# Vulnerability Assessments

Mayo Clinic – Clinical Information Security

# Contents

# Introduction

Cyber Security, while not a particularly new concern, presents new challenges broadly and frequently. It is now common understanding that all technology, especially as it grows more and more interconnected, will eventually be a target for threats in the digital space.

For the healthcare industry, some attack scenarios that in the past may have looked improbable enough to be negligible now start to seem increasingly plausible. It is time to take the concerns with patient privacy and safety to a new level, this is a different sphere of threats.

Through the prism of cyber security shines a number of vectors that enable unauthorized individuals and malicious software to get alarmingly "close" to medical devices and patient data. In Mayo Clinic's interpretation, the public research on medical device security as well as the Mayo's own assessments reveal that many of the systems handling this data and touching these patients do not offer sufficient protection.

While it is beyond the objectives of this document to discuss what factors may have contributed to the unsatisfying defendability of systems in clinical environments, some particular challenges are definitely worth noting.

One of the challenges that the industry faces is the length of the release cycles typically used by healthcare technology vendors, coupled with the slower-paced adoption of updates and overall security improvements. The critical nature of these systems encourages that changes to them are kept to a minimum and subject to thorough regression tests, in order to properly address the catastrophic risk of breaking important functionality or compatibility. This practice seems nearly incongruent with the evolution of cyber security, known to inspire immediate reaction to threat discovery and recommend keeping up to date with state-of-the-art defenses.

Analogous to the above, on the provider side of things, the same logic will often delay some systems from being refreshed even when an upgrade is available. Furthermore, financial aspects also play an important role here, once business justifications may give systems a prolonged life based on the resources invested in them (and, of course, the resources required to replace them).

These challenges, however, must not be taken as excuses for unfavorable security, but as motivators to strive for better technology and processes instead. It is under these premises that Mayo Clinic supports the execution of security assessments all across the industry of healthcare technology as well as for any systems that otherwise support and participate on the clinical environment.

## Objectives

The main goal of a security assessment is to help vendors and customers identify and remediate vulnerabilities in the product analyzed. Such assessments present a great opportunity to deep dive into one system at a time and look at it isolated from any particular network and unaware of any specific configuration. This environment-agnostic perspective helps elucidate problems that may affect any and all customers, and this prevalence may not only aid in prioritizing issue resolution but also often hints at how deeply rooted into the product a given vulnerability may be, which in turn helps in designing more effective fixes.

Mayo Clinic has singular and extensive experience in conducting and consuming vulnerability assessments by leveraging technical expertise, vendor cooperation and internal remediation coordination to their best. Through the lifecycle of an assessment, from its planning to the last steps of vulnerability management, Mayo Clinic has developed best practices that help gather, structure and communicate information so that solutions can be devised and acted upon, both from vendor and provider sides.

The remainder of this document describes the main concepts and methodology that Clinical Information Security has applied in its vulnerability assessments. It is expected that a reader employing the same processes will achieve equally adequate results, as long as resources of similar excellence are utilized. Vendors partnering with Mayo Clinic will be able to command assessments whose reports will contain the high quality information that we feel is best fit for properly and thoroughly addressing today's Cyber Security concerns.

# Assessment Preparation

Before the testing begins, it is crucial that all stakeholders are in agreement as to what kind of experiments are to be executed, on which scenarios, and what is required for their execution to happen. At this point a threat modeling discussion may be helpful and if this phase is carried out carefully, all access and information will be made available to the assessors from the start, avoiding misunderstandings and rework, and paving the way for a successful vulnerability assessment.

## Access and Information Provisioning

One of the decisions that must be made when planning a security assessment is how much access and information are to be conceded to the security tester at the beginning of the assessment. The options range from a “black box” type of testing, where basically no information is given (besides, maybe, the IP address of the target in the network), to a “white box” assessment, where a wealth of internal and confidential knowledge, source code included, may be provided. Between these extremes there are many shades of gray.

One of the driving factors behind this choice is what assets and artifacts actually can be made available to your assessment team. Under Mayo Clinic's initiative of assessing third party products, for example, source code (which is Intellectual Property belonging to the manufacturer) typically is not available and, therefore, a full white box kind of assessment is out of question. Still on negotiating access and information, another input to be considered is the result of your initial threat modeling (more on this in the next section); in particular, what kind of attacker profiles and scenarios are of greatest concerns. For Clinical Information Security at Mayo Clinic, one of the main threat actor profiles of concern is that of a very resourceful attacker with unrestricted physical (if applicable) access to the system/device for unlimited research time. This attacker invests considerable time into reverse engineering and analyzing his targets so that he can later proceed to reliably exploit production units in the wild.

In this case, then, we must opt for a gray box assessment where unrestricted file system access and administrator-level credentials are given to the assessor. This kind of assessment is essentially a simulation of the attacker's capabilities and methods, and access to the file system is something that our hypothetical attacker almost always has access to, even if it means extracting the system hard drives or imaging firmware out of on-board flash chips. Similarly, this threat actor also can login to the system as administrator at any time during his research, given this level of access to data and/or other methods made possible by leveraging physical manipulation of the system. Tangible access to the device is almost always provided to the assessor, except when it is not relevant (e.g. when the target is a pure software solution) or due to logistic challenges (e.g. when an oversized gantry would need to be shipped to the test location). Source code that is not public or not present in the file system of a production deployment might not be given to the assessor, since it is not something that the threat actors in our models would typically have access to.

Another type of asset that is employed in our security assessments is access to (sometimes internal) development documentation and Q&A sessions with engineers of the solution being analyzed. This usually starts with one briefing session just before the assessment begins, when the vendor's R&D staff will give an introduction to the system, its usage, features and architecture. While this is not something that attackers normally have at their disposal, we make use of it as means to help compensate for the resource imbalance: while security assessments are somewhat limited in time and manpower, sophisticated adversaries may be able to make virtually unlimited investments to reach their goals – especially when accounting for the collection of their independent efforts.

As one last point, it is worth noting that the scope of assets and information provided for an assessment very strongly influence the methodology (i.e. the 'color of the box'), and that each methodology has its utility because it tests how the security of a system fares against a unique kind of attacker/scenario. While Clinical Information Security feels that the gray box assessment described in this document provides the methodology desired to simulate the threat sources participating in our threat model (see next section), vendors are greatly encouraged to perform multiple kinds of assessments as part of their overarching security efforts, including those that Mayo Clinic often cannot perform, such as full white box assessments, in order to maximize the knowledge of vulnerabilities and the effectiveness of their product security programs as a whole.

## Threat Modeling

Threat Modeling, for the purpose of assessment preparation, aims at devising which attacks to a given system are relevant to the security assessor and should be focused on. This is important for the scoping and execution of security assessments because it directly influences the selection of tests which the security engineer/researcher will perform. Despite planning at the specificity level of test case determination not being an essential step for scoping vulnerability assessments (in fact, it will often negatively limit the creativity of a talented researcher), it is very useful to somehow document or at least communicate the threat models that are supposed to guide a given exercise, much like business requirements drive an engineering project. Implicitly or not, the work of a security assessor is always emulating some threat model.

There are many varying aspects one may want to capture in a threat model, and a number of these depend on the specific target – you can only draw the complete picture once you understand the components and their vulnerabilities well enough. Despite this document not concerning any system in particular; however, it is still possible and useful to discuss certain elements of the threat models used by Clinical Information Security that are mostly target-independent, such as attacker profiles.

