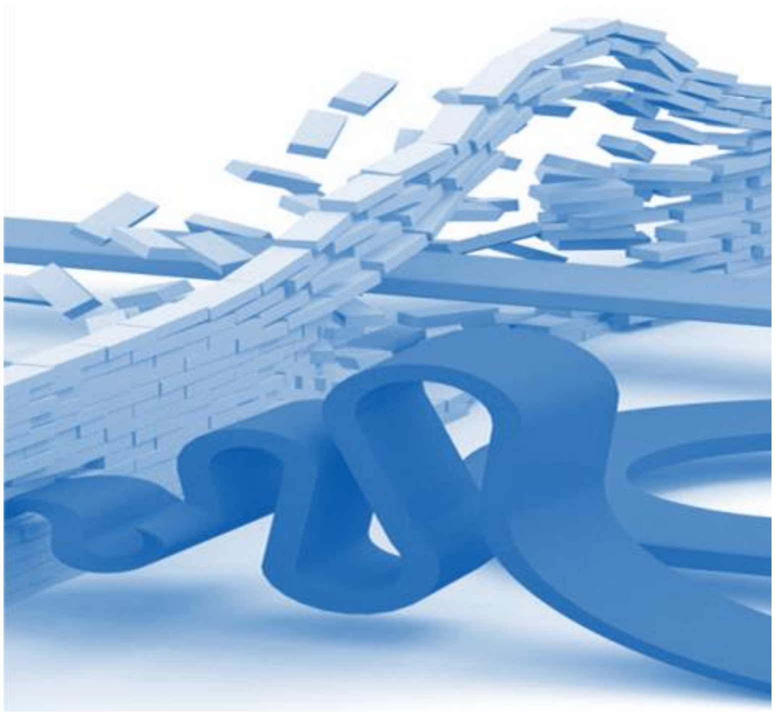


# SECURITY TESTING

Prof. Dr. Bernhard Tellenbach

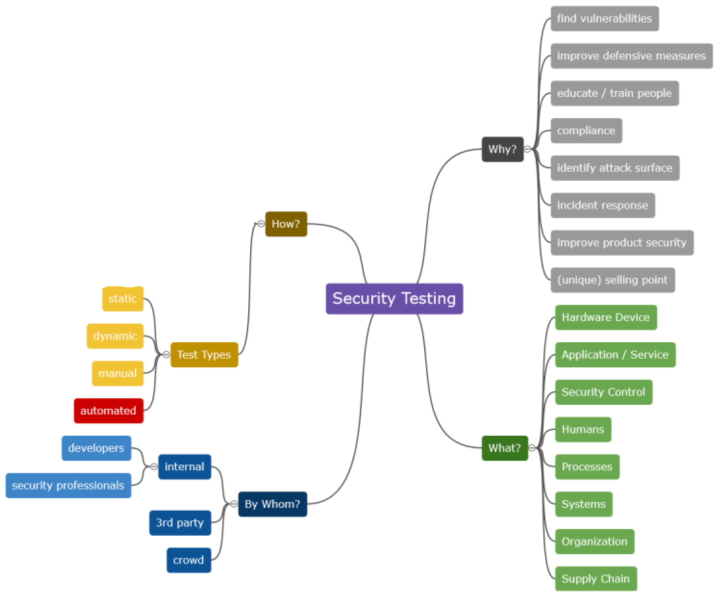
## Content

- Security Testing Methods
  - Vulnerability Scanning
  - Classical Penetration Testing
  - Red Teaming / Ethical Hacking
  - Purple Teaming
  - Breach & Attack Simulation
  - Bug Bounty Programs
- Penetration Testing
  - Penetration testing methodologies
  - Penetration testing phases
  - The pre-engagement phase

## Goals

- You can name six different security testing methods can discuss what method is best when given the motivation for a security test
- You know what a penetration test is and you can explain what it can be used for
- You can name at least two standards that provide guidance on how to do a penetration test
- You can explain the role and important parameters (scope, rules of engagement, test method) of the pre-engagement phase

## The Big Picture

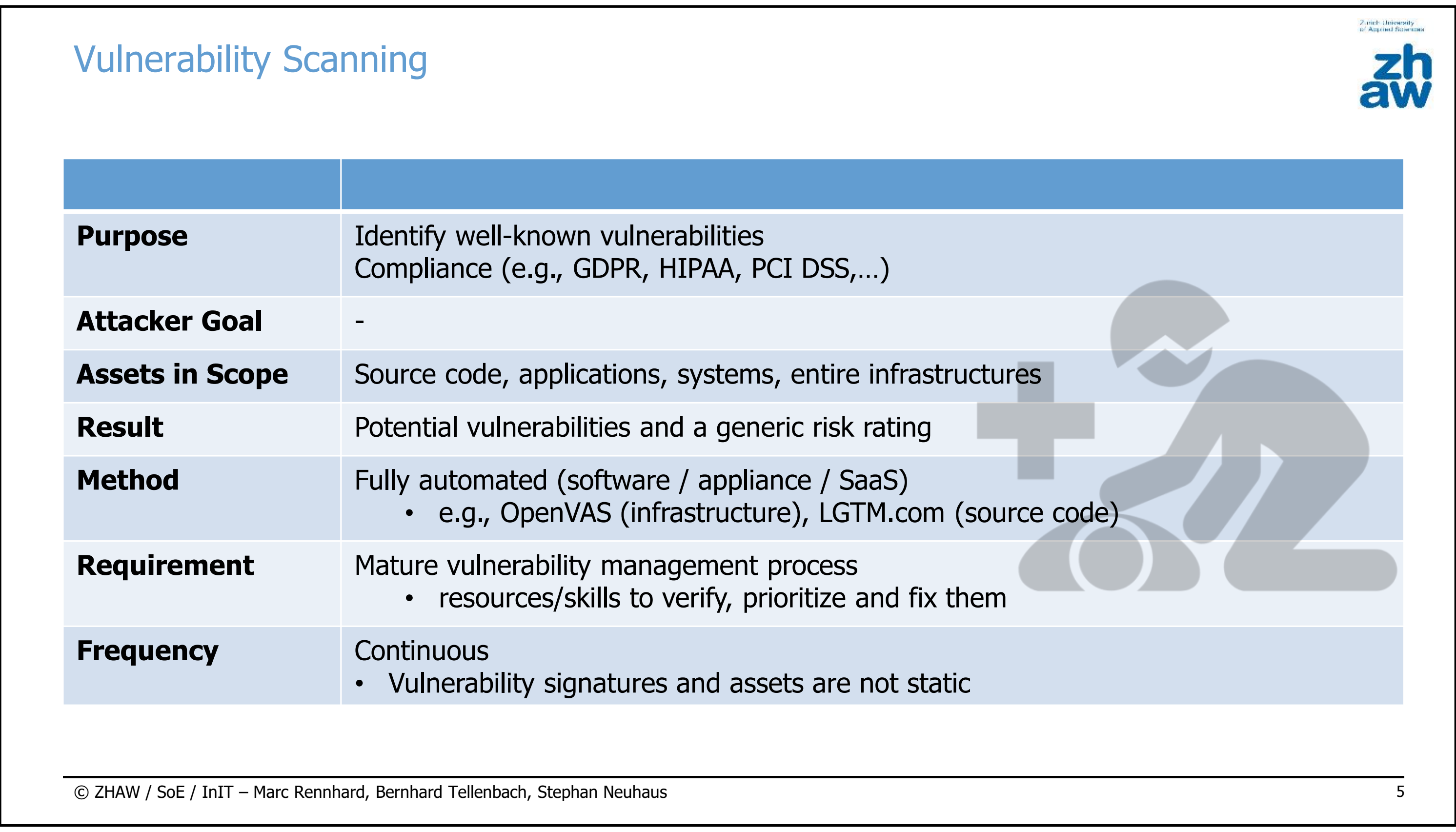


When talking about security testing, there are many aspect to consider when deciding how to do the testing. Some of these aspects are:

