Vulnerability Analysis

Contents

Testing

Vulnerability testing is the process of discovering flaws in systems and applications which can be leveraged by an attacker. These flaws can range anywhere from host and service misconfiguration, or insecure application design. Although the process used to look for flaws varies and is highly dependent on the particular component being tested, some key principals apply to the process.

When conducting vulnerability analysis of any type the tester should properly scope the testing for applicable depth and breadth to meet the goals and/or requirements of the desired outcome. Depth values can include such things as the location of an assessment tool, authentication requirements, etc. For example; in some cases it maybe the goal of the test to validate mitigation is in place and working and the vulnerability is not accessible; while in other instances the goal maybe to test every applicable variable with authenticated access in an effort to discover all applicable vulnerabilities. Whatever your scope, the testing should be tailored to meet the depth requirements to reach your goals. Depth of testing should always be validated to ensure the results of the assessment meet the expectation (i.e. did all the machines authenticate, etc.). In addition to depth, breadth must also be taken into consideration when conducting vulnerability testing. Breadth values can include things such as target networks, segments, hosts, application, inventories, etc. At its simplest element, your testing may be to find all the vulnerabilities on a host system; while in other instances you may need to find all the vulnerabilities on hosts with in a given inventory or boundary. Additionally breadth of testing should always be validated to ensure you have met your testing scope (i.e. was every machine in the inventory alive at the time of scanning? If not, why).

Active

Active testing involves direct interaction with the component being tested for security vulnerabilities. This could be low level components such as the TCP stack on a network device, or it could be components higher up on the stack such as the web based interface used to administer such a device. There are two distinct ways to interact with the target component: automated, and manual.

Automated

Automated testing utilizes software to interact with a target, examine responses, and determine whether a vulnerability exists based on those responses. An automated process can help reduce time and labor requirements. For example, while it is simple to connect to a single TCP port on a system to determine whether it is open to receive incoming data, performing this step once for each of the available 65,535 possible ports requires a significant amount of time if done manually. When such a

test must be repeated on multiple network addresses, the time required may simply be too great to allow testing to be completed without some form of automation. Using software to perform these functions allows the tester to accomplish the task at hand, and focus their attention on processing data and performing tasks which are better suited to manual testing.

Network/General Vulnerability Scanners

Port Based

An automated port based scan is generally one of the first steps in a traditional penetration test because it helps obtain a basic overview of what may be available on the target network or host. Port based scanners check to determine whether a port on a remote host is able to receive a connection. Generally, this will involve the protocols which utilize IP (such as TCP, UDP, ICMP, etc.), However, ports on other network protocols could be present as well dependent on the environment (for example, it's quite common in large mainframe environments for SNA to be in use). Typically, a port can have one of two possible states:

```
    Open - the port is able to receive data
    Closed - the port is not able to receive data
```

A scanner may list other states, such as "filtered", if it is unable to accurately determine whether a given port is open or closed.

When the scanner determines that a port is open, a presumption is made by the scanner as to whether a vulnerability is present or not. For example, if a port based scanner connects to TCP port 23, and that port is listening, the scanner is likely to report that the telnet service is available on the remote host, and flag it as having a clear text authentication protocol enabled.

Service Based

A service based vulnerability scanner is one which utilizes specific protocols to communicate with open ports on a remote host, to determine more about the service that is running on that port. This is more precise than a port scan, because it does not rely on the port alone to determine what service is running. For example, a port scan may be able to identify that TCP port 8000 is open on a host, but it will not know based on that information alone what service is running there. A service scanner would attempt to communicate with the port using different protocols. If the service running on port 8000 is able to correctly communicate using HTTP, then it will be identified as a web server.

Banner Grabbing

Banner grabbing is the process of connecting to a specific port and examining data returned from the remote host to identify the service/application bound to that port. Often in the connection process, software will provide an identification string which may include information such as the name of the application, or information about which specific version of the software is running.