There are also different frameworks to define and convey threat models, and establishing one that works for a given organization may encompass adapting existing standards and/or combining them with other frameworks that supplement them in some way, shape or form. For defining our attacker profiles, for example, we have used much of the taxonomy found in NIST's Special Publication 800-30 Revision 1, particularly where "threat sources" are discussed.

Specifically, when modeling threats for our vulnerability assessments at Mayo Clinic, we are mostly concerned with adversarial threat sources. This excludes or at least deprioritizes most accidental, structural and environmental incidents from our models, allowing the security researchers to focus their tests on attacks that would more likely be intentionally carried out by malicious actors.

As for adversary capability, we contemplate attackers demonstrating all kinds of resourcefulness, but we mainly model for threat sources with 'Very High' capabilities. These are defined by NIST as adversaries that have "a very sophisticated level of expertise, is well-resourced, and can generate opportunities to support multiple successful, continuous, and coordinated attacks". This is one of main reasons why our methodology requires full access to the device, non-public information, and exercises advanced reverse engineering and software and hardware hacking skills. Another important factor regarding capability is that we consider both internal (e.g. trusted, privileged insiders) as well as external threat sources (e.g. an established outsider organization or nation-state). Internal actors are more likely to have the prerequisites to exploit physical and credentialed vulnerabilities, while external adversaries have higher probability of starting their attacks by exploring different avenues.

In terms of intent, our attackers should range from 'Moderate' to 'Very High', where the latter means "the attacker seeks to undermine, severely impede, or destroy a core mission or business function, program, or enterprise by exploiting a presence in the organization's information systems or infrastructure". While for the most part it is expected that attackers to the Mayo Clinic will more often be seeking data such as Protected Health Information and Intellectual Property (i.e. of moderate intent), our systems must also be defendable against those that are after causing financial loss, service disruption and patient harm.

Similarly, on the aspect of targeting, we also model our attackers as ranging from 'Moderate' to 'Very High', where the latter means "the adversary analyzes information obtained via reconnaissance and attacks to target persistently a specific organization, enterprise, program, mission or business function, focusing on specific high-value or mission-critical information, resources, supply flows, or functions; specific employees or positions; supporting infrastructure providers/suppliers; or partnering organizations". On the moderate side, the adversary may be targeting Mayo Clinic because, for example, it is an important player in healthcare (when a campaign focuses on an industry vertical) or because it is a large and recognizable US institution (for national targeting, terrorism, and critical infrastructure). Under more targeted circumstances, the adversary may want to cause harm to Mayo Clinic specifically, going after a certain business unit or even an individual patient, having performed extensive reconnaissance and identified the systems, vulnerabilities and staff to single out for exploitation.

For networked attacks, our models mostly contemplate scenarios where the attackers are already in the same network as the targeted systems. This accounts both for insider threats as well as external threat sources that might manage to breach providers and traverse their networks. Defenses that are not native to the target systems, such as the provider's IPS or DLP solutions, are not considered in the models since they tend to be volatile and deployment-dependent. Our goal is to assess how the system fares under any circumstances.

The profile described above should give enough information to start modeling threats to a specific system by understanding which vulnerabilities fall under the capabilities of our adversaries. By using a realistic model that addresses the current threats to a notable organization such as Mayo Clinic, we avoid oversimplifications like exclusively addressing the more prevalent but less resourceful 'script kiddie' kind of attacker, and focus on more understated security issues and state-of-the-art protections that may defend critical systems from the skilled threat actors we are up against.

Finally, as part of devising threat models for the specific systems being assessed, threat sources may need to be adjusted. For instance, while Clinical Information Security chooses to concentrate on adversarial threats, accidental threats caused by the absence or inadequacy of protection mechanisms in highly accessible devices are also of concern and should be addressed by security assessments. Examples of this include a networked system that crashes upon an accidental (non-malicious) network scan or a patient-operated device that can harmfully malfunction if the network cable is unplugged.

# Methodology

This chapter discusses the methodology applied by Clinical Information Security for vulnerability assessments. It builds upon the experience of assessing several dozens of different systems pertaining to clinical environments and, as such, should serve appropriately for testing other healthcare technology products. Vendors partnering with Mayo Clinic on these exercises should follow the same methodology in order to preserve consistency in the quality of results produced.

## Duration and Manpower

On average, our security assessments are planned for 3 consecutive weeks, where the first 2 weeks are used for the actual testing and the last week is reserved for reporting. It is important that the reporting week immediately follows the testing and that the assessors still maintain the same level of access to the test environment. The immediate report ensures that the security testers document their findings when the details are freshest in their minds, while the access to the test environment allows for capturing last minute evidence and revalidating security issues by answering questions that may only be brought up at this late stage.

In terms of manpower, having a system assessed by two specialized professionals working in cooperation seems to work best on average. Different assessors have different ways of using their skills and approach systems from different angles, maximizing the number of vulnerabilities uncovered, particularly so when able to communicate among themselves.

The skillset required to perform these assessments is, unfortunately, relatively rare and a crucial point for the success of this kind of project. The extent and quality of the results are directly and largely influenced by the experience and knowledge of the individuals performing the assessment. Less experienced and less knowledgeable professionals will often fail to identify or to properly analyze some of the critical vulnerabilities, undermining the confidence in the results produced and making the exercise much less useful overall. Understanding of the importance and singularity of the skillset of good vulnerability researchers, Mayo Clinic has made extensive use of specialized third party assessors, and recommendations of some of the firms offering these abilities will be made later in this document.

Of course, all of the above may need to be adjusted for specific targets and scope. For example, a system with many components deployed across multiple computers will likely present a larger attack surface that will need more time and/or more manpower to be adequately covered. Additionally, skillset can be fine-tuned according to the nature of the target and the technologies involved. The abilities and knowledge of a good security assessor tend to be very broad, covering many platforms and being able to simulate various types of attack, but, for example, if the scope of the object being assessed is comprised simply of a web application, then it makes more sense to enlist assessors with more expertise in security of web applications and the software platforms supporting them, and less so of hardware security or physical security traits.

## Toolset and Equipment

Tooling is a choice that is rather personal to the security testers, and in general it is a good idea to leave it up for them to decide based on what they are most comfortable with. Also, when a contracting a third party, consultants will most often already have everything they need in terms of hardware and software. Yet, there may be a few reasons why a third party assessor or a vendor may want to have some of the tools at hand: when building its own vulnerability assessment capability or a more advanced analytical or auditing function, in order to replicate, validate and debug findings. Many of the tools are freely available in the Internet, so this section will focus on a few things that might require budgeting for.

In terms of infrastructure, at Clinical Information Security we have created an isolated network that is used only for security assessments. It provides wired connectivity via a couple of gigabit switches and also features a Wi-Fi access point. Systems on this network may be granted Internet access (for targets that require connective to cloud services, for instance), but are restricted by a firewall from any accidental interaction with the Mayo Clinic network.

