6.4 Vulnerability Assessment

A vulnerability is a weakness that could be exploited by a threat, causing the violation of an asset's security property. Conducting an enterprise vulnerability assessment helps to identify the weaknesses of the enterprise's assets and the systems that enable access to them, and evaluates the severity if a vulnerability were to be exploited.

Also Known As

Vulnerability Analysis

Example

The museum has begun a risk assessment and identified the following assets to be in scope:

Information asset types

- Museum employee data
- Museum financial/insurance data, partner financial data
- Museum contractual data and business planning
- Museum research and associated data
- Museum advertisements and other public data
- Museum database of collections information

Physical assets

- Museum building
- Museum staff
- Museum collections and exhibits
- Museum transport vehicles

The museum has also identified the potential threats to those assets and must now determine vulnerabilities that can compromise those needs.

Context

An enterprise has defined the assets to be included in a risk assessment, and has identified potential threats, for example through applying THREAT ASSESSMENT (113). It must now identify the vulnerabilities that can be exploited by those threats.

Problem

Enterprise assets and the controls protecting them may be fully secure, or may have numerous weaknesses, some of which may never be exploited, and some of which may be exploited every day. Without proper cataloguing of these vulnerabilities, an enterprise might never recognize the extent of the weaknesses of their assets.

How can an enterprise identify vulnerabilities to its assets and determine the severity of those vulnerabilities?

An enterprise must resolve the following forces:

- It might have experience with a single tool or method for discovering weaknesses, but may not be aware of other techniques that can reveal other, potentially critical, vulnerabilities.
- It need only identify vulnerabilities for which threats exist, and therefore the enterprise must be able to determine if a given vulnerability has an associated threat.
- It would like to develop a standardized way of identifying vulnerabilities and assessing their severity, in order to be consistent with subsequent vulnerability assessments.
- The solution should address all assets included in the scope of a risk assessment, including informational and physical assets, and, ideally, should be able to address vulnerabilities in non-IT systems.

Solution

Systematically identify and rate probable vulnerabilities of the enterprise assets. This process involves the following five steps:

1. Collect threat information.

   Collect information on threats. For example, if THREAT ASSESSMENT (113) has been used, appropriate threat information is available from the resulting threat table.
2. Identify vulnerabilities.

   Using the threat table, identify the vulnerabilities of the assets and the systems protecting them defined in the scope of the risk assessment.
3. Build a threat-vulnerability table.

   Extend the threat table by associating each vulnerability with a threat action.
4. Create a severity scale.

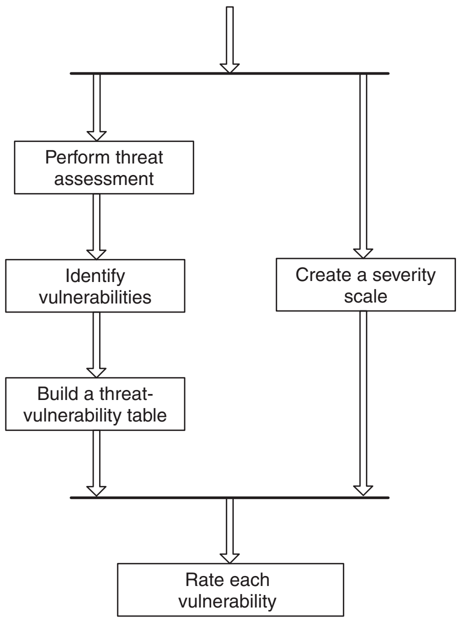
   Create a scale for rating the severity of vulnerabilities. This scale will represent the degree to which an asset is susceptible to a vulnerability, and the potential impact should the vulnerability be exploited.

5. Rate each vulnerability.

   Rate each vulnerability according to the severity scale and update the threat-vulnerability table to reflect this rating.

Dynamics

The allowable sequence for performing the vulnerability assessment process is shown in the figure.



