The Blunderdome: An Offensive Exercise for Building Network, Systems, and Web Security Awareness

George Louthan, Warren Roberts, Matthew Butler and John Hale
{*george-louthan, warren-roberts, matthew-butler, john-hale*}*@utulsa.edu*
Institute for Information Security
The University of Tulsa
800 S. Tucker Dr.
Tulsa, OK, 74104

Abstract

In spite of the controversy surrounding the practice of using offensive computer security exercises in information assurance curricula, it holds significant educational value. An exercise and architecture for an asymmetric (offense-only) security project, nicknamed "Blunderdome", has been deployed twice at the University of Tulsa: once to graduate students in a security engineering course, and once to high school students as part of a research internship program. This paper discusses the framework, the project, its educational value, and lessons learned for future deployments. Coverage is also given briefly to a summary of our position on the role of offensive exercises in security education.

1 Introduction

In 2009 as part of a course entitled *Information System Security Engineering* at the University of Tulsa, students were required to complete a linear, highly constrained offensive computer security exercise for the purposes of illustrating the importance of secure design principles taught in the course and promoting discussion of the engineering failures that enabled the students' penetration of the systems involved. The exercise, nicknamed *Blunderdome* and deployed as a final class project, involved the staged remote penetration of a simulated grades management system on an isolated, virtual "academic intranet."

Later that same year, a group of high school students spent the summer as research interns at TU's Institute for Information Security and completed a virtually identical exercise, this time with the dual goals of illustrating the importance and nature of the research they would be doing, as well as providing a fairly comprehensive motivating project with which to mentor them in various computer technology principles including networking, SSH, Linux command line, POSIX access controls, scripting, compiling software from source, and others.

Both deployments had notable successes and failures and demonstrated important lessons in the role of offensive educational exercises, which are discussed in this paper. The remainder of this paper is organized as follows. Section 2 provides some background into offensive exercises in information assurance education; Section 3 gives an overview of the exercise, description of the stages of the attack, and some details about the architecture supporting it; Section 4 describes the results and lessons learned from each of the two Blunderdome deployments to date. Section 5 draws conclusions, uses our experiences to make recommendations on the role of an offensive exercise in security education, and discusses the future of the Blunderdome.

2 Background

Related testbeds and simulated environments for purely or partly offensive cyber security exercises have been deployed for the purposes of collegiate education. For example, the University of California at Santa Barbara included a symmetric "capture the flag" exercise in a network security course [14], similar to those popularized at DEFCON and deployed elsewhere [8, 10, 13].

Much of the work in developing testbeds for security exercises has focused upon building flexible systems suitable for exploration or dynamic exercises with multiple participants [11] to allow "a large range of attack and defend activity" [12]. In contrast, Blunderdome was designed with the specific goal of inflexibility; students were expected to take a particular path to the final goal of changing their grade, and each step of that path was carefully planned.

Additionally, the Blunderdome testbed itself is similar to Foundstone's Hacme series, which includes a vulnerable banking web application [5], the Open Web Application Security Project's WebGoat, which attempts to be an "interactive teaching environment for web application security" [9], and other related vulnerable sys-

tems. Though Blunderdome can rightly be considered a descendant of these projects, it differs from them primarily because of its addition of network and systems security components, its strictly enforced linearity, and its clear goal-oriented narrative.

3 Exercise Details

3.1 Overview and Architecture

The objective was to provide students with a vulnerable but real world system that they would be able to completely control given only publicly available information about the system. The Blunderdome testbed simulates an academic network and grade management system, logically comprised of three main components: a web service, a login server, and a firewall.

Web Service A grades management web service runs on an "internal" web server, allowing access based upon a username and password pair; this service is vulnerable to SQL injection and was implemented as a single web server for the entire class backed by a separate database for each student.

Login Server An Ubuntu Linux server running OpenSSH with no additional patches installed is designated for each student; remote authentication is enabled using public keys only. This server simulates a remotely accessible machine within the academic network and serves as the entry point to the network. A root-owned file (readable only by the root user) on the login server contains the address and access credentials for the web service. Each student received different access credentials. Ubuntu 7.10 was chosen because it shipped with both key generation and privilege escalation exploits.

Firewall Largely off-limits for the purposes of the assignment, a firewall controls access to the internal services. Only port 22 (secure shell) was open to the outside, requiring the students to perform the attacks in proper sequence and the use of SSH tunneling to access the internal web service.