On our isolated network sits a virtual machine server powerful enough to run a couple dozen virtual machines with standard 1 CPU, 4GB configurations. This provides capacity to execute multiple assessments simultaneously. The virtual machines on this hardware are either components proprietary to the target systems, e.g. an application server, "generic" components emulating provider network infrastructure, e.g. Mayo Clinic's Active Directory server, or ancillary systems that support the network or the tools themselves, e.g. firewalls, license servers, etc. The virtual machines are stored on a network attached storage device with redundancy to protect against disk failures. Test environments and their different configurations can be archived and restored as needed. Microsoft Windows operating systems and their licenses come from a MSDN subscription. Images of installation discs for a variety of open-source operating systems are also kept in the network storage to facilitate deployment of any new test environment.

When it comes to software, having a vulnerability scanner is fairly basic and advisable. A good scanner will help mapping out the attack surface and alert for some vulnerabilities in the early stages of the assessment, prior and in preparation to the more manual phases. Nessus is very popular and accredited solution. Other vendors, such as Rapid7 and Qualys, also provide good tools for vulnerability scanning and management.

For web application security assessments, good tools are essential. Burp Suite is an inexpensive solution that is used by nearly all web security specialists, making it virtually essential. In addition to it, one may consider also adopting a web vulnerability scanner, such as Acunetix, as a step further.

Likewise, many of the reverse engineering tasks have its performance greatly depending on the toolset used. For native code, IDA Pro is the dominant solution in the field. Its decompiler plugin, Hex-Rays Decompiler, is also popular and found to be extremely useful. For .NET and Java code, there is a good variety of adequate decompiling tools and they are mostly free.

Fuzz testing is a great way to automate some of the experiments that lead to vulnerability discovery. While free fuzzing tools have been made available over the years, Clinical Information Security has found Codenomicon Defensics to be a convenient low-maintenance solution. Another tool by the same vendor called Protecode SC provides software composition analysis, i.e. alerts for vulnerabilities in 3rd party components bundled in the target system. While neither tool is as fundamental as the others previously mentioned, they have imparted useful information to our vulnerability assessments.

Finally, something that actually falls outside the scope of our methodology (because we do not do white box source code assessments) but may be useful for vendors is a good source code analyzer. Integrating a code analysis tool to the engineering workflow is a great practice for a secure software development lifecycle and very much encouraged. There are many solutions for this in the market, and it is definitely wise to shop around before adopting any of them, but some popular choices are made by vendors such as Fortify, Veracode and Coverity.

## Scope and Test Environment

Adequately scoping is essential for a vulnerability assessment. Underscoping may result in missing critical vulnerabilities in components the testers were not invited or allowed to assess. On the other hand, overscoping may incur in wasting valuable time testing systems that for some reason are not relevant for the current exercise, and more time will be wasted filtering this information out of the report and off the remediation phase. This time could have been better spent assessing the components and subsystems that do matter, and for that reason important issues might have been overlooked.

In terms of deployment of the test environment, it is best to try to simulate a typical production environment as realistically as possible. Otherwise if, for example, a database server and an application server are converged into one machine in the test environment for convenience, then some aspects of the relationship between both components will not be as evident and potential issues leading to lateral movement will be neglected. Sometimes, though, a real production-like environment will not be attainable due to logistics, e.g. shipping an oversized gantry to the test location, or otherwise prohibitive costs to replicate the external setting, e.g. buying and configuring an EMR system, PACS, etc. In either case, imperfections in the test environment must be communicated to the security researchers prior to the assessment start, so they can be mindful of missed attack surface and potential issues that could arise in different scenarios. These shortcomings must also be noted in the final report.

For most assessments, the testers must be physically present at the location where the test environment resides. For testing of devices, this is the only way that some of the tests for physical attacks can be performed.

In the case of software assessments, physical access to the hardware, e.g. a server in a datacenter, may be reasonably out of scope and, therefore, can be waived. The testers must still be on the local network, though, as this is the primary scenario for remote attacks. Most often this means physically connected to the local network. A remote VPN connection to the network may seem like a viable alternative at first, but besides introducing potential points of failure that can disrupt the assessment, its latency greatly hinders the execution of some experiments, e.g. brute forcing, fuzzing and timing attacks.

Another important point concerning the test environment is its configuration. It is recommended that the deployment is populated with at least some test data: a few user accounts with different privilege levels, bogus patient data, etc. Otherwise some vulnerabilities may go undetected. Setup of most options should follow the deployment manuals and documented best practices, even if more secure settings are available. Tuning the system to non-standard configurations may result in concealing issues that are widespread across the target's customer base.

## Vulnerability Landscape

It is fair to say that, in average, medical devices and systems supporting the clinical environment are fairly complex. In this sense, it is unrealistic and impractical[1] to expect the testers performing a security assessment to check for every single possible vulnerability that could ever affect any of the different components via the various vectors in a vast attack surface. A logical conclusion, then, is that some form of prioritization, a 'checklist' of sorts, is required for the exercise to complete with efficiency in the allocated timeframe.

[1] It is worth noting that the results of a vulnerability assessment should never be interpreted as a comprehensive list of all of the vulnerabilities in the given system. An assessment attests to the existence of vulnerabilities, but cannot guarantee the absence of them.

In cyber security, however, the threat scenario is always changing. On one hand, the threat sources change: their motives change and so do their skillset and resources. On the other hand, the actual vulnerability landscape, from a purely technical perspective, also changes all the time. As new technologies are introduced or made more prevalent in any specific environment, as new vulnerabilities, techniques, tools, and attack methods are developed, or even as new defense mechanisms are adopted… all of that influences the volatile state of risk likelihood, sometimes rather drastically, for many various threats.

Therefore, this aspect from the very nature of vulnerability assessments tells us that, differently from some other auditing functions, it is naïve and insufficient to strongly adhere to checklists expecting them to represent full coverage of the relevant threat scenario. Despite path traversal, for example, not being a very prevalent vulnerability in this day and age, it still has potential for severe harm and, thus, must be checked for. The challenge of using vulnerability assessment checklists is even bigger when focusing on a specific industry, such as healthcare, because the majority of the public resources on the subject tend to be of most concern to the more widespread consumer/enterprise technology, whose current state of security affairs might not be representative of the healthcare technology industry. In result, vulnerability checklists, in this environment and methodology, cannot be solely relied upon and only skilled and experienced assessors will possess the knowledge to be leveraged for going beyond guidelines and finding the issues pertinent to the proposed threat models.

Having that said, there are a few resources that list and discuss security vulnerabilities that greatly concern the Clinical Information Security team and that should be checked for in any vulnerability assessment implementing this methodology. Having a good understanding of each of the issues listed in these lists is absolutely essential for an assessor to perform the testing properly, so it may be useful to go over them before doing an assessment under this methodology.

- CWE/SANS Top 25 Most Dangerous Software Errors – This list not only enumerates a broad range of quintessential software vulnerabilities, but also provides valuable context, discussion and pointers to more information about them. This list can be correlated with the issues' CWE IDs, discussed in the 'Deliverables' chapter. http://cwe.mitre.org/top25/
- OWASP Top 10 – This list focuses on web application security vulnerabilities and has two editions published: 2010 and 2013, both very relevant. CWE IDs also make reference to this list. http://owasptop10.googlecode.com/
- Mayo Clinic Policies and Standards – This refers to the collection of standards and requirements devised from Mayo Clinic Information Security policies, processes and procedures, backed by and compliant with established security best practices norms and frameworks.