Vulnerability assessment sequence constraints

First collect appropriate threat information. Then, using the methods outlined, identify all vulnerabilities and associate them with threats in the threat table, creating the threat-vulnerability table. A vulnerability severity scale can be developed at any time. Finally, using this scale, rate each vulnerability.

Implementation

The implementation of the process for assessing vulnerabilities is described below.

1. Collect threat information.

   Threat information should include a list of events that could cause harm to assets and provide context for the vulnerabilities.

2. Identify vulnerabilities.

   Use any of the following methods to identify vulnerabilities exploitable by the threats in the threat table.

   2.1. System characteristics.

   [WT03] describes four main causes of system vulnerabilities, and while it focuses on software applications, the causes can be generalized to help identify weaknesses in non-IT systems.

   - Those that can be caused by dependency failure. Rarely, if ever, does an application not interact with other applications or systems to perform its function. These interactions may be with database tables, shared system libraries, network services or devices, or operating system resources. The behavior of the application in the event of a failure or unavailability of these dependencies is a prime target for attack. For example, how would an application respond when a security library could not be loaded? Does it bypass all security calls, log an error and continue, or halt all operation and alert operators? How does an application respond to low system resources such as low disk or memory conditions?
   - Those that can be caused by unanticipated data input. The absence of data input validation is a very common mistake. It can also be the most damaging, because the results can range from denial of service to complete subversion of the system through full administrator access. Buffer overflows and SQL injection are two of the most prevalent examples of this class of attack.
   - Those that can be caused by design vulnerabilities. As the size and complexity of an application grows, it becomes more difficult to identify and validate the flow and integrity of data. The potential for exploiting design flaws therefore increases. Such design flaws may include use of cleartext protocols where encrypted ones are necessary, acquiring escalated privileges by circumventing access or authorization controls, assumptions made by designers or developers regarding the use of or operation of the application, and jumping outside the bounds (and constraints) of the system to perform unauthorized tasks or operations.
   - Those that can be caused by implementation vulnerabilities. A most secure design can still lead to a substantial vulnerability if the implementation is faulty. This provides another reason why scale and complexity are impediments to secure systems. The larger and more intricate a design, the more opportunities there will be for implementation errors.

An example is software that deals with sensitive information. It must ensure that while processing the information, it is not temporarily copied to either disk or memory unprotected. Replay attacks, 'bait and switch,' and 'man in the middle' are all examples of implementation vulnerabilities: a replay attack is a form of network attack in which a valid data transmission is maliciously or fraudulently repeated or delayed, while a bait and switch is a form of fraud in which the fraudster lures in customers by advertising goods at an unprofitably low price, then reveals to potential customers that the advertised goods are not available, but that a substitute is. A 'man in the middle attack' (MITM) is an attack in which an attacker is able to read, insert and modify messages between two parties without either party knowing that the link between them has been compromised.

2.2. Development life cycle.

[NIST800-30] recognizes that vulnerability identification is dependent on the nature of an IT system and its phase in a development life cycle. Differentiation is made between systems being designed, systems being implemented, and systems in production.

For systems or applications that have not yet been designed, 'the search for vulnerabilities should focus on the organization's security policies, planned security procedures, system requirement definitions, and the vendors' or developers' security product analyses.' At this stage, these design documents and product specifications are all that are available for security review.

For systems that are in the process of being implemented, vulnerability identification 'should be expanded to include more specific information, such as the planned security features described in the security design documentation and the results of system certification test and evaluation.' This is where security auditing tools can first be used to test applications before they are released into production. These tools generally perform signature-based checks to test for known weaknesses.

Finally, for systems that are already in production, vulnerability identification 'should include an analysis of the IT system security features and the security controls, technical and procedural, used to protect the system.' For these systems, security auditing tools as well as penetration tests will identify weaknesses most effectively, although not necessarily safely. They function by directly testing for the presence of known exposures, as opposed to theorizing their existence based on documentation, policy or a countermeasure that is supposed to block an attack. Penetration tests can be a complex effort requiring the cooperation of many departments, including information security, operations, and application development. As mentioned, these tests can be most effective because they test the weakness of the IT system as well as the countermeasures that are (or should be) in place.