- **Why** do we want to test? **What's** our goal? Is it to **find** (and hopefully fix) **vulnerabilities**? Or is it to test the **incident response** processes and capabilities?
- **What** do we want to test? A **single service** or application, a system or an **entire organization**?
- **Who** should do the testing? Members of your **staff**, a **3rd party** or maybe even the **crowd**?
- Shall it be an **automated test** or shall there be **a human** in the loop?
- Do we want to test a **live system** or its **source code**?

To cover the different needs, different security testing methods have been developed. Among the more prominent ones are:

- Vulnerability Scanning
- Classical Penetration Testing
- Red Teaming / Ethical Hacking
- Purple Teaming
- Breach & Attack Simulation
- Bug Bounty Programs



## Vulnerability Scanning

| | |
|---|---|
| **Purpose** | Identify well-known vulnerabilities<br>Compliance (e.g., GDPR, HIPAA, PCI DSS,…) |
| **Attacker Goal** | - |
| **Assets in Scope** | Source code, applications, systems, entire infrastructures |
| **Result** | Potential vulnerabilities and a generic risk rating |
| **Method** | Fully automated (software / appliance / SaaS)<br>• e.g., OpenVAS (infrastructure), LGTM.com (source code) |
| **Requirement** | Mature vulnerability management process<br>• resources/skills to verify, prioritize and fix them |
| **Frequency** | Continuous<br>• Vulnerability signatures and assets are not static |

The first testing method is a very basic one: vulnerability scanning. Each organization should nowadays make use of this method and scan their systems and source code (if applicable) on a regular basis.

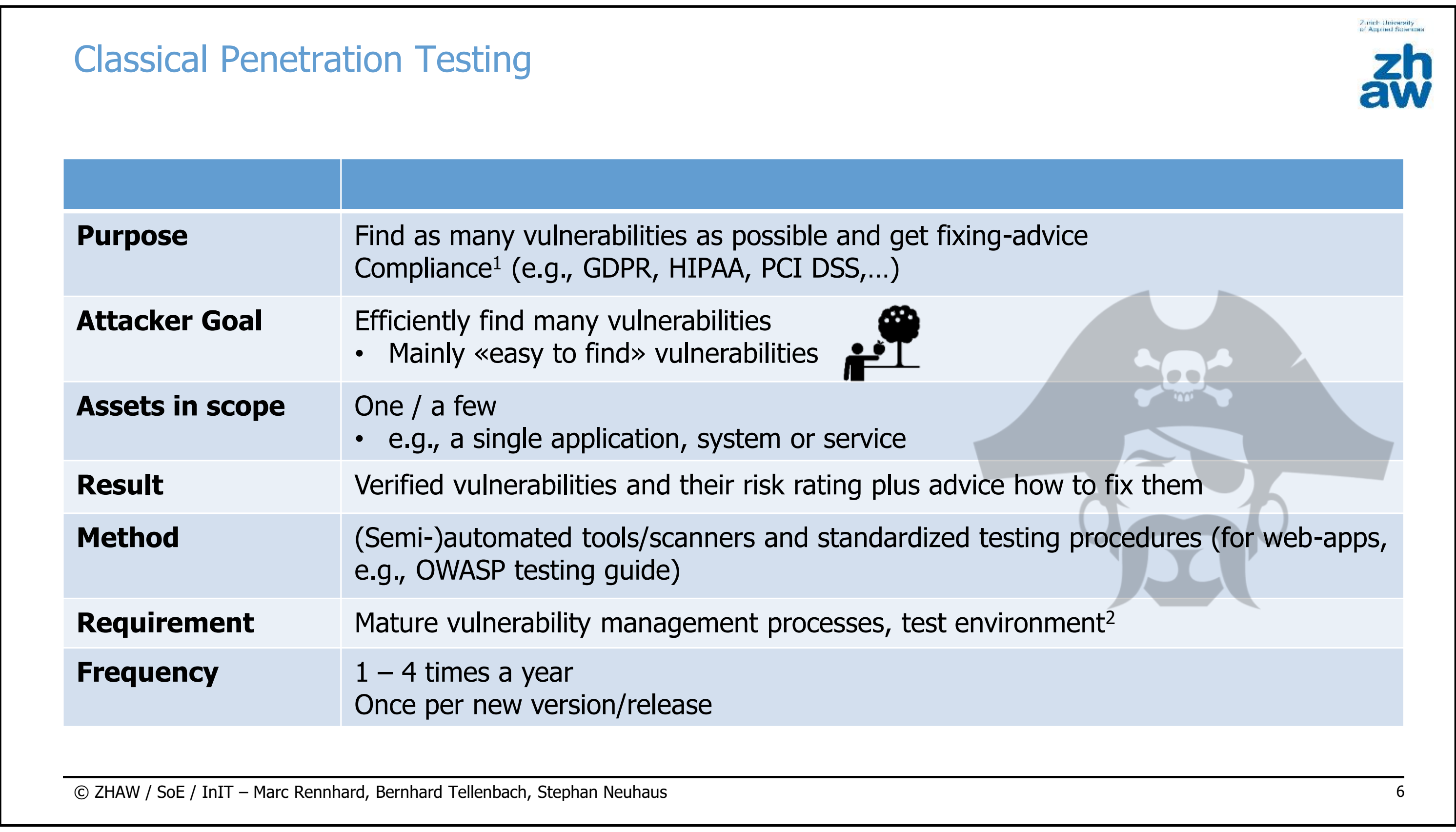
Vulnerability scanning is like a «**family doctor**» – everyone should have one to go to for regular medical checkups. Hopefully, the doctor identifies common and well-known health-issues before they cause harm.

Vulnerability scanning can be applied to all kinds of assets – from source code to entire IT infrastructures. Since it is usually done in a fully automated way and without executing potentially harmful attacks, we can say it is “attacker-free” – there is no simulated attacker pursuing a specific goal.

What you get as a result from a vulnerability scan is a list of potential vulnerabilities along with their generic risk rating. What is left to the user is the verification of these vulnerabilities and the mapping to a scenario-specific risk score and ultimately a business risk.

This implies that you should have a mature vulnerability management program in place that can do this and fix vulnerabilities fast, so that they don’t pile up.

