilities is that the data is *intentionally* being treated as code in an Integrity Failure. The problem in Integrity failures isn't that the data is being

treated as code, it's that it's being treated as code *without being properly verified*.

When Do Software and Data Integrity Failures Happen?

Integrity Failures can happen when untrusted data is deserialized. Deserialization is the process of reading external data and turning it into an object integrated into a program. Deserialization is a known attack vector, and there are dedicated toolkits for attacking various languages' deserialization processes. This can be a real problem if, for example, a web app serializes data, stores that data in the user's browser, and then retrieves and deserializes that data later without verifying its integrity.

Another common way Integrity Failures can happen is, as we mentioned, through software updates. Software updates are very important for security, but they must be done correctly. The easiest way to do it incorrectly is not to verify the origin of the update. Many routers and IoT devices don't properly verify the authenticity of updates, and attackers can and do try to trick these devices into downloading and applying malicious "updates." Verifying the origin and integrity of updates makes it harder (but not impossible) for attackers to hijack the update system for their purposes.

Finally, many pieces of software rely on other software, known as dependencies. These dependencies might be separate from the software or integrated into it when the software is built. If these dependencies are satisfied by downloading software, it is, again, very important to verify the integrity of the dependencies. It's bad enough to have an installer download something malicious onto someone's PC, but it's arguably worse to accidentally embed malware into your program.

Why Are Software and Data Integrity Failures Dangerous?

These Integrity Failures provide a relatively simple and direct path for an attacker to compromise a target. In a worst-case scenario, an integrity violation could allow an attacker to run arbitrary code on a target device - which is bad. With defense-in-depth, the immediate consequences might be less severe, but it can still give an attacker a foothold that can be difficult to detect.

One of the most famous examples of an Integrity Failure is the 2020 breach of SolarWinds. A nation-state actor was able to compromise SolarWinds, a cybersecurity company, and compromise their software update process. They then used the update process to distribute malware to thousands of organizations, including US federal agencies.

Conclusion

Integrity Failures are a type of vulnerability where data is used without its integrity being properly verified first. It typically relates to cases where the data is software or software components. It can happen in software updates, dependency management, or even simple deserialization. Integrity failures pose serious risks, including risk for arbitrary code execution, and Integrity failures have played a part in major cyberattacks in the past.