Project Summary (Section B): An End-Host Security Analysis and Training Environment

Trent Jaeger  Luke St. Clair  Sandra Rueda  
Will Enck  Patrick Traynor  Kevin Butler

The Pennsylvania State University  
Department of Computer Science and Engineering

Faculty POC: Trent Jaeger, tjaeger@cse.psu.edu and (814) 865-1042  
Note: All other participants are CSE graduate students

1 Summary

We propose building a computing environment, which we call the Playpen, to evaluate the configurations of end-host systems. The Playpen aims to provide in-depth (e.g., destructive) testing of configurations in such a way as to be useful for analysis of active end-host system configurations and for training of computer science students, system administrators, and others in solving practical problems in computer security. Tools for teaching computer security are rare, limited in scope, and primitive in function. For example, students are generally given a limited system (e.g., the Minix operating system) or examine only a subset of a security system (e.g., part of Java crypto library). The Playpen will provide a real, complete system, so training can go beyond introductory to actual hands-on administration of all facets of real systems. Further, current security analysis tools (e.g., SAINT) provide an external analysis of possible vulnerability paths, but the Playpen will be able to show the attack vectors and forensic results, because it will be able to support destructive testing. The Playpen work will consist of adding management features to the open-source Xen virtual machine system, to enable configuration of actual systems, a variety of testing upon that system, and checkpoint features to return to pre-attacked systems.

2 Product Need

The rise of the Internet as the foundation of computer-supported commerce is based on a model of an untrusted network to which smart end-hosts connect to receive services. Despite the lack of trust in the network, a seemingly robust infrastructure has been created that everyone relies on for their day-to-day needs. Every commercial entity has representation on the Internet, and for some, like Amazon and eBay, it is a lifeline.

The main flaw in the assumption that the Internet is an effective foundation for commerce is that the intelligence of the end-hosts is quite overrated. We have seen a lot of worms and viruses over the years exploiting the end-hosts, and we have also seen denial of service attacks on popular web-services. Cyberattacks are becoming more common, and they are getting more sophisticated as the attackers learn to get around the established defenses.

The problem is that many end-hosts are managed in an ad hoc way by inexperienced personnel. Training to provide administration skills is generally provided on-the-job by dealing with real attackers. For example, a student may start with a part-time job in the computer center where she learns what kind of countermeasures may be effective from particular kinds of attacks that the system sees or has seen. The principles that the student learns depend on the specific attacks seen, and often such administration becomes a trial by fire.
The two main problems that administrators face are maintaining a secure software suite and configuring the myriad of services on the system. In the first case, the problem is to recognize vulnerable software and make updates in a timely fashion. Automated updating has found its way into several software distributions, so administrators must understand the correct use of such services and remediation of exceptional cases that are not covered (e.g., zero-day worms). Second, administrators use a lot of different defenses, such as firewalls, application configurations, mandatory access control, virtual private networks, and intrusion detection systems, to defend the hosts. Each of these tools are complex in their own right, and the security afforded by the combination is challenging to understand.

Our proposal is to build a computing environment where end-host configurations can be constructed and tested. In general, such an environment can be applied to: (1) security testing of deployed configurations and (2) training of students and system administrators in the protection of end-hosts. Such an environment will provide hands-on experience in software suite maintenance and configuration management.

3 Commercial Potential

In our estimation, the commercial potential of the proposed product lies mainly in three areas: (1) developing the test cases to apply in the Playpen environment; (2) automating the packaging of systems for migration to the Playpen environment for testing; and (3) evaluating and presenting the results of the Playpen analysis.

First, for current commercial products, the state of the art is a set of products that provide vulnerability assessment. These products (Saint Corp.’s SAINT, Tenable’s Nessus, Symantec’s NetReCon, eEye’s Retina, etc.) test networks of systems to find vulnerabilities on any of these systems, such as old or misconfigured services. These products test the network from the outside by determining what kinds of services are running, whether they are vulnerable versions, and whether they are configured in a vulnerable manner. The main strengths of these products are their test set (generated from known vulnerabilities) and reporting mechanisms. Reviews in the 2001-2 timeframe indicate that the performance of these tools was spotty (some well-known vulnerabilities were not identified), although recent review data is not available.

We would expect that the development of test cases for the would be derived from known vulnerability, such as the CERT or SANS lists, much as for these tools. The difference is that the Playpen enables the tests to be destructive. Vulnerability cannot run destructive tests because they test real machines. In the Playpen environment, we aim to test snapshots (i.e., copies) of real systems run as virtual machines. Because we are testing a copy, we can run destructive tests.

Second, in order to test a copy of a system, we must have a mechanism for generating system copies. We plan to develop a client component, called the snapshot client, that runs on the machine to be tested to generate a snapshot of that system useful for testing. We need not create an exact copy, as typically user files are not relevant for testing, but the snapshot client will have to be smart enough to catch all the services that are running. Our present prototype environment runs on standard x86 hardware and our current VMs must be customized for this environment. The new virtualization technology processors from Intel (Pacifica from AMD) enable execution of systems without the need for modification, so we will be testing snapshots for this hardware.

Third, we will have to develop mechanisms for evaluating test results and reporting these to the user. In the Playpen tests so far in our undergraduate course, the testing code writes the test results back to the server where they are collected. Our tests thusfar are pretty simple, so extensive reporting remains an area in need of investigation. We expect that we can leverage vulnerability lists for information to report on failures, although a more detailed forensic analysis of how the vulnerability was leveraged by the test is also possible. In the Playpen, we can collect much finer-grained information than previous vulnerability scanners.