CS3551 Class Project Proposal

Mike Boby and Brad Whitehead
19 February 2020
Version 2.1

Project Title - Using Unikernels to Enhance the Attack-Resistance of Spire, a Network-Attack-Resilient Intrusion-Tolerant SCADA for the Power Grid

Project Goal - Convert the traditional ELF executables of Spire 1.2 [2] to self-contained unikernels and demonstrate that 1) they continue to operate correctly and 2) they exhibit the increased performance and reduced resource utilization characteristics of unikernel technology. Investigate the combined approach of using unikernel technology to reduce attack surfaces, and the Multicompiler to create a “moving target defense” while reducing “return-oriented programming (ROP)” vulnerabilities. If possible, demonstrate the increased compromise resistance of the unikernel(/Multicompiler) Spire executables.

Why:

Considerable thought and effort has been applied to the problem of making the executables used in the Spire system resistant to attack and successful compromise. This includes the use of the MultiCompiler to create polymorphic executable versions of the source code. While this is an excellent stand-alone approach, we believe that security and compromise resistance could be further enhanced by discarding the use of an operating system and converting the executables into unikernels, isolated from other applications through hardware-enforced virtual machine technology. Not only will this increase the compromise resistance, it will significantly enhance performance in the areas of initialization (“bootup”) and throughput, as well as decreasing resource utilization (memory). Unikernels are highly resistant to most attack vectors, with the exception of ROP exploits. While unikernels are less susceptible to ROP attacks because of the increased difficulty of initially compromising the executable, if a beachhead is established the unikernel provides no more defense against ROP than a conventional executable. If it is possible to use the Multicompiler in conjunction with a unikernel, the ROP vulnerability can be greatly reduced, closing the remaining gap in unikernel security[1].

Prior Research:

While there are a number of publications on the unikernel concept and its applicability to security, since the seminal paper in 2013 [3], we were only able to find one paper that specifically addressed the use of unikernels in a SCADA environment [4]. In this paper, unikernels were selected not for their security properties but rather for their fast instantiation and low memory requirements. There are two other papers that mention the possibility of using unikernels in industrial networks [5,6], but both authors felt that the unikernel orchestration systems were not mature enough. Interestingly enough, both papers chose to use containers

instead. While container orchestration systems such as kubernetes and mesos may be more mature, containers themselves have a number of well known and documented security issues. We believe that unikernels are sufficiently mature and that the minimal orchestration required for Spire is easily achievable.

Anticipated Project Steps:

1) Familiarization with the Spire system by obtaining the required external dependencies, compiling the code, and running the author-supplied benchmarks

2) Investigate the compatibility of unikernels and the Multicompiler. Consider the building order; should Multicompiler output be linked into unikernels, or should the Spire and unikernel source code be compiled by the Multicompiler?

3) Research available unikernel libraries, build systems, orchestration systems, and virtual machines, and select the most appropriate ones based on observations from Steps #1 and #2

4) Select an appropriate paper on unikernels and security to present in class

5) Identify evaluation hardware (either several “bare metal” servers or nested virtualization on one or more virtual servers)

6) Compile the Spire executables into unikernels, using the libraries and build system identified in Step #3

7) Iteratively, make necessary code changes required to accomplish Step #4

8) Test and benchmark Spires unikernels using the included benchmark suite and the test configuration in the “Read Me” file

9) Investigate the compromise resistance of the Spire unikernels. This step is dependent on the availability of any existing compromise/penetration tests or test tools, or the availability of the University’s CyberSecurity Club

10) Document the project

11) Prepare and deliver project presentation for class

Synergy - It is expected that this project will collaborate with another class project involving network intrusion detection for the Spines network. We anticipated this collaboration will involve the exchange of ideas and information (joint “brainstorming”), shared hardware, and joint participation in vulnerability/penetration testing.

References:

[1] Andrei Homescu, Steven Neisius, Per Larsen, Stefan Brunthaler, and Michael Franz, Profile-guided Automated Software Diversity. International Symposium on Code Generation and Optimization 2013, https://pdfs.semanticscholar.org/bdea/c542d3bdb9fd3666c2cef04ef4b20be14830.pdf (retrieved February 19, 220)

[2] Babay, A., Tantillo, T., Aron, T., Platania, M., & Amir, Y. (2018). Network-Attack-Resilient Intrusion-Tolerant SCADA for the Power Grid. *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 255-266.

[3] Anil Madhavapeddy, Richard Mortier, Charalampos Rotsos, David Scott, Balraj Singh, Thomas Gazagnaire, Steven Smith, Steven Hand, and Jon Crowcroft. 2013. Unikernels: library operating systems for the cloud. SIGARCH Comput. Archit. News 41, 1 (March 2013), 461–472. DOI:https://doi.org/10.1145/2490301.2451167

[4] Sakic E. et al. (2018) VirtuWind – An SDN- and NFV-Based Architecture for Softwarized Industrial Networks. In: German R., Hielscher KS., Krieger U. (eds) Measurement, Modelling and Evaluation of Computing Systems. MMB 2018. Lecture Notes in Computer Science, vol 10740. Springer, Cham

[5] Ahmed Ismail and Wolfgang Kastner, Vertical Integration in Industrial Enterprises and Distributed Middleware, Int. J. Internet Protocol Technology, Vol. 9, Nos. 2/3, 2016

[6] Spyridon V. Gogouvitis, Harald Mueller, Sreenath Premnadh, Andreas Seitz, Bernd Bruegge, Seamless computing in industrial systems using container orchestration, Future Generation Computer Systems, 2018, ISSN 0167-739X, https://doi.org/10.1016/j.future.2018.07.033.