## Targeted Network Vulnerability Assessment

Network vulnerability assessments focus on identifying security issues in networking devices and servers in the test environment. Following this methodology, the IP addresses of the network devices and servers that support the target systems are provided to the assessment team so that reconnaissance is not required. The primary steps performed in a targeted network vulnerability assessment are: host discovery, operating system identification, service enumeration, topology mapping, vulnerability identification and exploitation. Each phase is described below.

1. Host Discovery: Host discovery is initiated with ICMP ping sweeps to determine live hosts. However, this process does not always produce a comprehensive list of network-routable hosts because ICMP packets are sometimes rejected by firewalls or other filtering devices. Consequently, TCP and UDP-based discovery methods that query commonly available services such as HTTP, SMTP, DNS, VPN services and SNMP to detect

network-accessible hosts through positive responses are also used. In addition, source port scans are used to locate additional hosts that may be protected by router access controls. Tools often used in this phase include nmap, hping2, RuT, and scanline.

2. Operating System Identification: In order to build a comprehensive device inventory it is essential to determine the underlying operating systems of the live hosts. Operating system identification is performed by executing a number of publicly available tools. These tools identify subtle variances received in response to specially crafted TCP and UDP packets directed at the live hosts. The variations in the responses arise due to the minor differences in the implementation of TCP/IP stacks of different operating systems. In addition to using automated tools for this purpose, the banners returned by the various service daemons listening on the particular host are evaluated. These banners often provide information about the type of operating system deployed. Tools used in this process include nmap, xprobe2, netcat, sysinternals tools, and custom scripts.

3. Service Enumeration: Once live hosts have been determined, listening services on TCP and UDP ports are identified. This is achieved by executing automated and manual TCP and UDP port scans against the list of known active hosts. The type of scans include TCP connect scans, TCP SYN scans, UDP empty packet scans, and UDP data packet scans. Initial port scans are targeted to probe only the commonly available services like HTTP, SMTP, DNS, NetBIOS, etc. Depending on the responses received, the targeted scans are often followed by comprehensive scans that query all 65536 (0-65535) TCP and

UDP ports. In addition to determining open ports, assessors also verify the services running on them by attempting to map the ports to the corresponding service daemons by running automated service fingerprinting tools and evaluating the banners returned by the various service daemons listening. These banners normally yield information that indicates the version of the service running. The types of tools used in service enumeration include nmap, SinFP, amap, tcpdump, Wireshark, and custom scripts.

4. Network Topology Mapping: A combination of ICMP, TCP and UDP-based route tracing methods are used to determine the various paths into the network. The results of these network probes are assembled to create an external network map of the targeted servers or segment. Many of the tools solicit information from misconfigured routing devices, allowing an outside attacker to not only determine the network architecture available to the Internet but also to discover implemented protocols.

5. Vulnerability Assessment: Once the target environment has been profiled tools are executed to determine potential vulnerabilities associated with the operating systems and services of the live hosts identified. The tools identify potential issues and then manual verification is performed to eliminate false positives. The tools used at this step are mostly vulnerability scanners, like the ones discussed at the 'Toolset and Equipment' section of this document.

6. System Exploitation: On occasion, vulnerability exploitation will be attempted in order to verify that the vulnerability can actually be leveraged within the environment. The exploit process

entails using operating system commands and/or executing verified exploit code to gain unauthorized access to any vulnerable machines. If the initial compromise is successful additional steps to expand the compromise into the environment may be taken. Tools and approaches used include tested exploits, native MS Windows and Unix commands, netcat, osql, isql, brutus, metasploit framework, securityforest, and custom scripts.

## Web Application Assessment

Web application security assessments attempt to determine whether the application prevents users from performing unauthorized activities such as accessing data other than their own or gaining privileges on the system that allow them to perform unintended activities. While some automated tools such as fuzzers, web proxies and source code analyzers are used, much of the assessment activity in this area is performed manually. The primary activities performed in web application assessments are reviewing the host for operating system vulnerabilities, application mapping, HTML source sifting, authentication/authorization testing, session management analysis, input validation testing, web services assessment and limited source code analysis. Each phase is described in more detail below:

1. Server Vulnerability Assessment: Before the application logic is evaluated it is important to determine if the infrastructure running the application been appropriately secured. If the servers the application runs on have not already been evaluated as part of a network vulnerability assessment those steps should be taken at the beginning of a web application assessment. The primary steps performed in a targeted network vulnerability assessment are: host discovery, operating system identification, service enumeration, topology mapping, vulnerability identification and exploitation. Both automated tools and manual techniques would be used to assess the infrastructure.

2. Application Mapping: The structure of the web site is determined by using automated spidering tools and/or manual browsing. Generally, this is performed as both an unauthenticated user as well as with valid credentials. Techniques used include URL harvesting for users with different levels of permissions, parameter enumeration, directory structure identification, evaluating the use of client-side scripting languages and the identification of hidden fields. Automated tools used include AppScan, Nikto, netcat and SSLDigger.

3. Source Sifting: This activity involves reviewing client-side source code such as HTML, JavaScript or AJAX for insight into how the code is constructed, what information is being passed to/from the server, how validation is being performed, what comments exist and if hardcoded values are being used. Generally, the site is duplicated using a spidering tool and the source is saved locally for review. Automated tools used include Burp proxy, various web testing proxies and custom scripts.

4. Authentication/Authorization Testing: This testing begins by attempting to gain privileged access to the application as an unauthenticated user through the execution of a SQL injection attack or browsing to a known "privileged" URL of the application. Next, invalid logons are attempted and verbose error messages are enumerated. Error messages may indicate whether the username or password is incorrect. Differential analysis between the URLs accessible by users of varying privileges is also performed. Attempts are also made to access URLs with higher privileges from accounts with lower level of access. Tools such as Burp proxy and custom scripts are used.

5. Session Management Analysis: The application is reviewed to determine what session management mechanisms are used. Session management techniques such as cookies, URL parameters, URL rewriting and hidden form fields are reviewed to determine how the information is passed to the application. Session tokens are collected to determine their predictability. Brute force techniques are used to attempt to forge valid session tokens. Token persistence and time-out factors are also reviewed to determine if they are appropriate given the application functionality. Tools used include Burp proxy, netcat and CookieSpy.

6. Input Validation Testing: Server-side validation issues lead to SQL injection, cross-site scripting, buffer overflow, header injection and arbitrary command execution vulnerabilities. With the exception of the use of fuzzing tools, most testing in this area is performed manually and involves systematic entry of specific data strings into page fields in an effort to present the application with a situation it does not recognize or does not handle elegantly. In some cases HTML hidden tags are also modified. In addition to manual testing, tools such as Burp proxy and custom scripts are used.

7. Web Services Assessment: When web services are implemented they can provide another attack vector into a network. If web services are assessed for security vulnerabilities a number of aspects are reviewed. On the server the SSL, Web Services Security, XML Signature and XML Encryption configurations are evaluated. Authenticated user and unauthenticated user activity is simulated against the web services to verify that the functionality provided matches what was intended. Session

management testing is performed to confirm that authentication and authorization cannot be bypassed. Replay attacks and request delays are conducted to verify that time-based attacks are not possible. Parameter tampering involves testing exposed operations by inserting input in an attempt to verify server validation protects against buffer overflow, SQL injection, cross-site scripting, XPATH injection and XML injection. Tools used include Burp proxy and custom scripts.

8. Limited Source Code Analysis: If source code is provided a review of code structure, general coding practices and cryptographic practices is performed. The review is primarily manual and is limited to a few specific areas because automated source code analysis tools tend to produce a significant number of false positives. While automated code reviewers may be used custom scripts are the primary tools used.

