Comprehensive Exam - Question #2

Rob Gillen

Question

What impact do Real Time Operating Systems have on the vulnerabilities of cyber-physical systems. In other words, what characteristics do RTOSs have that make them more or less vulnerable (than general purpose operating systems), and what are some schemes attempted to mitigate these vulnerabilities.

Answer

Real-time Operating Systems (RTOSs) are, in many ways, similar to general purpose OSes and have, in recent years, gone through similar adjustments due to increasing threats. The primary difference is that RTOSs started out targeting disconnected/private networks and therefore were less susceptible to malicious actors. The developers of such systems focused on task scheduling and the real-time nature of their systems (the key objectives of the system) rather than hardening them from outside influence. As these systems have been increasingly connected to corporate networks and eventually to the Internet itself, the maintainers of these OSes have been forced to adopt techniques and protections that have long been present in general purpose OSes but must implement them in a manner that does not impede their schedulers. What follows is a list of advantages, disadvantages and some mitigations for these related to RTOSs.

Security Advantages

From the standpoint of platform security, there are some significant advantages to utilizing an RTOS. Some of these include near determinism, reduced attack surface, and limited hardware resources.

Determinism: RTOSs do not provide the same level of clock synchronicity available in an FPGA or ASIC, but the are designed to support time-dependent repeating tasks. This regularity can be a significant asset when monitoring the data coming into/going out of one of theses systems - the traffic patterns should exhibit a consistency that eases the tasks of anomaly detection.

Reduced Attack Surface: Because these OSes are not general-purpose, they are generally streamlined for a specific category of tasks or operations. As such, the quantity of software packages included in the OS and code compiled into the kernel is often far less that would be present in a general OS.

Limited Connectivity: While not a factor specifically of an RTOS, many of the control systems running an RTOS in cyber-physical system run in a private network or, at least a segmented network. As mentioned above, however, the increasing connectedness is rapidly diminishing.

Security Disadvantages

Memory Isolation: This is less of an issue with modern versions of RTOSs, but earlier on there was such a focus on time consistency that many of the process and memory isolation techniques we see today were not

incorporated into these systems. This allowed one malicious process to read and potentially overwrite data from other processes.

Determinisim: While previously listed as an advantage, the consistency of these systems can also present a weakness. Imagine, if you will, a system that polls for data once every 2 seconds and takes 0.1 seconds to complete. An attacker can observe this traffic pattern and then time their attack to function within the remaining 1.9 seconds and, so long as they return the state to "normal" before the polling occurs, they will proceed undetected (similar to pulse-width modulation).

Less "Hardened" External Services: This is not universally true, but based on the historical origins wherein many of these systems lived only on trusted networks, the attention paid to the security robustness of their IP services is often less than general OSes. As a colloquial example, just last week I was working with an SEL-351 box (electrical grid relay protection system) that had telnet and FTP services (all clear text) enabled by default.

Limited Hardware Resources: This is not necessarily unique to RTOSs but more to the notion of embedded systems in general. The fact that many of these operating systems are running on constrained hardware has a very clear impact on the security of the systems. For example, on a low-powered processor, the act of encrypting and decrypting data may be too computationally expensive to be allowed given the tight constraints of the scheduler.

Reduced Automated Patching/Updates: There are a handful of related issues that are bundled into this single heading. Due to the types of systems onto which these OSes are normally deployed (24x7 ops, deploy-and-forget appliances, etc.) the likelihood that these devices will be configured to automatically update themselves as security patches are made available is incredibly low. The growing connectivity to these devices only serves to exacerbate this issue. Closely coupled with this is the reduced set of enterprise-level management tools available for these OSes.

Mitigations

Since the public disclosure of Stuxnet, there has been an increase in the amount of research in the area of cyber physical systems. Many security researchers are taking a closer look at the devices and software that control the systems that affect our daily lives. Some of the mitigation work is listed below:

Memory/Time/Resource Isolation: This has likely been the largest single change in the RTOS domain relative to security. As early as 2004 organizations such as the FAA, NSA and others were pushing RTOS developers to support security standards such as ARINC 653[1], DO-178B[2], and EAL-7[3]. These standards require isolation between processes and memory accesses among other stability and safety related properties. In more recent years standards such as these have been updated, improved, and increasingly required by those acquiring RTOS-based systems.

Security-Focused RTOS: Much like the general OS market has security-hardened versions of the most common OS variants, there have emerged security-enhanced RTOSs. Also, based on the standard requirements described above, many of the previously-existing vendors have strengthened their compliance and security controls. Examples of security-aware or security-hardened RTOSs include (partial list): Integrity RTOS by Green Hills Software[4], FreeRTOS[5], QNX[6], and VxWorks[7].