Deployment makes heavy use of virtualization, requiring only two physical servers: a well-secured Linux machine with two network interfaces operating as the firewall; and a virtual machine host, which hosts a single virtual web server and one virtual login server per student. The login servers are produced from a master copy with a duplication script and are identical, except for (i) different SSH keys, generated by the guest OS; and (ii) different login credentials files for the web service, generated by the duplication script. See Figure 1 for a diagram of the exercise network's architecture.

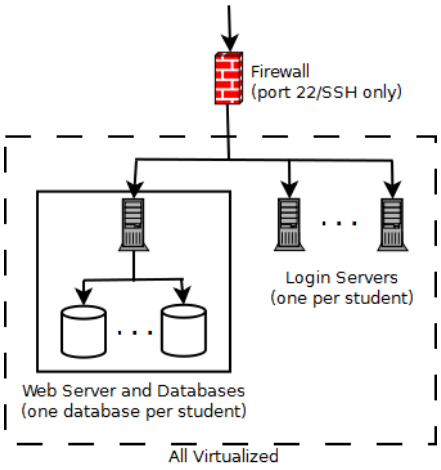

Figure 1: The Blunderdome architecture

By design, the assignment is highly linear: a precondition of each attack is a postcondition of the preceding attack. Furthermore, each individual stage includes a grading requirement allowing progress to be tracked (and, if necessary, partial credit to be assigned). The remainder of this section describes in detail each of the three stages of the assignment, summarized in Figure 2.

3.2 Network Attack

We used an Ubuntu 7.10 installation with an OpenSSH server installed and no additional patches applied for the login server, which hosted the first stage of the assignment. Ubuntu 7.10 shipped with a vulnerable version of OpenSSL. The random generator was inadvertently compromised by Debian package maintainers, and as a result it was only capable of generating 32,767 different key pairs [2]. The students were aware that the systems were running an unpatched version of Ubuntu 7.10, but were not initially provided details about this vulnerability, which they were expected to discover through their own research.

Given the very narrow key pair search space, students were expected to be able to determine a system's private key using the available public key. Students took several different approaches to this problem. Some attempted to generate all possible key pairs until they found the private key matching the public key to their systems. Others found online resources that provided all possible key pairs, and searched the content of the public keys to find a match. The simplest approach students found was to do a web search for the public key's fingerprint, which would return results with the key pair. Using the user-

Stage	Precondition	Attack	Proof
Gain remote user access	SSH public key available (given)	Break weak public key	Create a user-owned text file
Gain root access	User-level access	Execute `vmsplice` privilege escalation	Create root-owned text file
Change grade	Address and credentials for web service	Execute SQL injection	Altered grade in database

Figure 2: Precondition, attack, and grading requirements for each stage of the Blunderdome assignment

name appended to the public key and the private key derived through the methods above, students were able to log in to their target system as a non-administrative user.

3.3 Systems Attack

Once students had non-administrative access to their target system, they were expected to read a file that was owned by the root user without public read access. This file contained a username and password that the user would need for the next phase of the project. Once again, students had to research the system that they were attacking.

The Linux kernel that shipped with Ubuntu 7.10 was vulnerable to a kernel level exploit in the vmsplice system call [3]. Students were able to download C code and compile it for the target system. When executed, the software would exploit the vmsplice vulnerability, and present the user with a root shell. The user was then able to read the username and password out of the specified file to continue to the next phase of the project.

3.4 Web Service Attack

The last phase of the project used a custom-built web service that would allow students to check their fictional "grades." The site prompted each user to enter his username, and the web server would return the user's grade in the class. Students were instructed to change their grade from an F to an A. The web service was susceptible to a fairly simple SQL injection, which allowed students to make the change. As each student's login credentials result in the web service's accessing a different database, students could only modify their own grades, not the grades of other students in the class.

4 Deployments

4.1 Graduate Course

The Blunderdome project was first deployed to a graduate level course titled Information System Security Engineering. Students were given one week to complete the three phases of the project. Students were encouraged to discuss the project with others in the class, but each student was required to perform the exploits himself.

The students most experienced with a Linux environment were able to complete the assignment within two days of the assignment. Most of the students who struggled were unfamiliar with SSH public key authentication, so they were provided with a guide for using keys with PuTTY. Once the students understood how to use public key authentication, most of them completed the remainder of the assignment with relative ease.