Web Application Scanners

General application flaw scanners

Most web application scans start with the address of a website, web application, or web service. The scanner then crawls the site by following links and directory structures. After compiling a list of webpages, resources, services and/or other media offered, the scanner will perform tests, or audits against the results of the crawl. For example, if a webpage discovered in the crawl has form fields, the scanner might attempt SQL injection or cross-site scripting. If the crawled page contained errors, the scanner might look for sensitive information displayed in the error detail, and so on.

It should be noted that crawling and testing phases can be staggered and performed at the same time to reduce overall scanning time. This is the default behavior for many web application scanners.

Directory Listing/Brute Forcing

Suppose there are directories available on the website that the crawler won't find by following links. Without prior knowledge of these directories, provided by the user, the scanner has at least two additional options.

The scanner/crawler can search for "common" directories. These are directories with names and variants of names that are commonly found, and are included in a list that has been compiled as the result of years of experience and scanning. Most web applications have a "built-in" list of this sort, while some penetration testers maintain their own custom lists. Sometimes directory names are unique enough that they can be used to identify a 3rd party web application with reasonably high accuracy. An accurate directory list can often be the key to finding the "administrative" portion of a website - a portion most penetration testers should be highly interested in discovering.

Brute forcing directories is a similar approach, though instead of using a static list, a tool is used to enumerate every possibility a directory name could have. The downside of using this approach is that it has the potential to crash or inundate the web server with requests and thus cause a denial-of-service condition. Care should be taken to perform directory brute forcing while someone is keeping a close watch on the condition of the web server, especially in a production setting.

The reason you as the penetration tester would want to perform directory listing is to extend your attack field or to find directories that could contain sensitive information (which depending on the goal of the penetration test, may lead to a major finding within it).

Web Server Version/Vulnerability Identification

Many web application scanners will attempt to compare the version of the web server with known vulnerable versions in security advisories. This approach can sometimes lead to false positives; as there are some cases where open-source web servers are forked or copied and given new names, banners, and assigned different version numbers. Additional steps should be taken to verify that the web server is, in fact, running what the banner, or web scanner reports.

Methods

Several web server methods are considered insecure, and can allow attackers to gain varying levels of access to web server content. The fact that these methods are part of the web server software, and not web site content differentiates it from other vulnerabilities discussed thus far. Some insecure methods include:

OPTIONS

While the HTTP OPTIONS method is not insecure by itself, it can allow an attacker to easily enumerate the kinds of HTTP methods accepted by the target server. Note, the OPTIONS method is not always accurate and each of the methods below should be validated individually.

PUT/DELETE

Using the PUT method, an attacker can upload malicious content such as HTML pages that could be used to transfer information, alter web content or install malicious software on the web server. Using the DELETE method an attacker could remove content or deface a site causing a disruption of service.

Additionally, modern REST applications use PUT in a different manner:

Create->POST Read->GET Update->PUT Delete->DELETE

WebDAV

WebDAV is a component of the Microsoft Internet Information Server (IIS). WebDAV stands for “Web-based Distributed Authoring and Versioning” and is used for editing and file management. WebDAV extensions are used by administrators to manage and edit Web content remotely on IIS Web servers and can include PROPFIND, COPY, MOVE, PROPPATCH, MKCOL, LOCK, and UNLOCK .WebDAV interacts with core operating system components, which can expose a system to several possible vulnerabilities. Some of these potential risks include:

```
    Buffer overflow conditions due to improper handling of user requests
    Denial-of-service conditions from malformed requests
    Domain based scripting attacks
    Privilege escalation
    Execution of arbitrary code
```

TRACE/TRACK

Modern web servers support the TRACE HTTP method, which contains a flaw that can lead to unauthorized information disclosure. The TRACE method is used to debug web server connections and can allow the client to see what is being received at the other end of the request chain. Enabled by default in all major web servers, a remote attacker may abuse the HTTP TRACE functionality to disclose sensitive information resulting in a loss of confidentiality.

Network Vulnerability Scanners/Specific Protocols

VPN

Conventional vulnerability assessment tools are not capable of performing the correct protocol negotiations with VPN devices that service Internet Key Exchange (IKE). In situations where IKE is in use, it will be necessary to use additional toolkits that can perform functions such as accurate fingerprinting, back off patterns and identify authentication mechanisms that are in use. By identifying these attributes of a VPN device, weaknesses can be identified in running code versions as well as authentication types such as static preshared keys.