## Native Software Review

Native software reviews refers to the security assessment of software other than that hosted on web platforms. In this sense, 'native software' means both compiled to machine code as well as bytecode and interpreted language scripts. These include desktop applications, APIs, networks services and any auxiliary software. This step of the process investigates not only design and implementation aspects of the assessed target but also how the software is configured and whether it follows security best practices and adheres to Mayo Clinic's information security policies and standards. The tools used throughout this process are essentially disassemblers and decompilers for the relevant technologies, in addition to custom scripts. Following is a non-comprehensive list of technical reviews performed:

1. Hardcoding of Credentials: The software and its respective configuration files are inspected to see if they contain any form

of hardcoded credentials (i.e. credentials that cannot be safely modified by the customers to their discretion).

2. Credential Storage: If the software stores credentials of any form (e.g. of its user database), the security of this implementation is assessed to verify if the best cryptographic and access control measures have been taken.

3. Communication Channels: The communication of software over the network is inspected to check that transmission of sensitive data occurs over an encrypted channel, and that data authenticity and identity of both parties can be verified as appropriate.

4. Authentication and Authorization: The system is verified to see if access to critical functions is controlled by suitable authentication mechanisms and that authorization checks are correctly performed, enabling the system administrator to separate privileges with an appropriate degree of granularity. Included in the authentication review are credential handling, password policies and brute force protections.

5. Information Disclosure: Search for unnecessary artifacts that may provide a threat actor with useful information for an attack, such as in error messages, log files, leftovers of installation media, etc.

6. Limited Reverse Engineering: Executables, proprietary data formats and proprietary communication protocols are partially reverse engineered in order to perform many of the reviews above and additional checks, such as for secure input validation

(which, when not done properly, may offer an avenue for denial-of-service, privilege escalation or code execution attacks).

## Host Configuration Review

Host configuration reviews assess the security of various network devices and servers with specialized purposes. While some guidelines are consistent across all devices, industry best practices can vary depending on the nature of the device. Host configuration policies and standards are reviewed and a variety of technical aspects of the systems are evaluated. The types of technical reviews performed include:

1. Account and Password Settings: Default accounts, account lockout policies, password history maintenance and password expiration settings should all defend against brute force attacks.

2. Password Strength: Passwords should be complex and should not be easily cracked with brute force methods.

3. Account Privileges: User and service accounts should reflect the principle of least privilege.

4. Logging and Auditing: Verify that logging settings are sufficiently granular and the retention periods are sufficiently long to support the alerting and investigative requirements of the organization.

5. Software Patching: Patch levels should be current for the base operating system as well as applications.

6. Registry Settings (Windows Systems Only): Review selected items such as start-up services, guest logon restrictions, Dr. Watson logging and user credential encryption.

7. File System Access Controls: Verify that the principle of least privilege is being applied for key files and directories.

8. Backup Strategies: Verify backup and recovery functions are in place and periodically tested so that critical data is available even in the event of server failure.

9. Disk Encryption: Check that systems containing sensitive data have full-disk encryption enabled to avoid malicious data recovery.

10. Common Misconfigurations: Check for various common default misconfigurations in services such as WWW, E-Mail, FTP and DNS to ensure only required services are running.

11. Anti-Virus and Compromise Indicators: Review systems for the presence of backdoors and malicious start-up processes. Confirm that anti-virus or equivalent is installed, patched and executed on a regular schedule.

12. Server-Specific Activities: Activities that are unique to a certain class of server are also performed. For example application servers are reviewed for the presence of default scripts, server-supported web methods and unmapped file handlers. Database servers, on the other hand, would be reviewed for stored procedures with excessive permissions or default access controls in user tables.

## Physical and Hardware Review

Besides the most common network and software attacks, it is often important to analyze which aspects of the hardware and its disposition may serve as vectors for threat actors that can leverage physical access to the target systems. Examples of verifications in this category include:

1. Physical Construction Review: Verifies that access to critical physical mechanisms is restricted by some kind of control. For some devices it may simply be a check that are no active USB ports exposed, for others it may be the case that even an on/off switch should be protected.

2. Lock Picking: When parts of the device are protected by physical locks, the resistance of such locks may be tested by picking them with specialized tools.

3. Pre-Boot Controls: While not a hardware-specific problem, it must be assured that the integrity of the boot sequence cannot be interfered with. This includes checking that the BIOS and the boot loader are password-protected, that the boot order is not insecurely configured (e.g. prioritizing removable media), and that no accessible switches can bypass these controls. If Secure Boot or similar mechanisms are implemented, their security may also be assessed.

4. Tamper Protection: For some devices, it may be required that they are tamper resistant or, at least, tamper evident. Specialized tools and techniques can be applied to circumvent insecure protections and leave a maliciously altered device not only operational but also inconspicuous as to whether it has been modified.

5. Debug Ports: The hardware is searched for debug/service ports (e.g. JTAG, serial) that can be tapped into and used to compromise the device operation. The communication protocol used by these ports may be reverse engineered and commands

may be scanned for a more precise assessment of its functionality and the risks associated with it.

6. Firmware Analysis: In order to properly analyze the security of embedded devices, it is often necessary to extract firmware files and reverse the executables contained in them. For a black box kind of testing, when the firmware is not readily available, it may be extracted from the flash memory chips in the hardware board.

7. RF Analysis: Many devices employ radio frequency communication to operate wirelessly and, regardless of field range, its security must be assessed for it may feature remotely exploitable vulnerabilities. Most commonly, this means a review of the protocols and their parameters used in a Wi-Fi-enabled device. In other cases, it may involve a different RF technology (e.g. Zigbee) or a completely custom protocol, which must be reverse engineered for analysis.

## Technical Staff Interviews

In many cases system developers and support personnel can provide important context for a vulnerability assessment exercise. They may be aware of undocumented policies or be able to explain why certain approaches have been taken. The information requested from these individuals varies considerably depending on the scope of the engagement and the testing approach used. Examples of information that can provide useful perspective include:

1. Software development life cycle (SDLC) practices, standards and documentation.

2. Descriptions of secure coding practices and how they are incorporated into the SDLC.

3. Undocumented practices associated with system deployment and support.

4. Technical architecture overview and how that has affected different security approaches.

5. Past vulnerability assessment, audit and incident response experience.

6. Patch management cycles and how they are implemented.

7. Operational limitations that place limits on remediation measures or require compensating controls to be implemented instead.