2.3. Other.

Specialized techniques for identifying vulnerabilities in IT systems include the following:

- *Vulnerability scanning*. Vulnerability scanning is the act of running automated tools or procedural tests on networks and applications in order to detect or confirm the presence of vulnerabilities [WTS03].
- *Penetration tests*. Penetration testing is a procedure that attempts to circumvent, disable or otherwise defeat the security controls of a system using any available tool or technique.
- *Vulnerability catalogs*. These catalogs provide a list of vulnerabilities for specific applications and configurations. Examples include:
  - CERT Knowledgebase: `http://www.cert.org/kb`
  - CVE database: `http://www.cve.mitre.org`
  - NIST ICAT: `http://icat.nist.gov`
- Open Source Vulnerability Database: `http://www.osvdb.org/`
- *Vendor advisories and patch lists*. Commercial vendors and Open Source developers will often provide vulnerability advisories for their products.
- *Information security forums and mailing lists*. These lists provide a popular discussion and distribution forum for security vulnerabilities. For example:
  - Bugtraq: `http://www.securityfocus.com/`
  - SANS: `http://www.sans.org`
  - RISKS: `http://catless.ncl.ac.uk/Risks/`
  - Dartmouth College Institute for Security Technology Studies: `http://news.ists.dartmouth.edu/`

3. Build a threat-vulnerability table.

Extend the threat table by pairing vulnerabilities with threats, creating a threat-vulnerability table. Recall that threats are grouped by threat source (natural, hacker, criminal, and so on), accommodating situations in which the same threat action (for example theft) originates from multiple sources (for example employees and criminals).

This table format enforces the restriction that it is only necessary to consider vulnerabilities for which threats exist. If a vulnerability is found to have no associated threat, either remove the vulnerability from consideration, or update the threat table to include the threat.

To determine whether a vulnerability has an associated threat, ask yourself if there is any way that the security properties (confidentiality, integrity, availability, and accountability) of an asset could be compromised as a result of the

Table 6.16 Vulnerability severity scale

RATING	SEVERITY	DESCRIPTION
6	Extreme	1. The vulnerability is trivially exploitable and commonly found, or 2. Major loss of life and destruction of systems would occur
5	Very high	1. The vulnerability is easily exploitable and found in most systems, or 2. Some loss of life and major destruction of systems would occur
4	High	1. Exploiting the vulnerability would be a challenge but it exposes many systems, or 2. Human physical injury, some destruction of systems would occur
3	Medium	1. The vulnerability is difficult to exploit, and exposes some systems, or 2. Significant disruption of service and compromise of confidentiality, availability, or integrity of assets would occur
2	Low	1. The vulnerability would be very difficult to exploit, with no real gain, or 2. Slight disruption of service or mild compromise of security properties would occur
1	Negligible	1. This is a theoretical vulnerability only exploitable with massive infrastructure or computing power, or 2. Minor distraction to business processes and no compromise of security properties would occur

weakness. Importantly, this does not answer the question of who caused the compromise, or how it occurred, but simply whether it could occur.

4. Create a severity scale.

   Create a severity rating scale by first defining a rank, then assign a meaning and description to each rank. An example is shown in Table 6.16. The rating represents the degree to which the asset is susceptible to the vulnerability, and the potential impact should the vulnerability be exploited. Note that the range and description are at the discretion of the enterprise—it can change the range, severity term, and description as appropriate. The important consideration is that the table remain constant throughout the risk assessment and across the enterprise.

5. Rate each vulnerability.

   Rate the severity of each vulnerability according to the considerations listed below.

