Section 2: Week 6: Design a Vulnerability Assessment Procedure

Nate Bachmeier

TIM-7030: Managing Risk, Security, and Privacy

June 12, 2020

North Central University

# Design a Vulnerability Assessment Procedure

NCU Financial (NCU-F) operates on a finite budget and must prioritize investments into features and services. On the one hand, the business would like to spend all available resources delivering its core mission, enabling personalized investing. Allocating time and money into other projects might even appear to detract from this mission and feel like a waste. However, along the journey are risks from a litany of sources that can derail progress, cause financial hardship, and harm the organization’s reputation (Erickson & Neilson, 2018). Using a security framework, such as COBIT or NIST Cybersecurity, formalize processes for identifying and approaching threats from these risks (Devos & Van de Ginste, 2015). It would be economically prohibitive to remove every threat against the organization. Instead, a prioritization discussion must delineate between threats and vulnerabilities.

A vulnerability occurs at the intersection of (1) system susceptibility; (2) threat accessibility; and (3) threat capability (Baskerville, Rowe, & Wolff, 2018). Nullifying any of these predicates mitigates an attacker’s ability to compromise the confidentiality, integrity, and availability from a specific threat. The costs to address these predicates range substantially and are highly scenario specific. For instance, the legacy main-frame lacks support for modern network encryption and authorization protocols. Upgrading or replacing the system are not feasible solutions, though moving the server to a private network disconnects the threat’s accessibility. Another configuration, such as a public web application, might experience the opposite problem where patching the software defect is a more natural path forward. Over a long enough period, all vulnerabilities are discovered and exploited (McLane, 2018). Processes need to exist to defuse these issues before they explode, with combinations of proactive and reactive defenses.

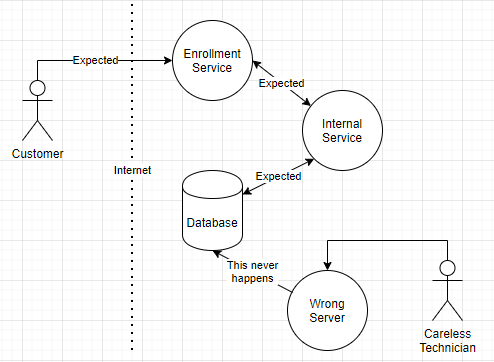
# Section I: Security Information and Event Management (SIEM)

Modern business topologies are dynamic and interconnected, containing components that originate from internal teams, external contractors, and third-party providers. Overtime priorities shift and follow economic incentives to churn out new products and features, causing bitrot to neglected services and new features that lack sufficient security controls (Li & Liao, 2018). Eventually, service failures occur within this complex environment leading to the natural question, “so what happened?” The cost associated with solving this mystery is dependent on the quality of the auditing information. Generally speaking, half of these moments come from employee negligence, a quarter from system errors, and the remainder from malicious sources (Valiente, 2017). According to this breakdown, there is significant value is auditing all change across every business process. However, a trade-off exists between performance, storage, and observability, which might limit NCU-F’s ability to collect and persist such an enormous volume of data (Adedayo & Oliver, 2015). When choosing what information to keep, a one-size-fits-all solution does not exist. Instead, the administrators need to categorize the potential value of these various events in terms of needs for experimental and retrospective reconstruction.

Audit events are paint dots that make up a larger picture. Process mining forms this picture by clustering related messages into traces, operations, and transactions. Despite extensive research, completely automating this process is challenging in real-world settings (Claes & Poels, 2014). Consider the variability in technology stacks and how this impacts design decisions of event schemas, data formats, and encodings. Strategies can improve the probabilities of success, such as including relational (e.g., request-id), functional (e.g., scenario-name), and temporal state (e.g., timestamps) information (Adedayo & Oliver, 2015). Along with these properties, other common aspects like the caller’s context can provide additional insights.

A data curation process needs to exist for resolving these discrepancies and transforming the raw data into information. Undergoing such a transformation requires event collection, transformation, aggregation, and correlation capabilities to present a holistic semantic model of the business. Next, administrators need functionality to author compliance and remediation policies that monitor changes to the ecosystem. Many enterprises choose to purchase holistic solutions like System Information and Event Management (SIEM) platforms, instead of building custom code or gluing third-party components together.

Figure 1: Enrollment App



While there are many benefits to having a formal SIEM product, it is not a magic box and only provides insights into integrated systems. For instance, NCU-F exposes a public enrollment portal that follows a standard three-tier architecture (see Figure 1). If monitoring exists only for the database, then it can be perplexing to investigate the reason behind specific alerts. Perhaps a careless technician is servicing a request against the wrong server. Alternatively, the inclusion of router and switch logs could detect this traffic anomaly. Unfortunately, SIEM providers often follow the “more for more” mantra and charge higher licensing fees for additional coverage. The support of different technology stacks can also depend on the focus areas of the SIEM platform (see Table 1). Before choosing a provider, the organization needs to assess the most concerning scenarios and acceptable costs. For example, a simple branch office that already uses McAfee anti-virus will likely find McAfee Security Manager a great fit.

