Penetration test

A **penetration test**, or the short form pentest, is an attack on a computer system with the intention of finding security weaknesses, potentially gaining access to it, its functionality and data[1] [2]

The process involves identifying the target systems and the goal, then reviewing the information available and undertaking available means to attain the goal. A penetration test target may be a white box (where all background and system information is provided) or black box (where only basic or no information is provided except the company name). A penetration test can help determine whether a system is vulnerable to attack, if the defenses were sufficient and which defenses (if any) were defeated in the penetration test.[3]

A penetration can be likened to surveying a rabbit proof fence, which must be whole to keep the rabbits out. In surveying the fence the penetration tester may identify a single hole large enough for a rabbit (or themselves) to move through, once the defense is passed, any further review of that defense may not occur as the penetration tester moves on to the next security control. This means there may be several holes or vulnerabilities in the first line of defense and the penetration tester only identified the first one found as it was a successful exploit. This is where the difference lies between a vulnerability assessment and penetration test - the vulnerability assessment is everything that you may be susceptible to, the penetration test is based on if your defense can be defeated.

Security issues uncovered through the penetration test are presented to the system's owner. Effective penetration tests will couple this information with an accurate assessment of the potential impacts to the organization and outline a range of technical and procedural countermeasures to reduce risks.

Penetration tests are valuable for several reasons:

1. Determining the feasibility of a particular set of attack vectors
2. Identifying higher-risk vulnerabilities that result from a combination of lower-risk vulnerabilities exploited in a particular sequence
3. Identifying vulnerabilities that may be difficult or impossible to detect with automated network or application vulnerability scanning software
4. Assessing the magnitude of potential business and operational impacts of successful attacks
5. Testing the ability of network defenders to successfully detect and respond to the attacks
6. Providing evidence to support increased investments in security personnel and technology

Penetration tests are a component of a full security audit. For example, the Payment Card Industry Data Security Standard (PCI DSS), and security and auditing standard, requires both quarterly and ongoing penetration testing (after system changes).[4]

1 History

By the mid 1960s, the growing popularity of online timesharing computer systems, which had made their resources accessible to users over communications lines, had created new concerns about system security. As the scholars Deborah Russell and G. T. Gangemi, Sr. explain, "the 1960s marked the true beginning of the age of computer security."[5] In June 1965, for example, several of the country's leading computer security experts held one of the first major conferences on system security, one that was hosted by the government contractor, the System Development Corporation (SDC). During the conference, it was noted that one SDC employee had been able to easily undermine the various system safeguards that had been added to SDC's AN/FSQ-32 timesharing computer system. In the hopes that the further study of system security could be useful, the attendees requested "studies to be conducted in such areas as breaking

security protection in the time-shared system." In other words, the conference participants initiated one of the first formal requests to use computer penetration as tool for studying system security.[6]

At the Spring 1967 Joint Computer Conference, many of the country's leading computer specialists met again to discuss their concerns about system security. During this conference, the computer security experts Willis Ware, Harold Petersen, and Rein Tern, all of the RAND Corporation, and Bernard Peters of the National Security Agency (NSA), all used the phrase "penetration" to describe an attack against a computer system. In a paper, Ware referred to the military's remotely accessible time-sharing systems, warning that "deliberate attempts to penetrate such computer systems must be anticipated." His colleagues Petersen and Turn shared the same concerns, observing that on-line communication systems "are vulnerable to threats to privacy," including "deliberate penetration". Bernard Peters of the NSA made the same point, insisting that computer input and output "could provide large amounts of information to a penetrating program." During the conference, computer penetration would become formally identified as a major threat to online computer systems.[7]

The threat posed by computer penetration was next outlined in a major report organized by the United States Department of Defense (DoD) in late 1967. Essentially, DoD officials turned to Willis Ware to lead a task force of experts from NSA, CIA, DoD, academia, and industry to formally assess the security of time-sharing computer systems. By relying on many of the papers that had been presented during the Spring 1967 Joint Computer Conference, the task force largely confirmed the threat to system security posed by computer penetration. Although Ware's report was initially classified, many of the country's leading computer experts quickly identified the study as the definitive document on computer security.[7] Jeffrey R. Yost of the Charles Babbage Institute has more recently described the Ware report as "by far the most important and thorough study on technical and operational issues regarding secure computing systems of its time period."[8] In effect, the Ware report reaffirmed the major threat posed by computer penetration to the new online time-sharing computer systems.