5.1. General factors:

- The number of threats that can be realized as a result of a given vulnerability being exploited. Also, the number of systems affected by the vulnerability. If a single vulnerability provides the opportunity for many threats to be realized (and perhaps many subsequent vulnerabilities to be exploited), then the severity should be reflective of this.
- The prevalence of the systems affected by the vulnerability. Some vulnerabilities may impact uncommon or infrequently-used applications or enterprise resources, while others may impact a ubiquitous Internet service or physical infrastructure.
- Whether or not the weakness exists in default configurations or installations.
- Whether there are any preconditions that need to exist before the vulnerability can be exploited, such as the compromise of other systems or security controls.
- Whether the affected asset is responsible for monitoring or protecting other assets.
- Whether the attacker needs to lure victims to a hostile server in order to exploit a vulnerability.

5.1. Existing security controls.

Security measures that are already in place significantly affect both an enterprise's susceptibility (resistance) to a vulnerability, and the severity of damage it causes.

- *Preventative controls*. These controls are employed to inhibit attacks and prevent harmful events from reaching their destination. Firewalls, anti-virus scanners, code reviews, encryption techniques, fences, door locks, and so on are all forms of preventative controls.
- *Detective controls*. Detective controls are employed to discover attacks. By the time these controls are used, an attack or event has already occurred. These controls must be capable of reacting very quickly to prevent loss or damage. Technical examples are intrusion detection systems (IDS)—either host-based or network-based, audit trails, and so on. Physical detective controls would include tripwires, and IR or motion sensors. An administrative control would be a policy dictating mandatory job rotation and job vacation.
- *Corrective controls*. A potentially harmful event has occurred, and the detective controls have recognized it. Now the corrective controls are employed to mitigate the impact or loss due to the event. Intrusion prevention systems (IPS), and auto-restore features (as found in Windows XP, for example) are some examples of technical corrective controls. Physical controls

would include doors or gates that lock automatically, trapping any intruders, fire suppressant systems, and security alarms.

- *Recovery controls*. These controls are designed to recover from the loss or damage incurred by the event. Backups, disaster recovery (DR) and business continuity plans (BCP) are examples of recovery controls.

Note that deterrent controls are not included, as they assist in reducing the threat or probability of an incident.

Example Resolved

After applying a sequence of THREAT ASSESSMENT (113) (providing the threat action frequencies) and VULNERABILITY ASSESSMENT (125) patterns, the museum has identified the vulnerabilities to information and physical assets shown in Tables 6.17 and 6.18 respectively. The threat action frequency values of both tables are taken from THREAT ASSESSMENT (113).

Table 6.17 Threat-Vulnerabilities table for information assets

THREAT ACTION (FREQUENCY)	VULNERABILITY (SEVERITY)
Natural	
Electrical spike in computer room (3)	Lack of surge protection, uninterruptible power system (UPS) (4)
Loss of electronic documents (3)	Incomplete or corrupt data backups (4)
Professional criminals	
Theft of information assets (3)	Susceptibility of employees to bribery (3)
	Lack of proper physical controls for document storage (locks, safe) (4)
Employees	
Unauthorized access of informational assets (5)	Weak information security controls enabling unauthorized access (3)
Data entry errors (5)	Lack of data validation during form input (2)
Leaking confidential information (3)	Exposure of information assets (3)

Table 6.18 Threat-vulnerability table for physical assets

THREAT ACTION (FREQUENCY)	VULNERABILITY (SEVERITY)
Natural	
Museum fire (3)	Failure of fire alarm system (6) Failure of fire suppression system (5)
Fatigue of support fixtures, building structural failure (3)	Lack of regularly scheduled inspections (4)
Failure of monitoring and alarm systems (4)	Lack of regularly scheduled inspections (4)
Professional criminals	
Theft of museum collections and exhibits (2)	Lack of regular alarm testing procedures (3) Lack of adequate storage and protection of physical assets (3)
Physical attack against employees (3)	Lack of security training for employees (4)
Employees	
Accidental damage to museum collections and exhibits (4)	Carelessness of employees when handling/cleaning exhibits (2)
Accidental damage to vehicles (4)	Carelessness of employees while driving vehicles(2) Lack of regularly scheduled maintenance checks (4) Lack of adequate employee background checks (4)
Theft of museum collections and exhibits (2)	Lack of regular alarm testing procedures (3) Lack of adequate storage and protection of physical assets (3) Susceptibility of employees to bribery (4)
Misconfiguration of monitoring and alarm systems (4)	Lack of regular alarm testing procedures (3)
Museum patrons	
Accidental damage to museum collections and exhibits (3)	Carelessness of museum patrons when viewing exhibits (2)