The same applies to the question “how frequently should you scan”? Well, whenever possible, you should do it in a continuous fashion - new vulnerability signatures might be added to the scanner at anytime and assets might be subject to change.



## Classical Penetration Testing

| | |
|---|---|
| **Purpose** | Find as many vulnerabilities as possible and get fixing-advice<br>Compliance[1] (e.g., GDPR, HIPAA, PCI DSS,…) |
| **Attacker Goal** | Efficiently find many vulnerabilities<br>• Mainly «easy to find» vulnerabilities |
| **Assets in scope** | One / a few<br>• e.g., a single application, system or service |
| **Result** | Verified vulnerabilities and their risk rating plus advice how to fix them |
| **Method** | (Semi-)automated tools/scanners and standardized testing procedures (for web-apps, e.g., OWASP testing guide) |
| **Requirement** | Mature vulnerability management processes, test environment[2] |
| **Frequency** | 1 – 4 times a year<br>Once per new version/release |

© ZHAW / SoE / InIT – Marc Rennhard, Bernhard Tellenbach, Stephan Neuhaus 6

Another security testing method that is quite popular is classical penetration testing. Depending on what the attacker knows about the target and what the target knows about the attack, there are different pentest-styles (see later).

However, common to most styles is that the attacker behave like pirates - they want to find as many treasure ships (the vulnerabilities) in their search space (or: in scope) as possible. To do so, they focus on vulnerabilities that are easy to spot. Don't expect the testers to find vulnerabilities that are well hidden (e.g., by an insider) or require a complex interaction of multiple components. To find vulnerabilities efficiently, classical penetration tests are loud: they fire all their tools and scanners and go through standardized testing procedures.

The reason for this strategy is that they understand that the only thing that you might get when they don't find any vulnerability is …. well, the feeling that there are none. If you should be tempted to trust such feelings: just don't. ☺

Pirates understand that the only true added value for their contractor is the treasure ships they find! But the list with the vulnerabilities found is only one part of it. The true value of a pentest is that the vulnerabilities listed contain true vulnerabilities only, no false positives and that the vulnerabilities come with a custom risk rating and advice how to fix them. As for the frequency of such tests, since pentesting involves manual work, it can be quite expensive and can't be done very often. A reasonable rule of thumb might be to test crititical applications and systems that are exposed to untrusted parties up to four times a year or at least once per new version/release.

[1] Pentesting and compliance:

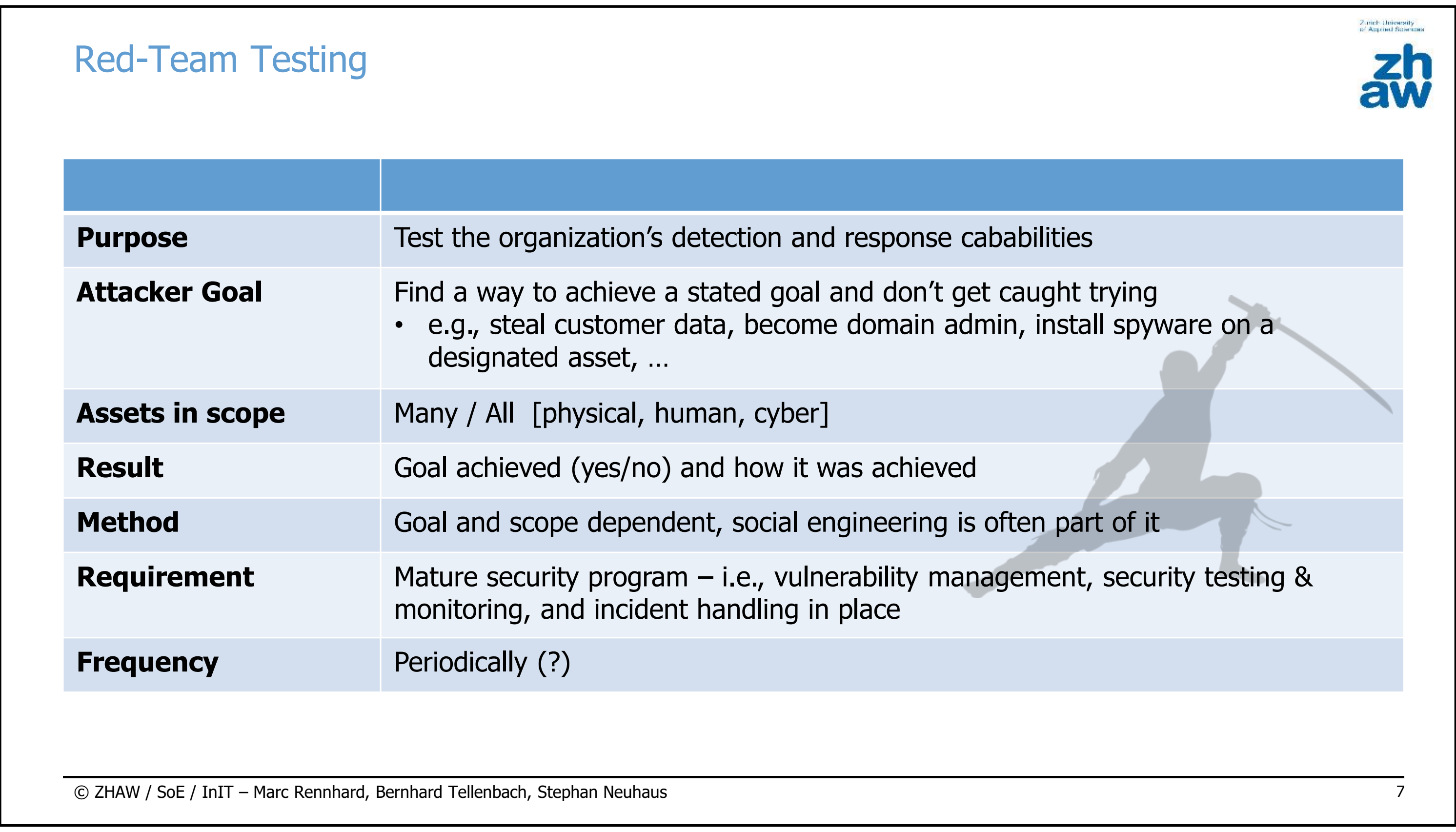
- HIPAA Evaluation Standard § 164.308(a)(8) specifically speaks to the safety, privacy, and electronic exchange of medical information. Pentesting is not outright required, but it's one of the limited number of ways to satisfy some of the requirement. Regardless of the evaluation performed, healthcare providers must regularly test data security or face fines ranging from \$100 to \$50,000 per record compromised.
- Payment Card Industry Data Security Standard (PCI DSS) serves as the information security standard for organizations that handle branded credit cards, including Visa, Mastercard,

American Express, and Discover. One of PCI DSS' 12 requirements is regular network monitoring and testing. As part of that requirement, the standards differentiate between vulnerability scans and penetration tests, though it requires both. To maintain penetration testing compliance, companies must complete a pen test at least once every six months, although many experts believe a quarterly test is more congruent with actual needs.