To get a better understanding of system weaknesses, the federal government and its contractors soon began organizing teams of penetrators, known as tiger teams, to use computer penetration as a means for testing system security. Deborah Russell and G. T. Gangemi, Sr. stated that during the 1970s extquotedbl'tiger teams' first emerged on the computer scene. Tiger teams were government and industry sponsored teams of crackers who attempted to break down the defenses of computer systems in an effort to uncover, and eventually patch, security holes."[9] One of the leading scholars on the history of computer security, Donald MacKenzie, similarly points out that "RAND had done some penetration studies (experiments in circumventing computer security controls) of early time-sharing systems on behalf of the government."[10] Jeffrey R. Yost of the Charles Babbage Institute, in his own work on the history of computer security, also acknowledges that both the RAND Corporation and the SDC had "engaged in some of the first so-called 'penetration studies' to try to infiltrate time-sharing systems in order to test their vulnerability."[11] In virtually all of these early studies, the tiger teams would succeed in breaking into their targeted computer systems, as the country's time-sharing systems had very poor defenses.

Of the earliest tiger team actions, the efforts at the RAND Corporation demonstrated the usefulness of penetration as a tool for assessing system security. At the time, one RAND analyst noted that the tests had "demonstrated the practicality of system-penetration as a tool for evaluating the effectiveness and adequacy of implemented data security safe-guards." In addition, a number of the RAND analysts insisted that the penetration test exercises all offered several benefits that justified its continued use. As they noted in one paper, "a penetrator seems to develop a diabolical frame of mind in his search for operating system weaknesses and incompleteness, which is difficult to emulate." For these reasons and others, many analysts at RAND recommended the continued study of penetration techniques for their usefulness in assessing system security.[12]

Perhaps the leading computer penetration expert during these formative years was James P. Anderson, who had worked with the NSA, RAND, and other government agencies to study system security. In early 1971, the U.S. Air Force contracted with Anderson's private company to study the security of its time-sharing system at the Pentagon. In his study, Anderson outlined a number of the major factors that were involved in computer penetration. The general attack sequence, as Anderson described it, involved a number of steps, including: "1. Find an exploitable vulnerability. 2. Design an attack around it. 3. Test the attack. 4. Seize a line in use... 5. Enter the attack. 6. Exploit the entry for information recovery."

Over time, Anderson's description of the general steps involved in computer penetration would help guide many other security experts, as they continued to rely on this technique to assess the security of time-sharing computer systems.[12]

In the following years, the use of computer penetration as a tool for security assessment would only become more refined and sophisticated. In the early 1980s, the journalist William Broad briefly summarized the ongoing efforts of tiger teams to assess system security. As Broad reported, the DoD-sponsored report by Willis Ware had "showed how spies could actively penetrate computers, steal or copy electronic files and subvert the devices that normally guard top-secret information. The study touched off more than a decade of quiet activity by elite groups of computer scientists working for the Government who tried to break into sensitive computers. They succeeded in every attempt."[13] While these various studies may have suggested that computer security in the U.S. remained a major problem, the scholar Edward Hunt has more recently made a broader point about the extensive study of computer penetration as a security tool. As Hunt suggests in a recent paper on the history of penetration testing, the defense establishment ultimately "created many of the tools used in modern day cyberwarfare," as it carefully defined and researched the many ways in which computer penetrators could hack into targeted systems.[14]

2 Standards and certification

The Information Assurance Certification Review Board (IACRB) manages a penetration testing certification known as the Certified Penetration Tester (CPT). The CPT requires that the exam candidate pass a traditional multiple choice exam, as well as pass a practical exam that requires the candidate to perform a penetration test against servers in a virtual machine environment.[15]

3 Tools

3.1 Specialized OS distributions

There are several operating system distributions, which are geared towards performing penetration testing.[16] Distributions typically contains pre-packaged and preconfigured set of tools. This is useful because the penetration tester does not have to hunt down a tool when it is required. This may in turn lead to further complications such as compile errors, dependencies issues, configuration errors, or simply acquiring additional tools may not be practical in the tester's context.

Popular examples are Kali Linux (replacing BackTrack as of December 2012) based on Debian Linux, Pentoo based on Gentoo Linux and WHAX based on Slackware Linux.[17][18] There are many other specialized operating systems for penetration testing, each more or less dedicated to a specific field of penetration testing.

3.2 Software frameworks

- Metasploit
- nmap
- w3af

4 Automated testing tools

The process of penetration testing may be simplified as two parts:

- Discovering a combination of legal operations that will let the tester execute an illegal operation: unescaped SQL commands, unchanged salts in source-visible projects, human relationships, using old hash/crypto functions

  A single flaw may not be enough to enable a critically serious exploit. Leveraging multiple known flaws and shaping the payload in a way that will be regarded as valid operation is almost always required. Metasploit provides a ruby library for common tasks and maintains a database of known exploits.