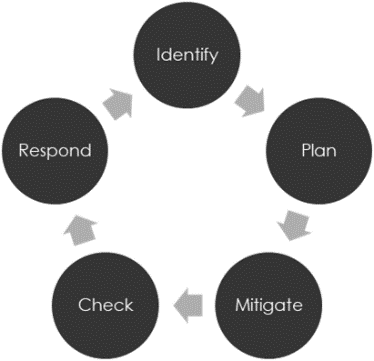
Table 1: SIEM Products

|  |  |  |
| --- | --- | --- |
| Provider | Pros | Cons |
| SolarWinds Security  Event Manager (Tek-Tools, 2020) (SolarWinds, 2020) | * Simple, easy to use * Built-in compliance Reporting * Covers the branch office scenario | * Intended for smaller environments |
| Splunk (Splunk, 2020) | * Advanced real-time monitoring * Over 200 integrations * Gartner leader | * Large hardware footprint |
| Sumo Logic (Sumo Logic, 2020) | * Hosted in cloud * Monitors SaaS, IaaS, and PaaS | * Limited support for non-server infrastructure |
| McAfee Enterprise Security Manager (McAfee, 2020) | * Rich signatures reduce false positives * Integrates with existing McAfee systems * Great coverage of desktop environments | * Traditional infrastructure and desktops only |
| Rapid7 InsightIDR  (Rapid7, 2020) | * Hosted in cloud * A rich collection of signatures * Gartner leader | * 500 asset minimum |

# Section II: Vulnerability Assessment

Vulnerability assessment strategies follow a feedback loop that identifies an issue and then carries through a remediation process (see Figure 2) (Radhakrishnan, 2015). Remediating the defect requires planning acceptable mitigations, delivering those changes, validating the fix is sufficient, and communicating status to stakeholders.

Figure 2: Vulnerability Assessment



## Identification

Detection of the vulnerability might come from static (offline) or dynamic (online) analysis. Static Analysis Tools (SAT), such as SonarQube and Checkmarx, parse the source code into graph-like structures and then run queries to find defects in the application. While there is the potential of encountering false positives and false negatives, these lint checks are effective at catching problems in proprietary systems. Dynamic Analysis Tools (DAT) use telemetry to monitor for security-critical events, making it ideal for black-box situations and discovering unknown unknowns. Outside of tooling, periodic process audits can surface that a problem exists.

## Plan

The full impact of a vulnerability might not be immediately visible, as engineering teams often duplicate existing code and infrastructure. During the planning phase, the senior leaders need to agree on a communication strategy and proposal for introducing the new change. Consider a scenario where insufficient controls exist for a shared database. Arbitrarily adding defenses will break downstream processes and cause a production outage. Similar to other software projects, an iterative design must occur that seeks the maximum immediate value (Lam, 2016). Though not ideal, the security team needs to weigh the potential schedule risk that comes from doing everything upfront. Perhaps that means only protecting against one of N situations initially, and returning to the others later.

## Mitigation and Verification

Mitigating the vulnerability could be a trivial change to a configuration file or require massive changes to the infrastructure. These changes present multiple risks to the organization, such as degrading performance (e.g., encryption or verbose logging) or creating new failure points (e.g., broken code). In parallel to standard regression automation, the quality assurance teams need to confirm other permutations of the exploit are unsuccessful. For example, a cross-site scripting defect existing in the enrollment portal (see Figure 1) via the query string. The mitigation checks also need to review other parameters to that page.

## Respond

Many factors influence NCU-F’s requirement to respond publicly about the vulnerability, such as legal and compliance requirements (Fonseca & Ramaswamy, 2014). Where regulatory mandates do not exist, the company must weigh the ethical obligation to customers and stakeholders to be fully transparent. If the business attempts to cover up the mistake, only to details leaked to the media, removes substantial trust and might irrefutably harm its public image. Given the numerous landmines that may exist, the security team should involve members of senior leadership and other stakeholders (e.g., general counsel or public relations).

## Value of Being Proactive

Fundamentally a software vulnerability is like any other application defect. The sooner the issue is found, the cheaper its resolution will be. For example, investing in static analysis and other lint checks at build time can prevent the defect from ever leaving the developer’s private workstation. Meanwhile, reacting to a media crisis adds time pressure and increases employee stress.