Voice Network Scanners

War Dialing

Many organizations still utilize out of band access over telephone lines. Using vulnerability assessment tools that are designed to conduct war-dialing can determine weaknesses in authentication and network architecture.

VoIP

Voice over IP technologies are now abundant within most organizations. Many tools have been developed to conduct vulnerability analysis of VoIP infrastructures. Using these tools, one can identify if VoIP networks are properly segmented and potentials for leveraging these networks to access core infrastructure systems or record phone conversations on a target network may exist.

Manual Direct Connections

As with any automated process or technology, the margin for error always exists. Instabilities in systems, network devices and network connectivity may introduce inaccurate results during testing. It is always recommended to execute manual direct connections to each protocol or service available

on a target system to validate the results of automated testing as well as identifying all potential attack vectors and previously unidentified weaknesses.

Obfuscated

Multiple Exit Nodes

Security monitoring and defense systems operate under the pretense of identifying malicious activity from a specific IP address. In situations where Intrusion Detection systems are deployed and monitoring activity, sourcing assessment and attack activities from multiple IP addresses provide more accurate results and lessen the opportunity for a monitoring device on a target network to identify and respond. Technologies such as TOR proxies can provide a means to conduct assessment activities without sourcing from a single IP address.

IDS Evasion

When conducting assessment activities against a target environment where IDS technologies are deployed, it may be necessary to perform evasion. Using methods such as string manipulation, polymorphism, session splicing, and fragmentation can provide more accurate results while bypassing signature matching patterns implemented in IDS devices.

Passive

Metadata Analysis

Metadata analysis involves looking at data that describes a file, as opposed to the file data itself. A Microsoft Office document for example, might list the document author, company, when the document was last saved, when the document was created, and so on. Many documents even allow for the entry of custom metadata. This could potentially contain internal addresses and paths to servers, internal IP addresses, and other information a penetration tester could use to gain additional access or information.

Though metadata is quite common on documents located on a company's internal network, companies should take care to purge metadata before making documents available to the public, or on the public Internet. For this reason, any metadata an attacker could gain access to passively (without directly attacking the target) should be considered a security issue.

Traffic Monitoring

Traffic monitoring is the concept of connecting to an internal network and capturing data for offline analysis. Route poisoning is excluded from this phase as these create "noise" on the network and can easily be detected. It is often surprising how much sensitive data can be gleaned from a "switched" network. This "leaking of data" onto a switched network can be categorized as follows:

ARP/MAC cache overflow, causing switched packets to be broadcast - this is common on Cisco switches that have improper ARP/MAC cache timing configurations.

Etherleak - some older network drivers and some embedded drivers will use data from system memory to pad ARP packets. If enough ARP packets can be collected, sensitive information from internal memory can be captured

Misconfigured clusters or load balancers

Hubs plugged into the network Note that some of these categories only result in data leakage to a single subnet, while others can result in leakage to much larger network segments.

Validation

Correlation between Tools

When working with multiple tools the need for correlation of findings can become complicated. Correlation can be broken down into two distinct styles, specific and categorical correlation of items, both are useful based on the type of information, metrics and statistics you are trying to gather on a given target.

Specific correlation relates to a specific definable issue such as vulnerability ID, CVE, OSVDB, vendor indexing numbers, known issue with a software product, etc. and can be grouped with micro factors such as hostname, IP, FQDN, MAC Address etc. An example of this would be grouping the findings for host x by CVE number as they would index the same issue in multiple tools.

Categorical correlation relates to a categorical structure for issues such as in compliance frameworks (i.e. NIST SP 800-53, DoD 5300 Series, PCI, HIPPA, OWASP List, etc.) that allow you to group items by macro factors such as vulnerability types, configuration issues, etc. An example of this would be grouping all the findings for hosts with default passwords into a group for password complexity within NIST 800-53 (IA-5).

In most cases penetration testers are going to focus on the micro issues of specific vulnerabilities found in redundancy between multiple tools on the same host. This redundancy can skew the statistical results in the test output leading to a false increased risk profile.