- GDPR - Article 32 of the Regulation requires organizations to implement technical measures to ensure data security. It outlines specific measures and highlights the need for "[A] process for regularly testing, assessing and evaluating the effectiveness of technical and organizational measures for ensuring the security of the processing". Defects in web servers, web browsers, email clients, point-of-sale (POS) software, operating systems and server interfaces can allow attackers to gain access to an environment. However, to patch these vulnerabilities, you need to identify them first.

[2] Pentesting and test environments: Whether you should do a pentest on the production system or a test system depends on several aspects:

- Risk appetite: A pentest on the production infrastructure can damage it and impact on your business. Hence, you should only consider this if one of the two following points apply.
- Availability of a test system: Actually, in theory there should always be a test system, but in practice, that might look differently…
- Equivalence of the test- and production system: The test-system might differ in many aspects from the production system (e.g., protective layers, interfaces to third-party services are only emulated, (security-relevant) configurations, …). These differences might be relevant for what you want to test.



## Red-Team Testing

| | |
|---|---|
| **Purpose** | Test the organization's detection and response cababilities |
| **Attacker Goal** | Find a way to achieve a stated goal and don't get caught trying<br>• e.g., steal customer data, become domain admin, install spyware on a designated asset, … |
| **Assets in scope** | Many / All [physical, human, cyber] |
| **Result** | Goal achieved (yes/no) and how it was achieved |
| **Method** | Goal and scope dependent, social engineering is often part of it |
| **Requirement** | Mature security program – i.e., vulnerability management, security testing & monitoring, and incident handling in place |
| **Frequency** | Periodically (?) |

Red-Team testing is a special form of pentesting where there are quite some misconceptions and misunderstandings around. Red-Teamining, is not very well-suited for improving your overall security at large. For this purpose, vulnerability scanning, and classical penetration testing provide much more value.

If you do Red-Teaming, your detection & response, including awareness programs for your employees, should be in place and mature. The reason why these things should be there is that Red-Teaming is mainly about testing the resilience of your organization against a «real attacker». Real means several things.

- First, that the attacker pursues a very specific goal set by you. For example, become domain admin.
- Second, the timing and goal of the testing is not shared with your blue team.
- Third, the attacker might use any assets and attack vectors.

Highly successful attack strategies like social engineering or bringing in physical spying devices are often part of the attack path. Red teams do not try out many different attack paths but focus on the most promising one. Depending on the amount of information the team is given, intelligence gathering can be an important part of an assignment. Because not many attack vectors/paths are used, this method won't help much in improving your security posture at large.

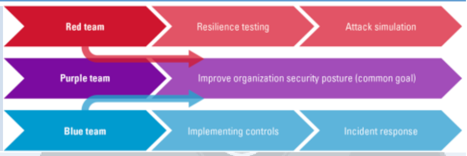
Bit if your defenses are good (or the attackers are sloppy) you might eventually detect the security breach (for example lateral movement activities) and learn whether your incident response works as expected.

As for the frequency – this is a difficult one here since the costs might be high and what you gain in terms of security is not always that clear.

So, red-teaming is a perfect fit if your main motivation for doing it is to have proof that an attacker can get in, for example to help justify parts of the budget for security.

But for other goals like testing and strengthening your defenses (your blue team), it is probably better to spend the money for purple teaming.

## Purple-Team Testing

| | |
|---|---|
| **Purpose** | Improve security posture<br>• focus on detective/preventive controls<br>• identify malicious actors in environment<br> Red team<br>Resilience testing<br>Attack simulation<br>Purple team<br>Improve organization security posture (common goal)<br>Blue team<br>Implementing controls<br>Incident response |
| **Attacker Goal** | Help the blue[1] team to catch them / learn from them |
| **Assets in scope** | Predetermined systems/employees |
| **Result** | Improvements to security controls (e.g., new detection rules) and plan to resolve issues that could not be addressed during the exercise |
| **Method** | Simulate (relevant) attack patterns and scenarios |
| **Requirement** | Interfaces and collaboration between incident response, security tooling, network engineers and vulnerability management work well |
| **Frequency** | Periodically (e.g., once per quarter) |

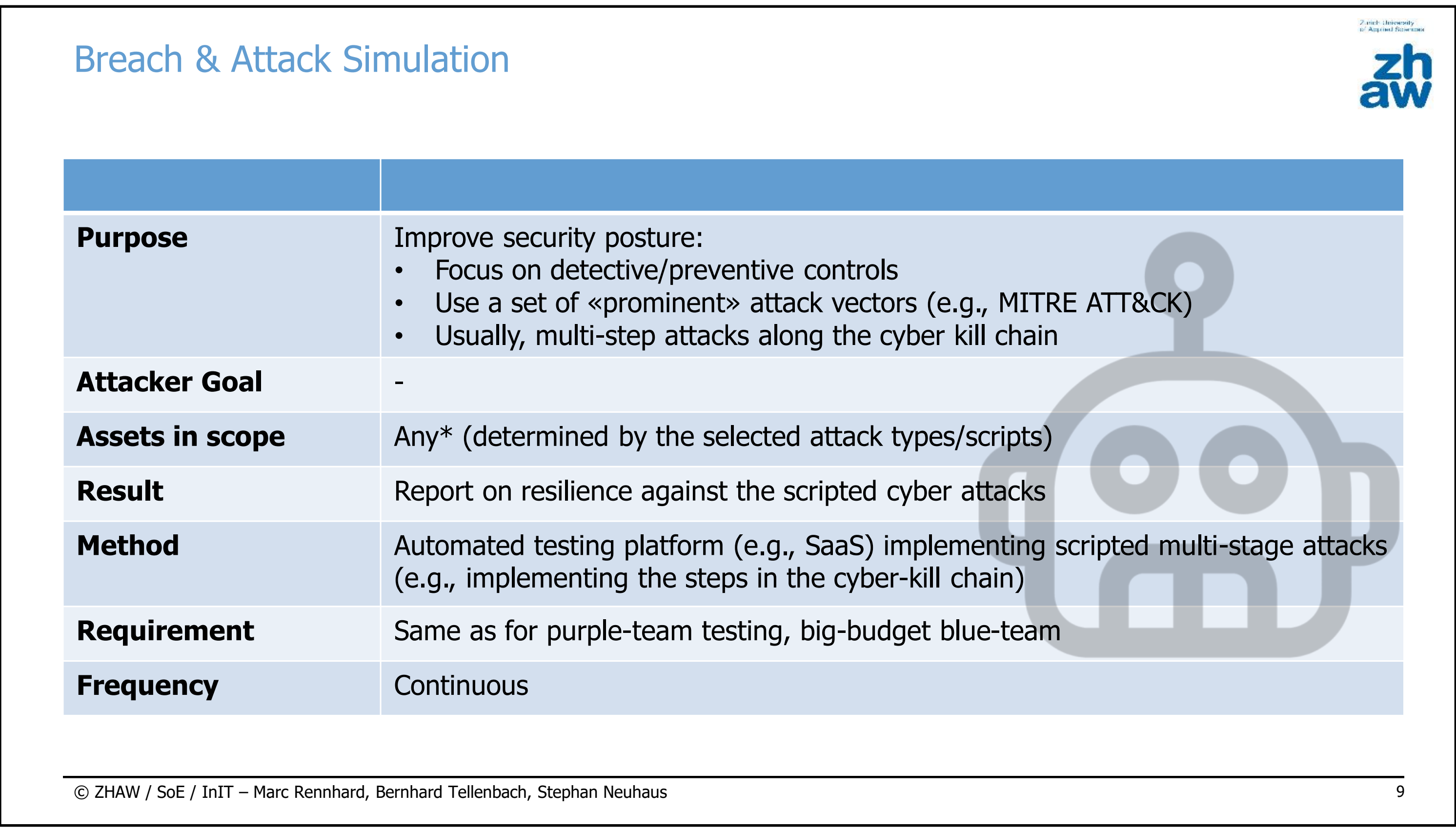
Red-teaming and purple-teaming both focus on testing the detective and responsive controls. However, purple teaming puts a much stronger focus on improving those controls.