  Under budget and time constraints, fuzzing is a common technique to discover vulnerabilities. What it aims to do is to get an unhandled error through random input. Random

input allows the tester to use less often used code paths. Well-trodden code paths have usually been rid of errors. Errors are useful because they either expose more information, such as HTTP server crashes with full info tracebacks or are directly usable such as buffer overflows. A way to see the practicality of the technique is to imagine a website having 100 text input boxes. A few of them are vulnerable to SQL injections on certain strings. Submitting random strings to those boxes for a while will hopefully hit the bugged code path. The error shows itself as a broken HTML page half rendered because of SQL error. In this case, only text boxes are treated as input streams. But software systems have many possible input streams such as cookie/session data, the uploaded file stream, RPC channels, or the memory. In any of these input streams, errors can happen. The goal is first, to get an unhandled error, and second, come up with a theory on the nature of the flaw based on the failed test case. Then write an automated tool to test the theory until it is correct. After that, with luck it should become obvious how to package the payload so that its execution will be triggered. If this is not viable, one can hope that another error produced by the fuzzer will yield more fruit. The use of a fuzzer means time is not wasted on checking completely adequate code paths where exploits are unlikely to occur.

- Specifying the illegal operation, also known as payloads according to Metasploit terminology: remote mouse controller, webcam peeker, ad popupper, botnet drone or password hash stealer. Refer to Metasploit payload list for more examples.

Some companies maintain large databases of known exploits and provide products to automatically test target systems if they are vulnerable.

- Nessus
- Retina Network Security Scanner
- OpenVAS

5 See also

- BackBox
- BackTrack
- IT risk
- ITHC
- Kali Linux
- Pentoo
- Tiger team

6 Notes

[1] *The CISSP® and CAPCM Prep Guide: Platinum Edition*. John Wiley & Sons. ISBN 978-0-470-00792-1. “A penetration test can determine how a system reacts to an attack, whether or not a system’s defenses can be breached, and what information can be acquired from the system”

[2] Kevin M. Henry. *Penetration Testing: Protecting Networks and Systems*. IT Governance Ltd. ISBN 978-1-849-28371-7. “Penetration testing is the simulation of an attack on a system, network, piece of equipment or other facility, with the objective of proving how vulnerable that system or “target” would be to a real attack.”

[3] “Penetration Testing: Assessing Your Overall Security Before Attackers Do”. SANS Institute. Retrieved 16 January 2014.

[4] Alan Calder and Geraint Williams. *PCI DSS: A Pocket Guide, 3rd Edition*. ISBN 978-1-84928-554-4. “network vulnerability scans at least quarterly and after any significant change in the network”

[5] Russell and Gangemi, Sr. (1991), p. 27

[6] Hunt (2012), pp. 7-8

[7] Hunt (2012), p. 8

[8] Yost (2007), p. 602

[9] Russell and Gangemi, Sr. (1991), p. 29

[10] MacKenzie (2001), p. 156

[11] Yost (2007), pp. 601-602

[12] Hunt (2012), p. 9

[13] Broad, William J. (September 25, 1983). "Computer Security Worries Military Experts", *New York Times*

[14] Hunt (2012), p. 5

[15] "CWAPT - CERTIFIED PENETRATION TESTER". IACRB. Retrieved 17 January 2012.

[16] Faircloth, Jeremy (2011). "1". *Penetration Tester's Open Source Toolkit, Third Edition* (Third ed.). Elsevier. ISBN 1597496278.

[17] *Kali Penetration Testing concepts*. ISBN 978-1-78216-316-9. "The creators of BackTrack have released a new, advanced Penetration Testing Linux distribution named Kali Linux"

[18] *Kali Penetration Testing concepts*. ISBN 978-1-78216-316-9. "Kali Linux is designed to follow the flow of a Penetration Testing service engagement"

7 References

- Hunt, Edward (2012). "US Government Computer Penetration Programs and the Implications for Cyberwar", *IEEE Annals of the History of Computing* 34(3)
- Long, Johnny (2007). *Google Hacking for Penetration Testers*, Elsevier
- MacKenzie, Donald (2001). *Mechanizing Proof: Computing, Risk, and Trust*. The MIT Press
- MacKenzie, Donald and Garrell Pottinger (1997). "Mathematics, Technology, and Trust: Formal Verification, Computer Security, and the U.S. Military", *IEEE Annals of the History of Computing* 19(3)
- McClure, Stuart McClure (2009) *Hacking Exposed: Network Security Secrets and Solutions*, McGraw-Hill
- Russell, Deborah and G. T. Gangemi, Sr. (1991). *Computer Security Basics*. O'Reilly Media
- Yost, Jeffrey R. (2007). extquotedblA History of Computer Security Standards," in *The History of Information Security: A Comprehensive Handbook*, Elsevier

