Srinivas Mukkamala Testimony

The Subcommittee on Cybersecurity and Infrastructure Protection
"Design vs. Default: Analyzing Shifts in Cybersecurity"
Thursday, December 5, 2024

Software errors can be a significant reason for safety issues, as malfunctions caused by bugs in code can lead to dangerous situations in systems that control critical functions like medical devices, transportation systems, or industrial machinery, potentially causing harm to users or the environment if not properly addressed.

A good example and a case study in most US universities software integrity course curriculum is The Therac-25.

The Therac-25 is a computer-controlled radiation therapy machine produced by Atomic Energy of Canada Limited (AECL) in 1982. It was involved in at least six accidents between 1985 and 1987, in which some patients were given massive overdoses of radiation. Because of concurrent programming errors (also known as race conditions), it sometimes gave its patients radiation doses that were hundreds of times greater than normal, resulting in death or serious injury. These accidents highlighted the dangers of software control of safety-critical systems.

We have 2 alarming scenarios at hand as I testify on Dec 5, 2024:

- A report published by Forbes: **A Trillion-Dollar Global Tech Problem,** a recent study examined the increase in tech debt from 2012 to 2023 across industries and regions reveals global tech debt has nearly doubled over this timeframe, increasing by around $6 trillion.
    - In the United States, three sectors are responsible for 64% of the estimated $2.2 trillion rise in tech debt: banking and investment services; communications, media and services; and government.
- $1.14 Trillion to Keep the Lights on: Legacy's Drag on Productivity published by Mechanical Orchard. January 23, 2023.

Legacy systems run code or rely on libraries that might contain known security vulnerabilities with no way to patch these. Dated, insecure code increases the attack surface for a business.

In its 2019 study of several critical federal government systems, GAO noted that several of the legacy systems were operating with known security vulnerabilities and unsupported hardware and software.

What is alarming is the lack of transparency and a catalog of legacy systems and software serving critical infrastructure. We have several initiatives to catalog know software (NSRL - The National Software Reference Library) and known vulnerabilities

(NVD – National Vulnerability Database). An initiative to catalog legacy software and publish know weakness and vulnerabilities will allow entities to better understand the Risk posed by legacy software.

Key Point 1: Transparency and Lack of Accountability

Initiatives like "**Secure by Design" and Build Security In (**a 2005 **initiative)** bring the required awareness that is needed to address the problem at hand.

"**Secure by Design"** is a voluntary pledge focused on enterprise software products and services. By participating in the pledge, software manufacturers are pledging to make a good-faith effort to demonstrate measurable progress towards the following areas.

We can't afford good faith in safety. Recent attacks have demonstrated the impact on critical infrastructure from Water Utilities to Food Processing units.

What's needed is radical transparency on the pledge, a catalog of software by the vendor what is covered and what is not. A clear timeline to demonstrate progress in securing legacy and vulnerable software that is been neglected for years.

How Food and Drug Administration (FDA) holds drug and product manufacturers automatically liable for any harm caused by their product. Congress can also establish a software liability regime for software manufacturers.

Key Point 2: Defining Classes of Vulnerability and Developing a Taxonomy

One of the goals in the pledge is to reduce entire classes of vulnerability. Within 1 year of signing the pledge, it will demonstrate a significant reduction in the prevalence of 1 or more vulnerability classes across the manufacturer's products.

With out a proper definition and a prescriptive list of Common Weaknesses and Vulnerabilities that needs to be reduced or eliminated there is room for interpretation and not yield desired results.

Based on our research and analysis of 25 Most Dangerous Software Weaknesses list (CWE™ Top 25) is the positioning of memory buffer operations. While MITRE ranks this at #20, it consistently appears as the top target for both threat groups and ransomware operators.

We also observed significant evolution in the threat landscape, particularly with the emergence of AI/ML systems.

Organizations must adapt their security priorities to address these changes, with special attention to emerging AI/ML-specific vulnerabilities that may not yet be reflected in standardized rankings.

Key Point 3: A Safety Seal for Software

Software runs multiple critical infrastructure sectors. What is rapidly changing is the digital transformation and autonomous nature of how these systems operate today. There are several sectors that follow rigorous quality tests and checks before they are deployed in production. Congress can establish a safety framework for software industry as well.