The inverse of this is with an over reduction or simplification in macro correlation (i.e. top 10/20 lists) as the results can skew the output resulting in a false reduced risk profile.

Manual Testing/Protocol Specific

VPN

Fingerprinting

Fingerprinting is useful to determine the type of VPN device and correct version of code released installed. By accurately fingerprinting the device, proper research and analysis can then be conducted against the target system.

Authentication

VPN devices can operate with various forms of authentication. Using VPN toolkits that are not part of conventional vulnerability assessment tools allow for proper identification of the authentication mechanisms and determine weaknesses that may exist such as pre-shared keys or default group IDs.

Citrix

Enumeration

Many default installations and poorly configured Citrix appliances provide a means to enumerate published applications and determine valid usernames that are configured to authenticate to the device. This information becomes crucial during brute force attacks and attempts to break out of

predefined profiles for authorized users.

DNS

Domain Name Systems can offer an abundance of information to an attacker when they are not properly hardened. Version information allow for proper identification and accurate research analysis. Weaknesses such as zone transfers provide an exhaustive list of additional targets for attack as well as information leakage of potentially sensitive data pertaining to the target organization.

Web

Web services provide a large landscape for an attacker. Unlike most other protocols and services, web services are often found running on multiple ports of a single system. Administrators may focus their hardening on the common ports for web services or published directories and neglect to properly harden additional attributes. Web services should always be reviewed in a manual fashion as automated assessment tools are not capable of identifying most weaknesses in their services.

Mail

Mail servers can provide an abundance of information about a target organization. Using inherent functions in the target device, confirmation of valid accounts can be conducted as well as developing a list of potential usernames for additional attacks on other systems. Vulnerabilities such as mail relaying can be leveraged for additional attacks on the organization such as phishing. Often, mail servers will provide a web interface for remote access that can be targeted in brute force campaigns.

Attack Avenues

Creation of attack trees

During a security assessment, it is crucial to the accuracy of the final report to develop an attack tree as testing progresses throughout the engagement. As new systems, services and potential vulnerabilities are identified; an attack tree should be developed and regularly updated. This is especially important during the exploitation phases of the engagement as one point of entry that materializes could be repeated across other vectors mapped out during the development of the attack tree.

Isolated Lab Testing

The accuracy of vulnerability analysis and exploitation is substantially greater when replicated environments are setup in an isolated lab. Often times, systems may be hardened with specific control sets or additional protection mechanisms. By designing a lab that mimics that of the target organization, the consultant can ensure that the vulnerabilities identified and exploits attempted against the desired targets are reliable and lessen the opportunity for inaccurate results or system inoperability.

Visual Confirmation

Manual Connection with Review

While proper correlation can help reduce false findings and increase overall accuracy, there is no substitute for visually inspecting a target system. Assessment tools are designed to review the results of a protocol/service connection or the response and compare to known signatures of vulnerabilities. However, tools are not always accurate in identifying services on uncommon ports or custom logic

that may be built into an application. By manually assessing a target system, its services available and the applications that provide functionality for those services, a tester can ensure that proper validation and vulnerability identification have been completed.

Research

Public Research

Once a vulnerability has been reported in a target system, it is necessary to determine the accuracy of the identification of the issue, and to research the potential exploitability of the vulnerability within the scope of the penetration test. In many cases, the vulnerability will be a reported software vulnerability in a commercial or open source software package, and in other cases the vulnerability can be a flaw in a business process, or a common administrative error like misconfiguration or default password usage.

Vulnerability Databases

Vulnerability databases can be used to verify an issue reported by an automated tool, or to manually review the vulnerability of a target application. Most tools will use the CVE identifier for a given vulnerability, which can be used to access the summary information and links to other sources in the CVE database. The CVE can also be used to search for the issue in vulnerability databases like OSVDB and Bugtraq, or in exploit databases and frameworks.

Vulnerability databases should be used to verify the accuracy of a reported issue. For example, an Apache web server flaw can exist on Windows, but not on Linux, which may not be taken into account by an automated scanner.