## Summary

The following figure shows the 3 phases of a vulnerability assessment, listing the steps carried out at each stage:

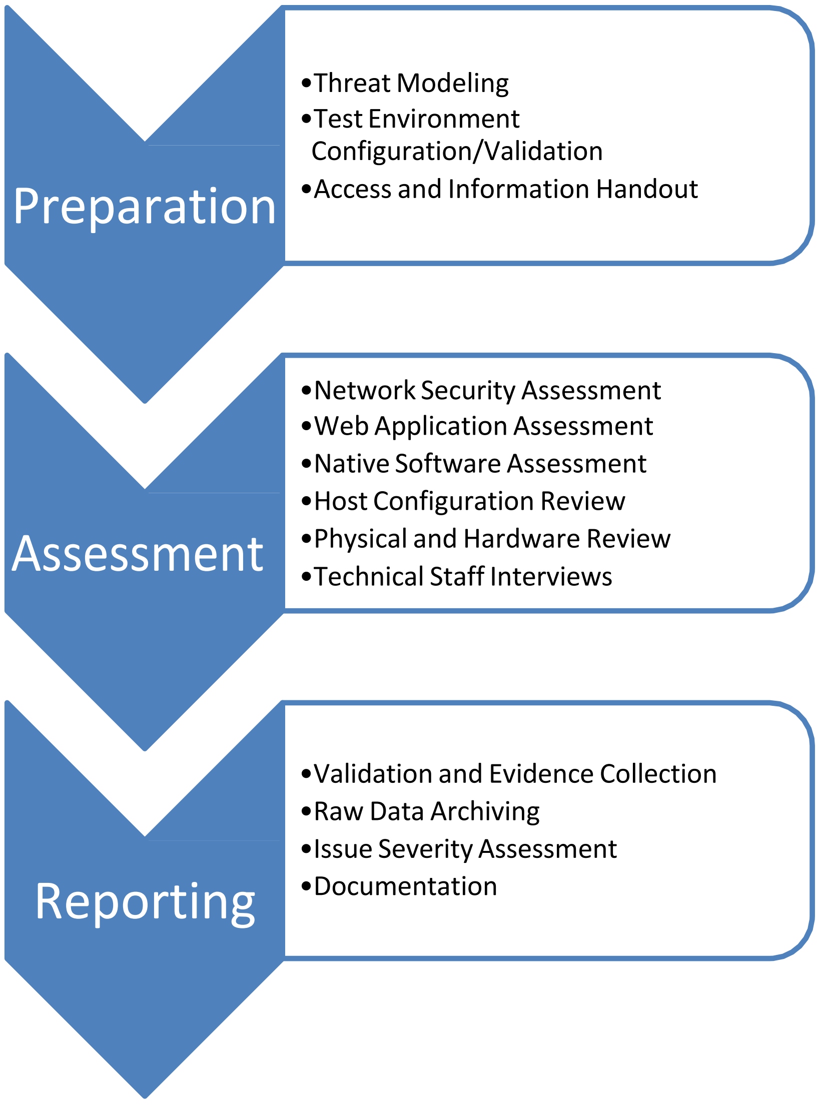
Preparation
•Threat Modeling
•Test Environment Configuration/Validation
•Access and Information Handout
Assessment
•Network Security Assessment
•Web Application Assessment
•Native Software Assessment
•Host Configuration Review
•Physical and Hardware Review
•Technical Staff Interviews
Reporting
•Validation and Evidence Collection
•Raw Data Archiving
•Issue Severity Assessment
•Documentation

**Figure 1. Methodology in a nutshell**

# Deliverables

The main deliverable of a vulnerability assessment is a report. The report contains detailed information about the manifestation of each issue, its severity and how it can be exploited. This information makes risk analysts both on the third party assessor as well as on the vendor side aware of the security vulnerability and helps them determine an action plan and track it. In most cases the plan is to address the issue and, therefore, the report must feature specific recommendations that explain how both parties can mitigate the issue to the greatest extent possible.

The next few topics go into detail about each of the sections required for proper vulnerability assessment reports.

## Introduction

The first few sections of a vulnerability assessment report should discuss the scope and environment of the assessment. It is important to state exactly what was assessed, when, and under which conditions. If any deviations from the standard methodology were employed, e.g. because of any project constraints, they must be properly noted so that the impact to the exercise is understood.

A short description of the system and its applications help in putting context around the assessment as well as understanding the actual risks involved with each finding, while a network diagram is useful to explain how a large target is laid out and how its subsystems communicate with each other. Then for such multi-component systems it is important to describe each component in one or two sentences so that the impact of the vulnerabilities is also comprehended at finer granularity.

An important function of the introductory segment of the report is documenting the exact version of the system that was tested. This will be useful for readers when validating the findings against varying device versions. To be comprehensive, assessors must note both software versions as well as firmware and hardware revisions, when applicable, for all relevant system components.

A table of findings provides the reader with an overview of the assessment results. In reports produced by Clinical Information Security, this table contains at a minimum the title of each issue, its severity, the components affected, and the responsible party (third party assessor, vendor or both). Findings on this table are sorted by severity, high to low.

## Regression Testing

In case the report is for the assessment of a system that has been tested before, it is useful to provide information that contrasts the results of the last assessment with the current one. Typically a re-assessment is performed on a newer version of the product, so this comparison enumerates which past issues have been fixed, which new ones may have been introduced as well as some that have been overlooked in the previous assessments.

In reports made by Clinical Information Security, regression testing is summarized by a table that lists the issues from the last assessment and has a column to state whether each one of the vulnerabilities has been fixed, partially fixed or not addressed at all. If the vulnerability persists, then the title and severity of the issue in the current report are also listed, to account for changes in title/severity across different assessments. Finally, a column of notes provides the assessors with a space to make relevant observations about the security problem.

## Issue Severity Rating

Security vulnerabilities must be evaluated in an objective and consistent manner. Clinical Information Security uses CVSS (Common Vulnerability Scoring System) v3 as the underlying methodology for rating issues found during vulnerability assessments. CVSS is a standard supported by a large number of frameworks and tools.

At a minimum, the report must contain, for each issue, the CVSS base score and its corresponding vector string. Other vectors and scores, such as temporal and environmental, are optional and may be included to provide additional context to the vulnerability. The base vector will also provide two subscores, namely impact and exploitability, which should be present in the report for clarity and easy reference.

The overall issue severity is calculated by correlating the base score to predefined numeric ranges. The ranges applied are the same as the ones used by NIST's National Vulnerability Database (NVD) as follows:

| Issue Severity Rating |
|---|
| High<br>(CVSS 7.0 – 10.0) |
| Medium<br>(CVSS 4.0 – 6.9) |
| Low<br>(CVSS 0.0 – 3.9) |

## Issue Entry

The largest portion of a vulnerability assessment report's contents is comprised of the details on each issue. Each vulnerability should have its own entry in the report, which in turn must contain a minimum set of data about the issue that guarantees the reports are useful and undeviating. More information can be included if found to contribute to the report, but the obligatory fields are:

- Issue title – a short name that is descriptive of the problem and easy to refer to, e.g. "Hardcoded Passwords in Application Binary".
- CVSS – the numeric base score as well as its corresponding vector string.
- CWE – the ID(s) of the corresponding Common Weakness Enumeration entry(ies) relevant to the issue.
- Description – a description of what is wrong with the system that allows for this issue to exist.
- Affected Systems – the list of systems, subsystems or components that are directly affected by this vulnerability.
- Impact – A detailed description of the potential impact of this vulnerability, if exploited. The numeric CVSS impact subscore may be included for reference. Unlike the technical details (discussed later in this document), the impact must be expressed in functional terms for consumption by non-technical audiences. So, for example, when a vulnerability is able to "grant unauthorized admin privileges", the impact description is supposed to list what malicious actions an attacker would be able to perform with such privileges. Any constraints that reduce impact of a given issue (e.g. memory corruption that can only be used for DoS but not code execution) must be documented with accompanying evidence in the 'Technical Details' section.
- Exploitability – this field describes what preconditions are required for the issue to be exploitable, such as a valid user account, a privileged network position (e.g. man-in-the-middle) or the combination of another vulnerability. Any constraints that reduce exploitability of a given issue (e.g. memory protections) must be documented with accompanying evidence

in the 'Technical Details' section. Like with the impact, the CVSS exploitability subscore may be included for reference.