In a red-teaming exercise, the red and blue teams start interacting with each other only after the cyberattack simulation has been completed. For example, the blue team will indicate which indicators of compromise they have found, and the red team will provide a detailed storyline of all actions performed.

In contrast, a purple team exercise brings the traditionally segregated red and blue teams together for an exercise where they work together. The format can vary from sitting in a room together and going through attack behaviors, to a red team conducting an exercise with the awareness of the blue team and a red team representative helping and giving tips to the blue team. This can remove the feeling of competitiveness that usually exists between red- and blue-teams and which might harm the end goal.

Ideally, such exercises are conducted on a regular basis to account for new attack vectors and changes to the environment. Since purple-teaming is focusing on detective and preventive controls, their maturity level should be quite high and the communication and decision paths between relevant entities should work well. However, even if this is not yet the case, a purple team exercise could be an option to gain immediate value and help develop long-term action plans.

[1] More about the different “colors” of teams in information security can be found in the article on the BAD (build, attack, defend) pyramid: https://danielmiessler.com/study/red-blue-purple-teams/



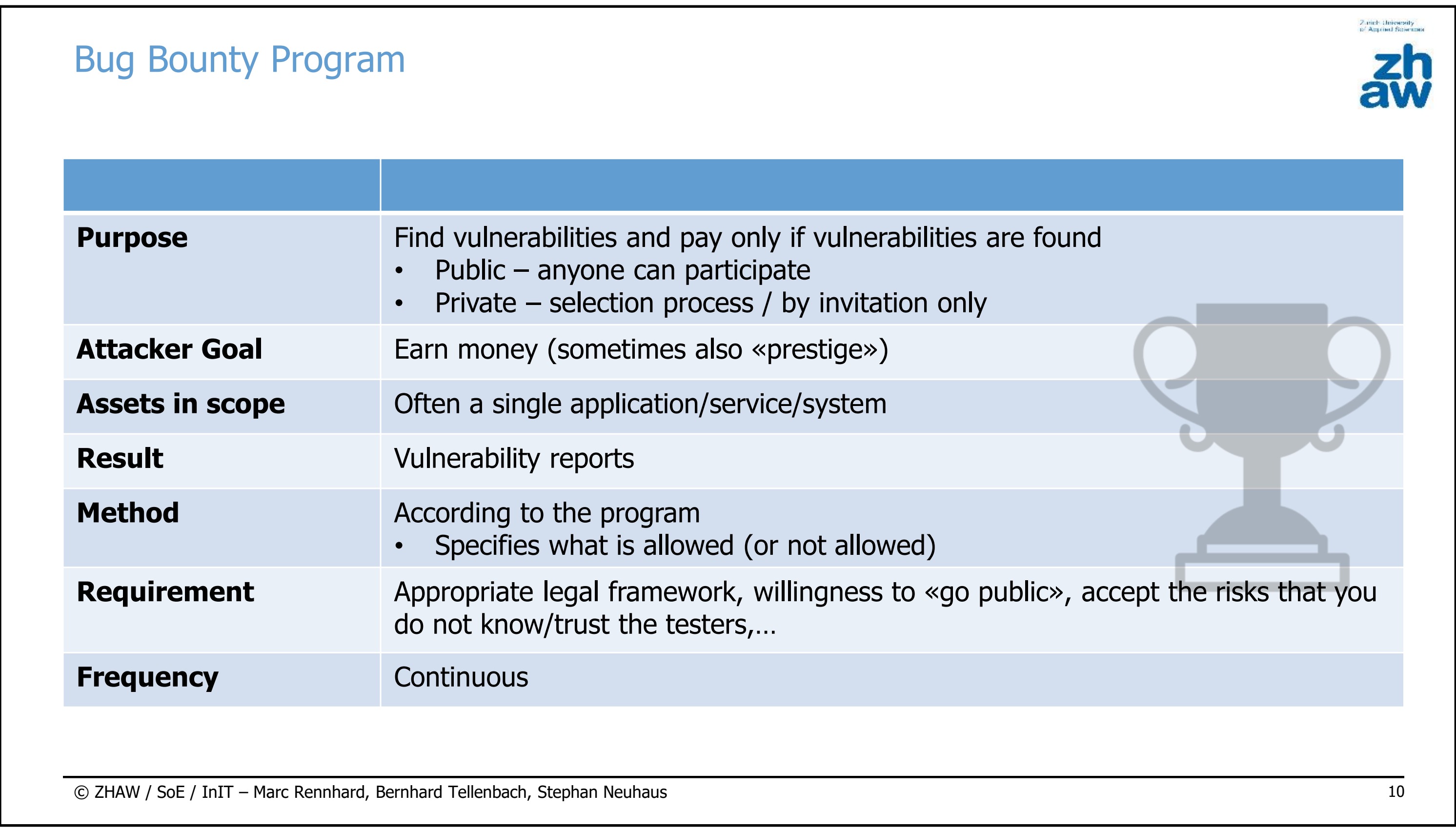
## Breach & Attack Simulation

| | |
|---|---|
| **Purpose** | Improve security posture:<br>• Focus on detective/preventive controls<br>• Use a set of «prominent» attack vectors (e.g., MITRE ATT&CK)<br>• Usually, multi-step attacks along the cyber kill chain |
| **Attacker Goal** | - |
| **Assets in scope** | Any* (determined by the selected attack types/scripts) |
| **Result** | Report on resilience against the scripted cyber attacks |
| **Method** | Automated testing platform (e.g., SaaS) implementing scripted multi-stage attacks (e.g., implementing the steps in the cyber-kill chain) |
| **Requirement** | Same as for purple-team testing, big-budget blue-team |
| **Frequency** | Continuous |

When testing with the «breach and attack simulation» method, one usually makes use of a platform that **replaces the attacker in a red-** or purple-teaming exercise. This platform is typically a Software-as-a-Service (SaaS) that comes with a significant number of **scripted attacks**. These attacks usually consist of **multiple steps along the cyber-kill chain**. For example, a first step might be a phishing attack on a set of employees to grab credentials, then the use of these credentials to authenticate to the phished service. And finally extracting data from this service.