Vendor Advisories

Vendor-issued security advisories and change logs can provide pointers to vulnerability information that may not be reported by any automated tools. Many major software vendors report limited details on internally discovered issues and issues where an independent researcher coordinates the disclosure of a vulnerability. If the researcher chooses to remain silent on the details of the vulnerability, the vendor advisory is frequently the only data available. In these cases, other researchers may discover more details independently, and add the details to vulnerability databases. Searching for the CVE used in a vendor advisory may turn up more detail on a potentially exploitable issue.

Change logs can provide guidance for additional research, especially in open source products, where a diff between versions can reveal a vulnerability which was fixed but not widely known, and perhaps not prioritized for upgrade or installation as a result.

Exploit Databases and Framework Modules

Many exploit databases are actively maintained and publicly accessible on the Internet. Security researchers and exploit writers do not always submit their exploit code to multiple sites, so it is advisable to become familiar with several sites, and check each one for exploit code to use against potentially vulnerable applications. While some vulnerability databases track exploit availability, their coverage is usually incomplete and should not be considered exhaustive.

Commercial and open source exploit frameworks can also prove useful in researching vulnerabilities. In most cases, available exploit modules are listed on their public web sites, and can be a valuable indication of the exploitability of an issue.

Common/default Passwords

Frequently, administrators and technicians choose weak passwords, never change the default or do not set any password at all. Manuals for most software and hardware can be easily found online, and will provide the default credentials. Internet forums and official vendor mailing lists can provide information on undocumented accounts, commonly-used passwords and frequently misconfigured accounts. Finally, many web sites document default/backdoor passwords and should be checked for every identified system.

Hardening Guides/Common Misconfigurations

One of the primary goals of penetration testing is to simulate the tactics and behavior of an actual attacker. While automated scanning can reduce the time window of a test, no scanner can behave like a human being. Hardening guides can be an invaluable reference for a penetration tester. They not only highlight the weakest parts of a system, but you can gain a sense of the diligence of an administrator by validating how many recommendations have been implemented. During every penetration test, time should be taken to review every major system and its recommended hardening settings, in order to discover vulnerabilities left in place by the administrator.

User forums and mailing lists can provide valuable information about systems and the various issues administrators have in configuring and securing them. A tester should research target systems as if he were installing one himself, and discover where the pain points and probable configuration errors will lie.

Private Research

Setting up a replica environment

Virtualization technologies allow a security researcher to run a wide variety of operating systems and applications, without requiring dedicated hardware. When a target operating system or application has been identified, a virtual machine (VM) environment can be quickly created to mimic the target. The tester can use this VM to explore to configuration parameters and behaviors of the application, without directly connecting to the target.

Testing Configurations

A testing VM lab should contain base images for all common operating systems, including Windows XP, Vista, 7, Server 2003 and Server 2008, Debian, Ubuntu, Red Hat and Mac OS X, where possible. Maintaining separate images for each service pack level will streamline the process of recreating the target's environment. A complete VM library in combination with a VM environment that supports cloning will allow a tester to bring up a new target VM in minutes. Additionally, using a snapshot feature will allow to work more efficiently and to reproduce bugs.

Fuzzing

Fuzzing, or fault injection, is a brute-force technique for finding application flaws by programmatically submitting valid, random or unexpected input to the application. The basic process involves attaching a debugger to the target application, and then running the fuzzing routine against specific areas of input and then analyzing the program state following any crashes. Many fuzzing applications are available, although some testers write their own fuzzers for specific targets.

Identifying potential avenues/vectors

Log in or connect to a target network application to identify commands and other areas of input. If the target is a desktop application that reads files and/or web pages, analyze the accepted file formats for avenues of data input. Some simple tests involve submitting invalid characters, or very long strings of characters to cause a crash. Attach a debugger to analyze the program state in the event of a successful crash.

Disassembly and code analysis

Some programming languages allow for decompilation, and some specific applications are compiled with symbols for debugging. A tester can take advantage of these features to analyze program flow and identify potential vulnerabilities. Source code for open source applications should be analyzed for flaws. Web applications written in PHP share many of the same vulnerabilities, and their source code should be examined as part of any test.