Variants

The SANS Institute and the CERT Coordination Center at Carnegie Mellon are two renowned information security centers. They provide vulnerability lists and databases of common vulnerabilities. [CERTb] uses a purely quantitative scale of 0 to 180 to rank the severity of a vulnerability., whereas [SANSe] uses the following qualitative scheme:

- *Critical* vulnerabilities are those where essentially all planets align in favor of the attacker. These vulnerabilities typically affect default installations of very widely-deployed software, result in root compromise of servers or infrastructure devices, and the information required for exploitation (such as example exploit code) is widely available to attackers.
- *High* vulnerabilities are usually issues that have the potential to become critical, but have one or a few mitigating factors that make exploitation less attractive to attackers.
- *Moderate* vulnerabilities are those where the scales are slightly tipped in favor of the potential victim. Exploits that require an attacker to reside on the same local network as their victim, or only affect non-standard configurations or obscure applications, are likely to be rated moderate.
- *Low* vulnerabilities usually do not affect most administrators, and exploitation is largely unattractive to attackers. Often these issues require the attacker to have some level of access to a target already, require elaborate specialized attack scenarios, and only result in limited damage to a target.

[NIST800-30] uses the following definitions for vulnerability severity:

- *High*. Exercise of the vulnerability (1) may result in the highly-costly loss of major tangible assets or resources, (2) may significantly violate, harm, or impede an organization's mission, reputation, or interest, or (3) may result in human death or serious injury.
- *Medium*. Exercise of the vulnerability, (1) may result in the costly loss of tangible assets or resources, (2) may violate, harm, or impede an organization's mission, reputation, or interest, or (3) may result in human injury.
- *Low*. Exercise of the vulnerability (1) may result in the loss of some tangible assets or resources, (2) may noticeably affect an organization's mission, reputation, or interests.

The Common Vulnerability Scoring System [CVSS] is an open framework that can be used by any security or application vendor to determine the overall severity posed by a vulnerability. Three categories of metrics are scored and combine to produce a final score.

- The base metric represents the properties of a vulnerability that do not change over time, such as access complexity, access vector, degree to which the vulnerability compromises the confidentiality, integrity and availability of the system, and requirement for authentication to the system.
- The temporal metric measures the properties that do change over time, such as the existence of an official patch or functional exploit code, and the level of effort to remedy the vulnerability.
- The environmental metric measures the properties of a vulnerability that are representative of users' IT environment, such as prevalence of the affected system and overall potential loss.

Known Uses

A vulnerability assessment is a key component of all widely accepted risk assessments, including those from [NIST800-30], [ISO13335-3], [Pel01], and others. While they differ slightly in their approach, the purposes and overall goals are consistent.

Consequences

This pattern has the following benefits:

- An enterprise obtains a list of all vulnerabilities that could impact their systems, some which may have been previously unknown.
- The enterprise is able to rank the vulnerabilities according to severity and potential impact.
- An enterprise is able to recognize which vulnerabilities can be discounted where there are no accompanying threats.

It also has the following liabilities:

- A thorough vulnerability scan involves the coordination of many departments and may be difficult to initiate if these departments are not in cooperation.
- This pattern cannot be used in isolation to patch or eliminate vulnerabilities. The results of VULNERABILITY ASSESSMENT (125) should be returned to the RISK DETERMINATION (137) pattern, where the final risk can be determined and an appropriate control implemented.