While this looks like it is something that is easy and promising to do, to benefit from it, you need people with a very good understanding of the attacker and their methods and your defenses and tuning options. It is mainly a tool to **test your infrastructure vs. a set of standardized attacks** and see how well your defenses work. It is especially good, if you want to make sure that **when you change stuff in your defensive controls**, you can still detect the attacks that you could detect before the changes.

As for the frequency of testing: since it can be automated, you could go for a continuous approach. However, it is especially useful when new attack patterns become available, or you change something in your defenses.



## Bug Bounty Program

| | |
|---|---|
| **Purpose** | Find vulnerabilities and pay only if vulnerabilities are found<br>• Public – anyone can participate<br>• Private – selection process / by invitation only |
| **Attacker Goal** | Earn money (sometimes also «prestige») |
| **Assets in scope** | Often a single application/service/system |
| **Result** | Vulnerability reports |
| **Method** | According to the program<br>• Specifies what is allowed (or not allowed) |
| **Requirement** | Appropriate legal framework, willingness to «go public», accept the risks that you do not know/trust the testers,… |
| **Frequency** | Continuous |

This testing method bears many similarities with a penetration test where:

- the penetration testers are **replaced by the crowd**
- and where they **don't get paid or the work** but for vulnerabilities found

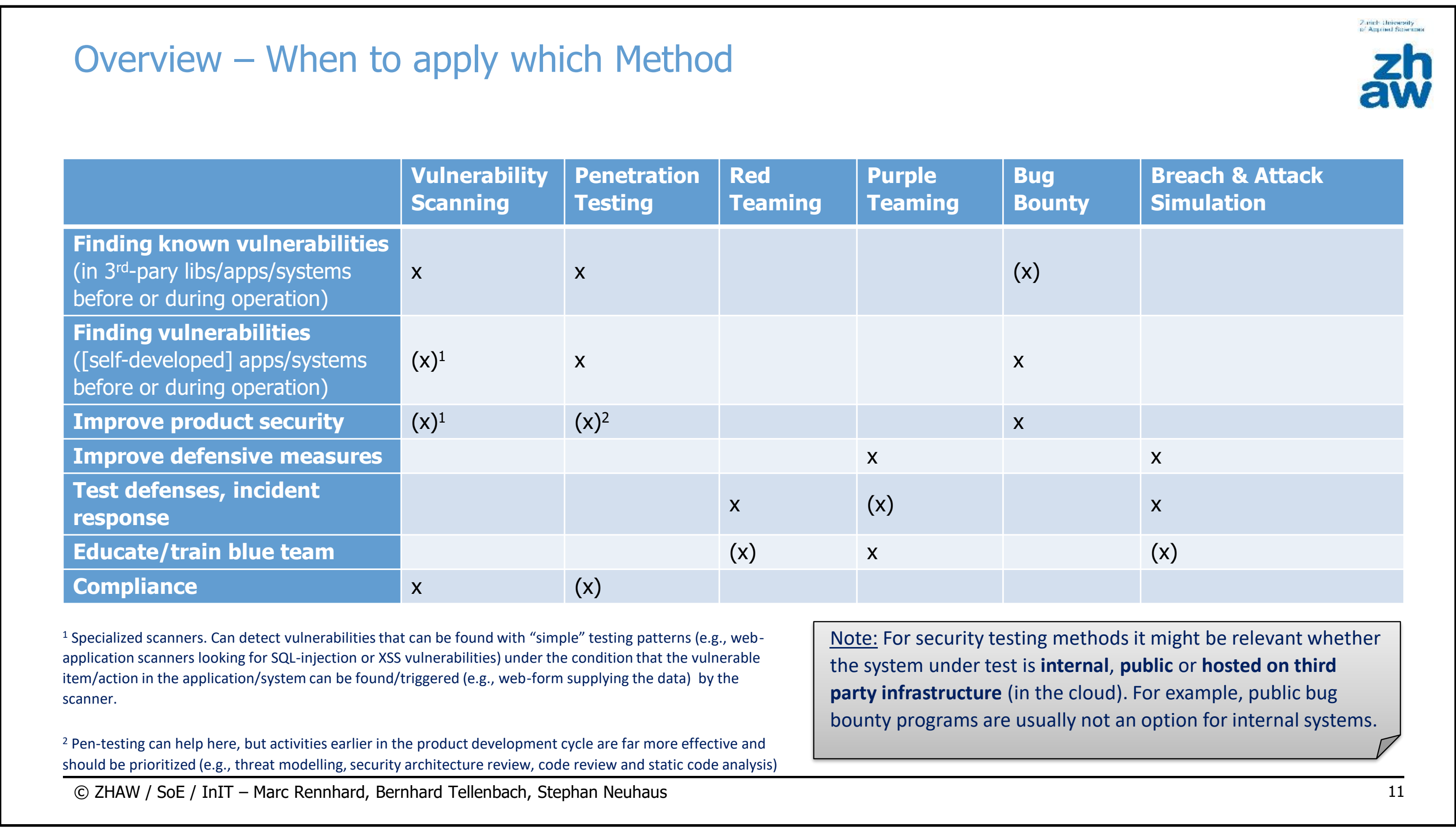
While this might sound interesting from a financial and security point of view, bug bounty programs should only be considered when you have already **a mature security program**.

If you do not have a good security posture already, it might get expensive fast. After all, it is very likely that there are quite many low hanging fruits (vulnerabilities) for the bounty hunters to grab.

Implementing a successful bug bounty program is a complex and resource intensive task:

- In most countries, some form of **contract/legal framework** between the bounty hunters and you is required to make their activities legal. At the same time, this framework should not enable malicious actors to get away with malicious activities.
- You can expect lots of reports coming in from bounty hunters – they expect a **quick and professional reaction** to their reports. After all, they invested quite some work in the hope of earning some money. If you don't react quickly and take all reports seriously, your program **is likely to be shamed** for this and see a **decrease in the number** of bounty hunters and the skills they have.
- You have to expect many submissions on stuff that has no real impact on your security and gets the level «informational» in classic penetration test. For example, if a webserver discloses ist version and patch-level in its banner.
- You have to expect submissions from people that disagree with your judgement of the vulnerability and want to extort more money from you

See also: https://www.scmagazine.com/news/security-news/vulnerabilities/what-to-do-when-a-bug-bounty-request-sounds-more-like-extortion



## Overview – When to apply which Method

| | Vulnerability Scanning | Penetration Testing | Red Teaming | Purple Teaming | Bug Bounty | Breach & Attack Simulation |
|---|---|---|---|---|---|---|
| **Finding known vulnerabilities** (in 3rd-pary libs/apps/systems before or during operation) | x | x | | | (x) | |
| **Finding vulnerabilities** ([self-developed] apps/systems before or during operation) | (x)[1] | x | | | x | |
| **Improve product security** | (x)[1] | (x)[2] | | | x | |
| **Improve defensive measures** | | | | x | | x |
| **Test defenses, incident response** | | | x | (x) | | x |
| **Educate/train blue team** | | | (x) | x | | (x) |
| **Compliance** | x | (x) | | | | |