- Level of Effort to Remediate – a simple low/medium/high estimate is enough to give a sense of priority, when combined with the issue severity.
- Responsible Party – the entity responsible for remediating the vulnerability. May be the vendor (most design and implementation issues), the user/third party assessor (some configuration issues) or a combination of both. An example of shared responsibility is when changing hardcoded or default passwords that may only be changed by the vendor but the new passwords must be chosen and further managed by the provider.
- Recommendations – descriptions of the steps necessary for the vendor and/or provider to address the vulnerability. If the actions to be taken by the vendor and the third party assessor are different, the recommendations may be split in two. The priority and main purpose of this information is to provide a fix that mitigates the risk to the best extent possible and improves the security posture of the system to the greatest level, even if it is non-trivial. However, if short-term workarounds are feasible and available, they should be included so that a multi-stage action plan may be considered.
- Technical Details – all the remaining technical information about the issue, including mainly how and "where" it was found, but also any additional commentary that may be relevant about prevalence, impact, exploitability or remediation of the issue. This section commonly contains a number of screenshots illustrating the discovery and analysis of the vulnerability, pinpointing to the specific sources of the problem. The screenshots must be annotated with text, arrows and boxes that highlight the pertinent data. If unclear from the screenshots, all tools relevant to discovering and exploiting the issue must be

noted in the text. In any case, all steps for both the discovery and the exploitation processes must be thoroughly described. This information is crucial for the remediating party to identify, replicate and acknowledge the issue, and for the vulnerable party to comprehend the vulnerability in sufficient detail. Proof-of-concept exploit code may be included for validation of the issue. When there are multiple points vulnerable to the same issue they should all be listed in this section, but if the relation is too long (e.g. list of all patches missing on an outdated OS or all URLs and parameters vulnerable to SQL injection), however, the list may be moved to an appendix to the report in order to improve readability of the document.

## Raw Data

Security assessors typically generate more content than what makes it into the report. This data is compiled all the way along the assessment and is filtered out of the report either because it is not indicative of any particular vulnerability, purely informational, or because there is better evidence that takes precedence. Regardless of being convenient for the report or not, all of this raw material must be archived and preserved, since it provides additional and unique insight into some of the issues or the overall state of the test environment, information that is very useful when analyzing any particular finding in depth. Examples of raw vulnerability assessment data include:

- Output from automated scans (e.g. nmap, Nessus reports, etc.);
- Video, additional screenshots or other media not used in the report;
- Decompiled code and reverse engineering databases (e.g. IDB files);
- Memory dumps, execution traces and other dynamic analysis artifacts;
- Raw notes taken by the assessors;

- List of hashes cracked, passwords recovered;
- Exploit code, tools and custom scripts developed throughout the assessment.

# Internal vs. External

There are subtle differences in how the vulnerability assessment process is executed depending on whether it is exercised by Clinical Information Security itself or third parties acting in cooperation, such as a device vendor consulting with security specialists. This chapter discusses these variations in the arrangements and provides guidance for assessments to be performed either way.

## Internal Vulnerability Assessments

When a vulnerability assessment is to be conducted by Clinical Information Security, its test environment must be deployed at the team's office in Rochester, MN. In rare cases when this office will not attend to the testing needs, a different Mayo Clinic space in Rochester may be chosen. The vendor must provide loaner equipment to be tested, i.e. the devices in the testing environment. This equipment may be shipped to the test labs or delivered in person by the vendor, as agreed by both parties. Although some resources from the lab itself, such as a virtual machine server, may be available to be used for the assessment, it is usually better if the vendor can provide all the basic hardware. All networked equipment must be plugged onto the isolated test lab network and not on the hospital network.

The initial testing environment must be deployed and configured by a representative from the vendor, who will sign off a statement confirming that the software and hardware were updated to the relevant versions and that everything has been configured to a state that is archetypal of a common production setup. In some instances, e.g. when the scope is purely software, this step may be done remotely. For that, a Mayo Clinic engineer would stand up the necessary number of virtual machines for the required platforms and let the vendor "come" in via secure methods to deploy and configure all the software, operating system included. In any case, this part of the process includes populating the system with test users and data.

At some point, usually once the configuration of the test environment is complete, both parties must agree on a procedure for Mayo Clinic to return the vendor loaned equipment. Very often the vendor will come back onsite, perform the uninstallation and collect the equipment. For small devices, however, Clinical Information Security may be able to ship the hardware without requiring the vendor presence, if so the vendor prefers. In such cases, the vendor must provide the shipping labels and Mayo Clinic is not liable for any damage potentially done to the equipment during shipping and handling. Clinical Information Security preserves all original packaging during the assessment so that it can be used again when returning the hardware. The vendor is then responsible for executing all the procedures required for "refurbish" the equipment and validating it is suitable for clinical/production application before being put to such kind of use.

One or more vendor representatives must come onsite to the test labs for at least the duration of one day to brief the assessors in the usage and internals of the system. This should be done on the first day of the assessment (after deployment and configuration are finished) and must be done in person. Due to the deeply technical nature of the information that is exchanged at this moment, the vendor staff involved must possess extensive knowledge about the engineering aspects of the product – usually software engineering leads that work directly on the development of the target system. The assessors may have important inquiries at any time during the assessment, so it is useful to also exchange contact information at this point and establish a preferred channel for questions and answers.

When writing the vulnerability assessment report, the assessors must produce a version of the document that has all Mayo Clinic confidential information sanitized, i.e. blurred out or completely removed. This includes user and patient data which, although unlikely to be present in most test environments at all, might turn out being accessible if real data is used for populating the test systems. The sanitized report is the one that must be shared with third parties.

## External Assessments

External vulnerability assessments are those conducted by a third party, usually a consultancy firm commissioned by the vendor to test the vendor's product for security issues. Vendors working in cooperation with Mayo Clinic will arrange external vulnerability assessments and have Clinical Information Security review the results to discuss action plans for remediation. For this process to achieve the desired goals, attention and adherence to the methodology described in this manual is expected, including but not limited to:

- Ensuring the assessment is performed by experienced professionals that excel in all of the skills mentioned in the methodology chapter (reverse engineering, web application security, embedded device hacking, cryptography, etc.);
- Requesting that a threat modeling exercise is performed to devise a test plan that addresses Mayo Clinic's security concerns and threat scenarios;
- Requiring that the resulting report contains at least the information mentioned in the 'Deliverables' chapter and that additional 'raw data' is preserved and supplied;

- Allowing appropriate time and providing an adequate test environment;
- Requesting that all techniques of the methodology are considered and exercised when applicable, and also that special attention is paid to the common vulnerabilities discussed in the 'Methodology' chapter;
- Providing the testers with substantial access and information so that otherwise low-hanging issues are not overlooked.