System-Call Based Anomaly Detection: In their paper[8] Cheng et al. present an anomaly-detection system that monitors system calls to attempt to determine if someone is performing data-injection attack. They are specifically focused not on network-based attacks, but local-device attacks wherein an attacker attempts to modify the memory locations holding the data on which the control system responds. In these cases, the logical control flow of the application has not been violated, but the data is simply wrong, causing the program to respond in a fashion consistent with its programmed logic but contrary to the actual physical state of the system. Systems such as this could be used to help mitigate the risks of shared memory models in deployed RTOS platforms.

Traffic Classification Systems:

In a fashion similar to that in the IT space, the data science community has turned its attention to monitoring and protecting the networks supporting control systems. While there are many similarities between IT and OT networks, the protocols, structures, and physical media are often unique. Work in machine-learning based classification[9][10] of traffic flowing into and out of these specific systems is progressing.

A Potential Bellwether

While not explicitly stated that they were dealing with RTOS-based devices, Formby et. al. [11] performed a three-year longitudinal study of the network traffic within a power company's substations for the purpose of characterizing the traffic and they uncovered a number of interesting items that are relevant to this topic.

The traffic was *not* nearly as regular and predictable as one would expect, even given a highly over-provisioned network. There is an assumption among many that in a relatively isolated network that has more capacity than needed and contains only embedded systems devices (RTOSs, bare-metal) monitoring a physical system, the traffic should be highly regular and consistent. This type of environment should (logically) be a perfect fit for anomaly detection systems as the model of "normal" should be clear and easy to build and abnormal traffic should be clear and obvious. Contrary to this assumption, they found significant variance in response times, in inter-packet timings, etc. They performed the majority of their work in one substation and validated it in two others (all the same company, however).

The traffic was unpredictable in predictable ways. What I mean by this, is that they were able to model and cluster the features from their traffic analysis and, given their access to ground truth, could confirm that the devices from a given manufacturer behaved consistently. This allowed them to monitor the traffic and then predict with high accuracy the manufacturer of a given device on the network (fingerprinting). We have long been able to do similar assessments in the traditional IT network realm to predict device operating systems and versions (e.g. NMAP[12]) but not the hardware manufacturer. The assertion of Formby et. al. is that this is likely due to the fact that the devices under consideration were initially designed (as were their programming logic) to perform a particular function such as monitoring current and voltage and then opening a relay if either are too high. Pressures to integrate systems have driven manufacturers to add LAN/WAN capabilities which were secondary to the primary objective of these systems and were therefore possibly under-supported via system resources. The network interactions become, essentially, "best effort" relative to the primary device functionality.

References

[1] S. ITC, "653P5 avionics application software standard interface, part 5, core software recommended capabilities." [Online]. Available: https://www.aviation-ia.com/products/653p5-avionics-application-software-standard-interface-part-5-core-software-recommended-2.

[2] I. RTCA, "Software considerations in airborne systems and equipment certification." [Online]. Available: http://listofavailabledocs.realviewdigital.com/?iid=158440&crd=0&searchKey=178B#folio=104.

[3] C. Criteria, "Common criteria for information technology security evaluation." [Online]. Available: https://www.commoncriteriaportal.org/cc/.

[4] G. H. Software, "Integrity real-time operating system." [Online]. Available: https://ghs.com/products/rtos/integrity.html.

[5] A. W. Services, "FreeRTOS - market leading rtos." [Online]. Available: https://www.freertos.org/.

[6] a subsidiary of B. QNX Software Systems Limited, "QNX operating systems." [Online]. Available: https://blackberry.qnx.com/en.

[7] I. Wind River Systems, "VxWorks." [Online]. Available: https://www.windriver.com/products/vxworks/.

[8] L. Cheng, K. Tian, D. Yao, L. Sha, and R. A. Beyah, "Checking is believing: Event-aware program anomaly detection in cyber-physical systems," *CoRR*, vol. abs/1805.00074, 2018.

[9] J. M. Beaver, R. C. Borges-Hink, and M. A. Buckner, "An evaluation of machine learning methods to detect malicious scada communications," in *Proceedings of the 2013 12th international conference on machine learning and applications - volume 02*, 2013, pp. 54–59.

[10] R. C. B. Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," in *2014 7th international symposium on resilient control systems (isrcs)*, 2014, pp. 1–8.

[11] D. Formby, A. Walid, and R. Beyah, "A case study in power substation network dynamics," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 1, pp. 19:1–19:24, Jun. 2017.

[12] G. Lyon, "NMAP: THe network mapper." [Online]. Available: https://nmap.org/.