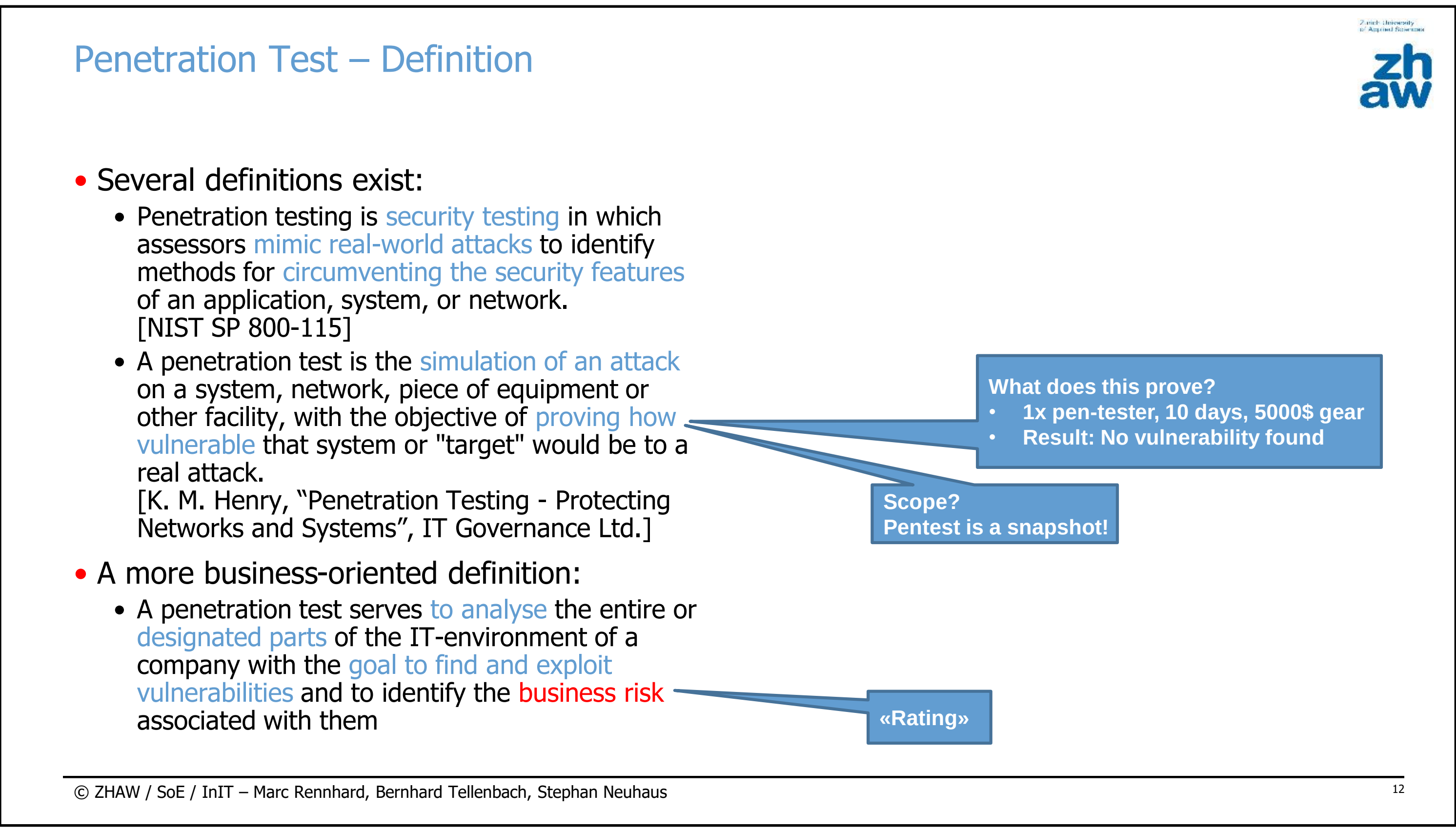
[1] Specialized scanners. Can detect vulnerabilities that can be found with "simple" testing patterns (e.g., web-application scanners looking for SQL-injection or XSS vulnerabilities) under the condition that the vulnerable item/action in the application/system can be found/triggered (e.g., web-form supplying the data) by the scanner.

[2] Pen-testing can help here, but activities earlier in the product development cycle are far more effective and should be prioritized (e.g., threat modelling, security architecture review, code review and static code analysis)

Note: For security testing methods it might be relevant whether the system under test is **internal**, **public** or **hosted on third party infrastructure** (in the cloud). For example, public bug bounty programs are usually not an option for internal systems.

The above table tries to summarize the different testing approaches with respect to how well suited they are for different purposes. Take this with a grain of salt.

If there is an «x» in brackets, this means that this method serves this purpose, but is probably not the best choice. If there is a footnote, the footnote might narrow down why the «x» is in brackets.



## Penetration Test – Definition

- Several definitions exist:
  - Penetration testing is security testing in which assessors mimic real-world attacks to identify methods for circumventing the security features of an application, system, or network. [NIST SP 800-115]
  - A penetration test is the simulation of an attack on a system, network, piece of equipment or other facility, with the objective of proving how vulnerable that system or "target" would be to a real attack. [K. M. Henry, "Penetration Testing - Protecting Networks and Systems", IT Governance Ltd.]
- A more business-oriented definition:
  - A penetration test serves to analyse the entire or designated parts of the IT-environment of a company with the goal to find and exploit vulnerabilities and to identify the business risk associated with them

**What does this prove?**
- **1x pen-tester, 10 days, 5000$ gear**
- **Result: No vulnerability found**

**Scope?**
**Pentest is a snapshot!**

**«Rating»**

© ZHAW / SoE / InIT – Marc Rennhard, Bernhard Tellenbach, Stephan Neuhaus 12

It is difficult to find a definition that matches the different uses of the term **penetration tests**.

For example, the first definition claims that penetration tests focus on applications, systems or networks. While many penetration tests are indeed focusing on those components, this is not true in the general case. A test might also involve attacks on the security of processes or people (social engineering attacks). As for the second definition, it is hardly possible to prove how vulnerable a system would be to a real attack. Reasons for this are that a penetration test is a snapshot only and that the effort spent during a penetration test does not have to be related to the effort an attacker would have to spend for the same findings and penetrations.