To put the assignment in context, the course is concerned with methodologies for security engineering and elements of secure software development. The course engaged case studies of well known endeavors in secure system design to present core principles security engineering. A study of the design of video game consoles (e.g. the Xbox) focused attention on the development of security requirements and architecture. A review of seminal writings on the Multics operating system shone the spotlight on principles of secure design.

With respect to secure software development, students were exposed to a variety of network and system flaws, attack vectors, and corresponding mitigative best practices in programming. Treatment of this portion of the course encompassed a range of vulnerabilities and exploits, spanning system domains and levels. For example, coverage was given to buffer overflow attacks, weaknesses in cryptographic protocols, and flaws in web applications (intentionally the three classes of attacks in the Blunderdome exercise). The educational target here is twofold: (i) make students aware of common weaknesses in system design and implementation, and (ii) catalyze an internal dialogue on best practices of secure software

programming and maintenance as part of the the system development life cycle.

Lessons Learned

Lessons from the Blunderdome experience in its graduate course deployment can be framed in terms of the course's students and in terms of the exercise itself. On the part of the students, the exercise served primarily to cement or emphasize the coursework's importance and relevance. As for the Blunderdome exercise itself, three major lessons are discussed: how specific difficulties that students experienced reflect upon the university's curriculum, how the course and exercise could have been better integrated with each other, and concerns about a lack of student buy-in to the exercise.

Students. The Blunderdome exercise challenged students to accomplish three sequential exploits with four key messages, italicized below. Investigations required in the first step of the exercise attack chain revealed to students some of the *frailties in software development and distribution processes*. The privilege escalation exploit challenged students to find, acquire and launch exploit code for a documented vulnerability, which proved for most to be a frighteningly simple task. A key takeaway from this was *the importance of regular patching and updates of system software*. The last link in the attack chain – the SQL injection attack – reinforced input validation as best practice in secure coding, and in a broader context *the value of a mature software development process that respects security as a fundamental system property*. Ultimately, the exercise demonstrated *the potential for vulnerabilities at all layers of the information system stack*; the main lesson to the students was visible, hands-on reinforcement of the importance of design issues taught in the course.

Blunderdome. Perhaps the most striking lesson of the Blunderdome exercise was revealed by the nature of the students' difficulties in completing it. Two aspects of the exercise caused a disproportionately large number of students to ask for assistance: configuring secure shell authentication using keys, and performing SSH tunneling. After the basic concepts were explained or references to instructional material regarding those two techniques were provided, those same students had little to no problems executing the rest of the exercise.

This calls to mind the position espoused, among others, by Dornseif, *et al.*, (who also advocate teaching using offensive exercises) that the focus on "long-term methodical knowledge instead of short-term system knowledge" results in gaps in practical knowledge among students [4]. This view holds with the Blunderdome experience, in which the architectural, theoretical, and security-oriented aspects of the exercise (breaking the keys, deploying the buffer overflow exploit, and writing an SQL injection) were met with less difficulty than what might be considered the technical minutae of the particular systems in use.

Though the Blunderdome exercise was devised and deployed as a practical final course project, this was probably not the most suitable way to integrate it into the course. A key potential benefit of the exercise is to generate in-class discussion regarding the causes and possible remediation of design issues resulting in the deployed vulnerabilities; as a final project these opportunities were extremely limited to the three lecture sessions remaining at the end of the term.

Furthermore, because of the staged and linear nature of the exercise (which we still regard as an advantage from an educational standpoint), it or a richer variation thereof would have been very well suited to a much longer period of time, with in-class discussion and lectures associated with issues relating to each stage of the attack occuring as each stage is assigned and completed. Instead, as it was deployed, much of the exercise took place in something of a curricular vacuum.

Perhaps because of the apparent isolation of the exercise, we were concerned and somewhat disappointed with what we perceived as a lack of "buy-in" from the students. Several seemed unclear as to the way that the project fit in with the coursework so far. For this reason, a significant effort should be made to ensure tight integration of the exercise with the course – we regard this as the most important practical lesson about future deployments.

4.2 High School Interns

During the summer semester the University of Tulsa offers high school students the opportunity to gain exposure to university level academic research. The summer after deploying Blunderdome for graduate students it was deployed for the visiting high school students. Most of the students had very little experience with most of the technologies involved in Blunderdome, having had little formal computer science education except for perhaps one or two semesters of computer programming high school coursework.

The interns were given an assignment sheet similar to that which was provided for the graduate course, but were given as much time as necessary to complete the task. For many of the high school students Blunderdome was their first exposure to Linux and asymmetric key authentication, let alone the vulnerabilities they were charged with exploiting. For this reason, the high school students were given a greater degree of guidance than the graduate students were; in fact, a graduate student supervised their progress directly during the duration of their

on-site work.