It is understood that one particularly difficult and vital point from the above is the engagement of excellent professionals with all the necessary capabilities. Assessors with this specific combination of skills and that have the level of experience and creativity to simulate the capabilities of the most advanced threats are known to be hard to find. For this reason, it is hereby provided a list of consultancy firms that Clinical Information Security understands as possessing the desired skillset for this kind of vulnerability assessment:

| Company | Contact |
|---|---|
| Synopsys | • Matthew Ziobro, MZiobro@synopsys.com, 262-757-5499<br>• Kevin Nassery, knassery@synopsys.com, 224-520-2424 |
| Black Hills Information Security | • C.J. Cox, cj@blackhillsinfosec.com, 701-484-2447<br>• Email: consulting@blackhillsinfosec.com<br>• Fill out a request for contact at this link: www.blackhillsinfosec.com |
| Batelle | • Rick Brooks, brooks@batelle.org, 614-424-6358<br>• David Giles, GILESD@battelle.org, 614-424-5612 |
| CENSUS S.A. | • Nikolaos Tsagkarakis, ntsag@census-labs.com<br>• Phone: +30 2310 947 233 |
| Immunity Services LLC | • Email: sales@immunityinc.com<br>• Web Site: www.immunityinc.com<br>• Phone: 786-220-0600 |
| IOActive | • Email: sales@ioactive.com |
| NCC Group | • Allison Arvizu, allison.arvizu@nccgroup.trust, 714-625-3466<br>• Web Site: www.nccgroup.trust |
| Norwin Technologies | • Phone: 978-767-4350<br>• Web Site: http://www.norwintechnologies.com/contact.html |
| Protiviti | • Adam Brand, adam.brand@protiviti.com, 213-260-4660 |
| Recurity Labs | • Dirk Breiden, dirk@recurity-labs.com<br>• Florian Grunert, florian@recurity-labs.com<br>• Phone: +49 30 69539993-0 |
| Tangible Security | • Matt Rahman, mrahman@tangiblesecurity.com |

# Glossary[2]

- **Attack scenarios** – Hypothetical situations that satisfy all requirements for one or more vulnerabilities in a given system to be exploited by a capable adversary.
- **Attack surface** – The set of ways in which an adversary can enter a system and potentially cause damage. An information system's characteristics that permit an adversary to probe, attack, or maintain presence in the information system.
- **Attacker profiles** – Descriptions of capability, intent, targeting and motivators that outline an adversarial threat source.
- **Black box assessment** – A test methodology that assumes no knowledge of the internal structure and implementation detail of the assessment object.

[2] Including definitions from:
- https://www.sans.org/security-resources/glossary-of-terms/
- http://nvlpubs.nist.gov/nistpubs/ir/2013/NIST.IR.7298r2.pdf
- https://niccs.us-cert.gov/glossary

- **Brute force** – A cryptanalysis technique or other kind of attack method involving an exhaustive procedure that tries all possibilities, one-by-one.
- **Buffer overflow** – A buffer overflow occurs when a program or process tries to store more data in a buffer (temporary data storage area) than it was intended to hold. Since buffers are created to contain a finite amount of data, the extra information - which has to go somewhere - can overflow into adjacent buffers, corrupting or overwriting the valid data held in them.
- **Code execution attacks** – Any attack that results in the target system executing (malicious) code arbitrarily chosen by the attacker.
- **Cross-site scripting** – Also known as XSS, a vulnerability that allows attackers to inject malicious code into an otherwise benign website. These scripts acquire the permissions of scripts generated by the target website and can therefore compromise the confidentiality and integrity of data transfers between the website and client. Websites are vulnerable if they display user supplied data from requests or forms without sanitizing the data so that it is not executable.
- **CVSS** – Common Vulnerability Scoring System, an open framework for communicating the characteristics and impacts of IT vulnerabilities.
- **CWE** – Common Weakness Enumeration, a formal list of software weakness types created to provide a common baseline standard for weakness identification, mitigation, and prevention efforts.
- **Denial-of-service** – The prevention of authorized access to resources or the delaying of time-critical operations. (Time-critical may be milliseconds or it may be hours, depending upon the service provided.)

- **External threat sources** – An adversary originating from outside the target organization. See Threat Source.
- **Fuzz testing** – The use of special regression testing tools to generate out-of-spec input for an application in order to find security vulnerabilities.
- **Grey box assessment** – A test methodology that assumes some knowledge of the internal structure and implementation detail of the assessment object.
- **Host discovery** – The process that determines which hosts, comprising a given system, are live and network-accessible from an attacker's perspective.
- **HTTP, SMTP, DNS, VPN, SNMP** – Various application-layer protocols commonly served or used by communicating computer systems.
- **ICMP ping** – A status request message defined by Internet Control Message Protocol, an Internet Standard protocol that is used to report error conditions during IP datagram processing and to exchange other information concerning the state of the IP network.
- **Insider threats** – A threat agent originating from inside the target organization, e.g. one of its employees, contractors or business partners. See Threat Source.
- **Network vulnerability assessments** – Exercises that focus on enumerating and validating remotely identifiable/exploitable security issues in networking devices and servers in the environment.
- **NIST, NIST's NVD** – The National Institute of Standards and Technology, part of the U.S. Department of Commerce, and its National Vulnerability Database, the U.S. government's repository of vulnerability management data.

- **Principle of least privilege** – Least Privilege is the principle of allowing users or applications the least amount of permissions necessary to perform their intended function.
- **Privilege escalation** – The result of successful attacks that aim at augmenting the privileges an attacker has on a given target system, e.g. turning a low-privileged user into an administrator.
- **Reverse engineering** – Acquiring sensitive data by disassembling and analyzing the design of a system component.
- **Secure software development lifecycle** – A set of practices that ensure systems are developed against high security standards. Secure SDLC covers all stages of development: planning, analysis, design, implementation and maintenance.
- **Security assessment** – A test methodology in which assessors, typically working under specific constraints, attempt to circumvent or defeat the security features of an information system.
- **Source code analyzer** – A tool with some degree of automation that sifts through source code and identifies security problems.
- **SQL injection** – A type of input validation attack specific to database-driven applications where SQL code is inserted into application queries to manipulate the database.
- **TCP and UDP port scans** – A port scan is a series of messages sent by someone attempting to break into a computer to learn which computer network services, each associated with a "well-known" port number, the computer provides. Port scanning, a favorite approach of attackers, gives the assailant an idea where to probe for weaknesses. Essentially, a port scan consists of sending a message to each port, one at a time. The kind of response received indicates whether the port is used and can therefore be probed for weakness.

- **Threat modeling** – A threat model is used to describe a given threat and the harm it could to do a system if it has a vulnerability. Threat modeling is the exercise of establishing an documenting threat models for a given system.
- **Threat source** – The intent and method targeted at the intentional exploitation of a vulnerability or a situation and method that may accidentally trigger a vulnerability. Synonymous with Threat Agent.
- **Vulnerability** – Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source.
- **Vulnerability assessment** – Systematic examination of an information system or product to determine the adequacy of security measures, identify security deficiencies, provide data from which to predict the effectiveness of proposed security measures, and confirm the adequacy of such measures after implementation.
- **Vulnerability scanner** – A tool that automatically probes a system for a number of known security vulnerabilities and reports which ones the target seems to be affected by.
- **Web application security assessments** – Exercises that are specific to the realm of web applications and that try to identify common vulnerabilities in them.
- **White box assessment** – A test methodology that assumes explicit and substantial knowledge of the internal structure and implementation detail of the assessment object.
- **XML injection** – Attack technique that injects unintended XML content to alter the intended logic of the application.
- **XPATH injection** – Attack technique that injects content from malicious user input into XML Path Language queries to alter the intended logic of the application.