8 External links

- List of Network Penetration Testing software, Mosaic Security Research

9 Text and image sources, contributors, and licenses

9.1 Text

- **Penetration test** *Source:* http://en.wikipedia.org/wiki/Penetration_test?oldid=627735884 *Contributors:* The Anome, Edward, Ronz, Timc, Flyingbird, Everyking, AlistairMcMillan, Ragib, Utcursch, Beland, Special, Giraffedata, Pearle, Walter Görlitz, Atlant, Ransak, Martinship, Caesura, M3tainfo, 2mcm, Randy Johnston, Rzelnik, Rossheth, Bobrayner, Mindmatrix, Al E., AllanBz, NeonMerlin, Aapo Laitinen, Alejos, FlaBot, Adfectio, SusanneOberhauser, Pinecar, Xample, Janet13, Irishguy, TechnoGuyRob, Piesechki, Raistolo, Chris Chittleborough, Palapa, SmackBot, Systemf, S charette, Mauls, Ohnoitsjamie, GoneAwayNowAndRetired, Bluebot, Ian13, Haidut, Martin Blank, Frap, JonHarder, Radagast83, Kukini, Doug Bell, Ckatz, Ehheh, IReceivedDeathThreats, Wjejskenewr, JHP, Heqs, Amalas, CWY2190, MessedRobot, Random name, Mattj2, Mblumber, Kaldosh, Njan, Thijs!bot, JustAGal, AntiVandalBot, Widefox, Dubbon, Bulzeeb, MER-C, Godcast, Ermanon, Frank Kai Fat Chow, JCarlos, Trusilver, Ankit bond2005, Mamyles, Jpubal, C. Foultz, Qu3a, Pjvenda, Philip Trueman, Jedi 001, Securitytester, Softtest123, Whitehatnetizen, Derekcslater, Ml-crest, Sephiroth storm, Pxma, Riya.agarwal, KPH2293, ClueBot, The Thing That Should Not Be, Mild Bill Hiccup, PolarYukon, Ajcblyth, Patricioreyes, Davbradbury, SchreiberBike, Muro Bot, Jinxpuppy, Johnuniq, Jasburger, Shamanchill, SF007, DumZiBoT, XLinkBot, Dan Aquinas, Addbot, PatrickFlaherty, Fieldday-sunday, CanadianLinuxUser, ToddSweeney, MrOllie, Chrismcnab, Shifat2sadi, Unicityd, Killiondude, Cantasta, Aneah, Frosted14, Pradameinhoff, Manuelt15, Nameless23, Page vanda1iser, FrescoBot, HamburgerRadio, Poeticos, Winterst, LittleWink, Infosec23, Daemonaka, Lotje, VernoWhitney, EmausBot, Stevehwiki, Newman12a, Dcirovic, OmidPLuS, SonOfBuzz, Kzl.zawlin, Askedonty, Ivhtbr, Nezzalisio, Tolly4bolly, Mile2, Yashartha Chaturvedi, Ultra1ne, Zabanio, ClueBot NG, Lsamaras11, Martinmr79, Helpful Pixie Bot, BG19bot, Ostendali, Citizen4096, Mayast, IsraelZulu, Richu jose, Mdann52, ChrisGualtieri, Atmega644, Itsmyline, Patrick.bausemer, Mohaab, Aperks1811, Decko12, BurritoBazooka, Maniesansdelire, Kencordero, I am One of Many, Amitkankar, Krankes-kind, Kellyenglish1, K0zka, Rons corner, Tdittes, Ca2james, S166865h, Akifumii and Anonymous: 253

9.2 Images

- **File:Green_bug_and_broom.svg** *Source:* http://upload.wikimedia.org/wikipedia/commons/8/83/Green_bug_and_broom.svg *License:* LGPL *Contributors:* File:Broom icon.svg, file:Green_bug.svg *Original artist:* Poznaniak, pozostali autorzy w plikach źródłowych
- **File:Question_book-new.svg** *Source:* http://upload.wikimedia.org/wikipedia/en/9/99/Question_book-new.svg *License:* ? *Contributors:* ? *Original artist:* ?

9.3 Content license

- Creative Commons Attribution-Share Alike 3.0