There were several goals in deploying the exercise to the interns: to demonstrate the nature and importance of the security research they would be doing; to provide a realistic exercise that would hopefully disillusion them about the nature of security and hacking (i.e. demonstrate that computer security is more about understanding the systems involved than dramatic hacking scenarios); to gauge their level of technical expertise in order to inform their assignment to projects; and to provide a unified scenario to motivate and teach the basic technical principles involved.

Lessons Learned

The lessons learned are again separated into those successfully imparted to the students and those regarding the Blunderdome exercise itself. Many of the goals for student education were achieved quite successfully. We also learned important lessons about the level of technical ability that can be assumed for high school students, and the usefulness of an interesting motivating exercise for teaching basic systems and network principles.

Students. The exercise exposed them to a Linux command line. They learned how to use key pairs to securely log into a remote system. They learned about POSIX access control matrices and how one user could prevent another user from gaining access to a particular file. Particularly successful were the aspects that required compiling the exploit from source, which allowed us to teach the GNU toolchain; scripting the breaking of the key pair; and the requirement to tunnel traffic through SSH, which enabled a fairly comprehensive discussion of computer network fundamentals.

At the same time they learned about the vulnerabilities exploited by Blunderdome. Like the graduate students, they were exposed to the risks of not keeping a system's software up to date. They gained the experience of investigating known weaknesses in software, and seeing how easily those weaknesses can be exploited. While researching the causes of the vulnerabilities the interns were able to see the importance of following good programming practices, in particular the lapses that made Blunderdome's exploits possible. They learned the vocabulary used by security professionals and a basic taxonomy of the exploits.

Blunderdome. One lesson was that, although it would be a mistake to underestimate the capacity for the students to learn lessons typically taught in graduate computer security classes, there was a significant need for a mentor to be available continuously in order to teach the principles involved. Given an experienced teacher to assist them, the Blunderdome exercise can be walked through over the course of a fairly intensive week of work, during which the mentor must be available to explain the background knowledge virtually from scratch.

Perhaps the most powerful takeaway from the high school students' experiences with the Blunderdome is related to the problem mentioned above with the graduate students' lack of "buy-in" to the exercise. This problem did not exist for the interns. In fact, the very fact that the students bought into the exercise so thoroughly is what permitted its success. Several of the high school students, though they were not encouraged to do so, worked on the Blunderdome exercise from home. Even though they made little progress on the assignment proper, we discovered that they returned the next day with notably improved background knowledge. The power of a motivating exercise to drive this sort of eagerness should not be underestimated. In fact, this was the most valuable aspect of Blunderdome in the high schooler deployment.

5 Conclusions

5.1 Offensive Exercises in a Security Curriculum

Although the precise role of offensive exercises in a cyber security education program remains somewhat controversial [6, 7, 15], they have become commonplace in cyber defense competitions and information assurance curricula [8, 10, 12, 13] and are a method of measuring or comparing the health and rigor of educational programs. The benefit of concrete exposure to vulnerabilities and exploits in these classes may spark the imagination and raise the level of enthusiasm of students. The integration of ethics, law and policy is an important counter balance to incorporating these kinds of exercises early on in an information assurance curriculum.

Educational objectives of including offensive exercises in specialty or elective information assurance classes are more targeted, and typically require extending the exercises to meet those specific targets. For courses in security audit and penetration testing, the objective may be exposure to methodologies and processes or to gain proficiency with certain exploitation techniques and tools. In either case, an infrastructure similar to Blunderdome may be used to create a rich environment in which to master requisite skills and concepts.

In courses that stress security operations, a Blunderdome-like framework tied to a virtual enterprise network can be used to underscore and implement management principles of monitoring, patching and incident handling. Courses that emphasize secure coding or security engineering also may benefit from the inclusion of offensive exercises. An offensive exercise may illuminate architectural weaknesses of an engineered security solution or provide a launching

point for discussion of secure coding practices. Used in this way, offensive exercises should be augmented with task elements that challenge the student to remediate or re-engineer flawed systems and code. Moreover, they offer valuable insight into an adversary's frame of mind for system engineers and developers.

Finally, offensive exercises are, to many students, simply more *fun* than other practical technical assignments. Although they must always be tempered with an ethical emphasis, the benefits of capturing students' attention (particularly younger students) with an exercise carrying the perception of drama or excitement in order to teach important technical content should not be overlooked.

