Chris Yocum

ECE531 Intro to IoT – Summer 2024

Analysis of NC220 Firmware

Introduction:

Internet of Things (IoT) devices today are more common than ever. Companies extend from small startups to conglomerates such as Amazon. They extend from controlling lights in homes, doors and locks, microwaves, coffee makers and, now, refrigerators. Washing machines and dryers have been outfitted with software, effectively making them IoT devices. And this leaves people vulnerable to attacks. Hardening is not only necessary but a way of life. It has become a silent war.

The Dynamics of Hardening:

Hardening refers to the attack surfaces of IoT devices. There are several roles of responsibilities, depending on a person and the position in society. The roles are hardware manufacturers, software application or code creators, and users. Each role has control over their area and is only capable of influencing those in the other roles. Let's discuss them.

The Hardware Manufacturers:

The hardware manufacturers create hardware with public and private keys. These keys validate certain aspects of the bootloader as the device progresses towards the kernel. This ensures that the device has a secure kernel to work from. The manufacturers are frequently updating the firmware to ensure that vulnerabilities are resolved. The private key is protected by the company while the public key is embedded during the manufacture process.

The Application Coder:

The software developer or application coder utilizes firmware/bootloader of the device that the hardware manufacturer publishes. The coder depends on regular updates from the

manufacturer to ensure that the device that the user utilizes is secure. This mitigates the risk, financially and privacy among others, for all participants involved.

The coder needs to create configuration files with the proper configuration to that particular hardware to ensure that only what is needed to run the IoT device makes it to the user. Anything more or a misconfiguration could result in a weakness that could cause a ransomware incident. Other incidents are also certainly possible.

The daemon that is loaded upon start up needs to have arguments and input validation checked for best coding practices. The functions used will need to be screened for potential bypass or other vulnerabilities. Does printf and syslog functions have any exploitable weaknesses, either in themselves or their implementation. The buffer overflow may be a concern if the system has 32 bit registers and fixed memory locations upon startup. Has SSH access been implemented and transporting data over SCP?

The coder also needs to consider network traffic the device will be connected to. The type of connection may be HTTP originally but only to have connection switch over to HTTPS. HTTPS is encrypted and often uses SSL or another method.

The user will log in and validate credentials. The coder has an interest in reminding the user to choose a strong password and perhaps provide an indicator on how strong of a password it is.

What if the IoT device was a physical access device such as an electronic lock on the front door? The coder may have to choose hardware that has no physical access from the outside. Furthermore, the coder will also need to refrain from hard coded passcodes that an educated intruder would have access to. This would be unlike a default factory password for a router within a secure home.

The coder needs to recognize that databases, file systems and configuration files have weaknesses that can be exploited. Certain databases may have code embedded them that may be accidently executed, leaving the device vulnerable.

If there are known vulnerabilities and wish to be documented, it is best practice to document them in a document file type and not in the code itself. It is on the coder to recognize at the time of creation what type is most secure.

The User:

The user has a responsibility to research what devices are known to have common issues. One particular device may be a Network Allocation Storage (NAS) unit for home data storage. Synology is one of many and comes with applications built in that have external webpage access to data. It comes on by default. Surely, the Synology Red Team tried to hack it in penetration testing and found something. Either way, I personally bought one and the web access was on by default. No data was stored on it but a hacker used a published exploit, didn't have anything to ransomware me with so he/she changed my password. Fortunately, Synology published the factory password to reset the device from within my secure home and leave it offline long enough to turn off this vulnerable feature.

The user also needs to either choose a random password or follow conventions such as 10-15 characters which may be letters, numbers and/or special characters. But not everyone thinks of these. Some have poor memories. There are such device applications now that will assist in keeping passwords. Web browsers have them too. This may be a solution as they have built in encryption. Is it the best solution? No one knows without consultation of others.

Conclusion:

The small daemon program attached may be implemented into any number of devices. The device will boot up the daemon program as part of the initiation process. This will monitor the

application and perform the functions that is required of it until it dies, the device dies, or the user kills it. This feature can be used to harden the application and IoT device.