5.2 Future Work

The immediate future of Blunderdome lies in further integration with our information assurance curriculum. Primarily, Blunderdome will be revised to better match the material presented in the Information System Security Engineering course and to include both more recently discovered vulnerabilities and more recent software. In keeping the exercise current, we hope to increase student interest in the assignment—there is a significant difference in compromising three-year-old software and compromising software that the student might have used less than a year ago.

In addition to being revised for Information System Security Engineering, Blunderdome may also form the foundation of the new semester project for Secure Electronic Commerce. Secure Electronic Commerce has long featured a semester-long practical exercise focused on internet banking in which was offensive strategies such as brute-forcing PINs and phishing for encryption keys were allowed, but not required. The exercise is supported by a monolithic, custom-written server that simulates bank and trading operations and supports a linear progression of assignments which implement increasingly advanced forms of authentication to protect student transactions. Basing the Secure Electronic Commerce exercise on Blunderdome would not only give it an increased sense of verisimilitude and practical worth by using common software, but also allow for a better contained and more nuanced simulation environment.

Finally, we are interested in expanding the Blunderdome exercise into the realm of exploiting server misconfiguration. While software vulnerabilities pose significant risks to organizations' security stances, misconfiguration is often the mistake that lies at the heart of real life security breaches [1], and a practical offensive exercise would complement our courses which focus on secure system administration and enterprise security management.

6 Acknowledgments

This material is based on research sponsored by DARPA under agreement number FA8750-09-1-0208. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, or DARPA or the U.S. Government.

References

[1] BAKER, W., HUTTON, A., HYLENDER, C., NOVAK, C., PORTER, C., SARTIN, B., TIPPETT, P., AND VALENTINE, J. 2009 data breach investigations report. Tech. rep., Verizon Business RISK Team, 2009.

[2] CVE-2008-0166. National Vulnerability Database, January 2008. [online] `http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2008-0166`.

[3] CVE-2008-0600. National Vulnerability Database, February 2008. [online] `http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2008-0600`.

[4] DORNSEIF, M., GAERTNER, F., MINK, M., AND PIMENIDIS, L. Teaching Data Security at University Degree Level. In *N. Milovslaskaya & H. Armstrong (Eds.), Proceedings of the IFIP TC11 WG 11.8, Fourth World Conference Information Security Education (WISE4* (2005), pp. 213–222.

[5] GUPTA, S. Foundstone hacme bank v2.0 software security training application user and solution guide. Tech. rep., Foundstone Inc., 2006.

[6] HARRIS, J. Maintaining ethical standards for a computer security curriculum. In *InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development* (New York, NY, USA, 2004), ACM, pp. 46–48.

[7] LOGAN, P. Y., AND CLARKSON, A. Teaching students to hack: curriculum issues in information security. In *SIGCSE '05: Proceedings of the 36th SIGCSE technical symposium on Computer science education* (New York, NY, USA, 2005), ACM, pp. 157–161.

[8] O'LEARY, M. A laboratory based capstone course in computer security for undergraduates. *ACM SIGCSE Bulletin 38*, 1 (2006), 6.

[9] OPEN WEB APPLICATION SECURITY PROJECT. Owasp webgoat project. Web site, May 2010. [online] http://www.owasp.org/index.php/Category:OWASP_WebGoat_Project.

[10] SCHEPENS, W., RAGSDALE, D., SURDU, J., AND SCHAFER, J. The Cyber Defense Exercise: An evaluation of the effectiveness of information assurance education. *The Journal of Information Security 1*, 2 (2002).

[11] VIGNA, G. Teaching hands-on network security: Testbeds and live exercises. *Journal of Information Warfare 3*, 2 (2003), 8–25.

[12] WAGNER, P., AND WUDI, J. Designing and implementing a cyberwar laboratory exercise for a computer security course. *ACM SIGCSE Bulletin 36*, 1 (2004), 402–406.

[13] WALDEN, J. A real-time information warfare exercise on a virtual network. In *Proceedings of the 36th SIGCSE technical symposium on Computer science education* (2005), ACM, p. 90.

[14] WHITE, G., AND WILLIAMS, D. The collegiate cyber defense competition. In *Proceedings of the 9th Colloquium for Information Systems Security Education* (2005).

[15] YURCIK, W., AND DOSS, D. Different approaches in the teaching of information systems security. In *Proceedings of the Information Systems Education Conference* (2001).