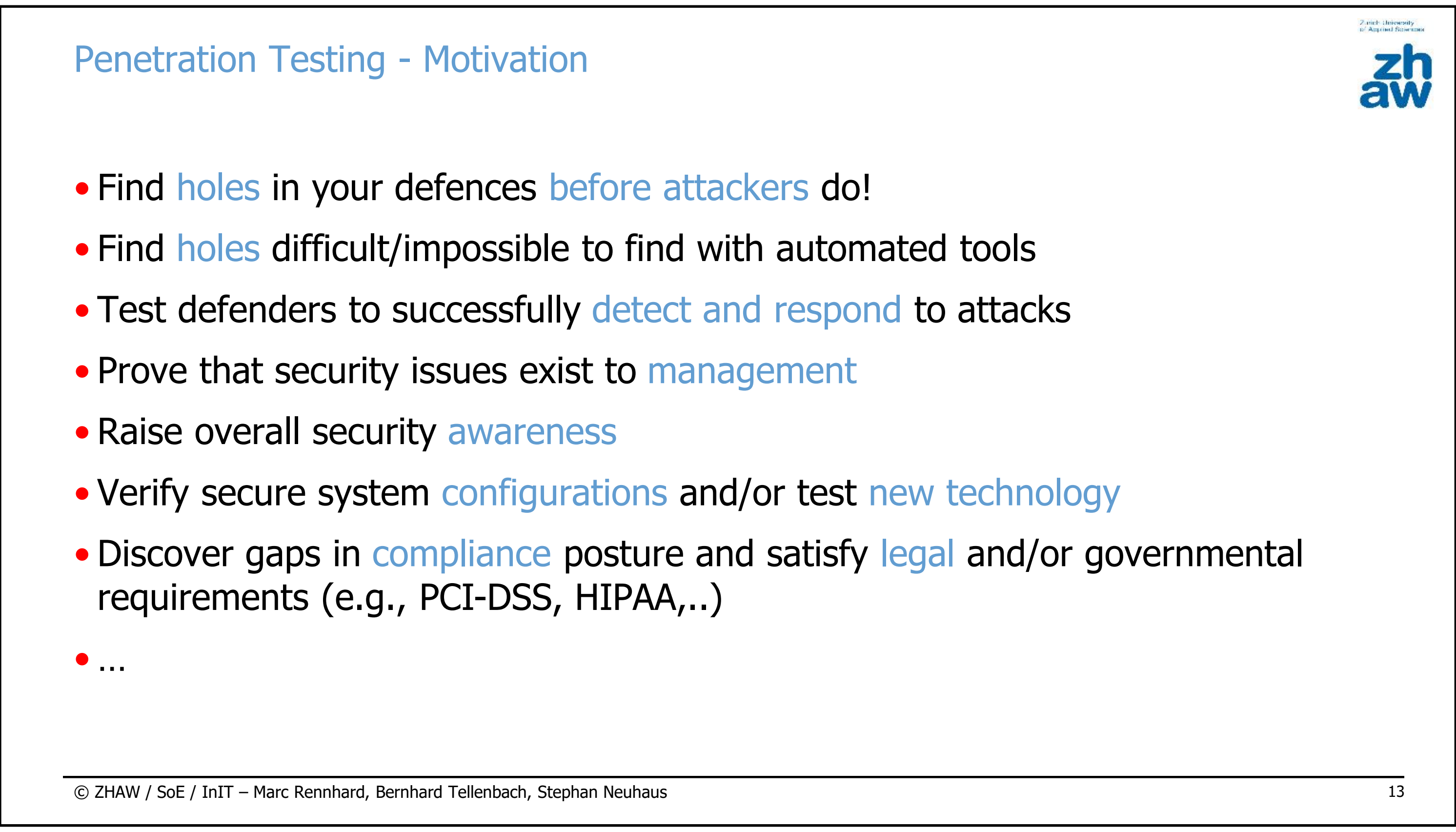
**Penetration test = snapshot**

While the test may have provided valuable results, its results are useful within the same context the test was conducted. Furthermore, consider that a penetration test is only a snapshot at a specific point in time; tomorrow the system and the test results could look totally different. This is why standards like PCI DSS require a retest if significant modifications are made.

Another point to consider is that the limited scope of penetration tests yields limited *real-world* understanding of security risk.

**Effort spent during a Penetration Test**

When a penetration tester spends 10 days to test a system and uncovers two exploitable vulnerabilities, then this is an indication that an attacker with similar skills would likely also uncover these vulnerabilities during the same time, so a penetration test is an indication about the required effort to compromise a system. Fixing the vulnerabilities and assuming that there are no easier-to-detect vulnerabilities than the one the tester has detected, then we can conclude that the system should resist an attacker that spends 10 days to find vulnerabilities to exploit the system. However, this is of course quite a "risky" conclusion and bases on the assumption that the tester has uncovered all vulnerabilities that one realistically would uncover during 10 days. Nevertheless, the more effort a (skilful) penetration tester must invest to uncover vulnerabilities, the more secure a system can be considered.



## Penetration Testing - Motivation

- Find holes in your defences before attackers do!
- Find holes difficult/impossible to find with automated tools
- Test defenders to successfully detect and respond to attacks
- Prove that security issues exist to management
- Raise overall security awareness
- Verify secure system configurations and/or test new technology
- Discover gaps in compliance posture and satisfy legal and/or governmental requirements (e.g., PCI-DSS, HIPAA,..)
- …

**A penetration test can help to:**

- Improve security
  - Identifying vulnerabilities that may be difficult or impossible to detect with automated network or application vulnerability scanning software
  - Testing the ability of (network) defenders to successfully detect and respond to the attacks

- Improve risk management
  - The spent effort and detected vulnerabilities are an indication of the effort an attacker would have to invest to achieve similar result
  - Identifying the magnitude of potential business and operational impacts of successful attacks

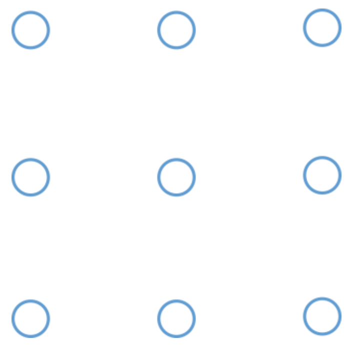
- Compliance/Investments
  - Evidence to support increasing investments in security personnel and technology to C-level management, investors, and customers
  - Meeting compliance requirements (e.g., PCI DSS)

**PCI DSS:** The Payment Card Industry Data Security Standard requires annual and ongoing (after any system changes) penetration testing.
Source: https://www.pcisecuritystandards.org/document_library

**HIPAA** (Health Insurance Portability and Accountability Act of 1996) is United States legislation that provides data privacy and security provisions for safeguarding medical information. Even though HIPAA does not specifically require penetration testing, it is mentioned in the NIST Resource guide which specifically states that penetration testing should be done *if reasonable and appropriate*.

## Focus and Scope

- Common types of penetration tests focus on:
  - Hacking and controlling IT assets
    - e.g., a network or a web application
  - Getting access to a certain piece of information
    - e.g., customer data, credit card numbers
  - Penetrating physical security
    - e.g., enter a building and access a specific asset
  - Social engineering
    - e.g., convincing someone to open an attachment
- It has a clear but limited scope
  - Web application penetration test conducted only from the point of view of the Internet browser
  - Access a system managing bank accounts of non-Swiss citizens from an outsider's point of view.
- Success factors:
  - Experience, skills, equipment, lateral and out of the box thinking



Connect the dots!

4 Straight lines only!

Limited scope of a penetration test:

- A penetration test allows for **multiple attack vectors** to be explored against the same target. Often it is the combination of information or vulnerabilities across different systems that will lead to a successful compromise. While there are examples of penetration testing that limit their scope to only one target via one vector (example, a web application pen test conducted only from the point of view of the Internet browser), their results should always be taken with a grain of salt.

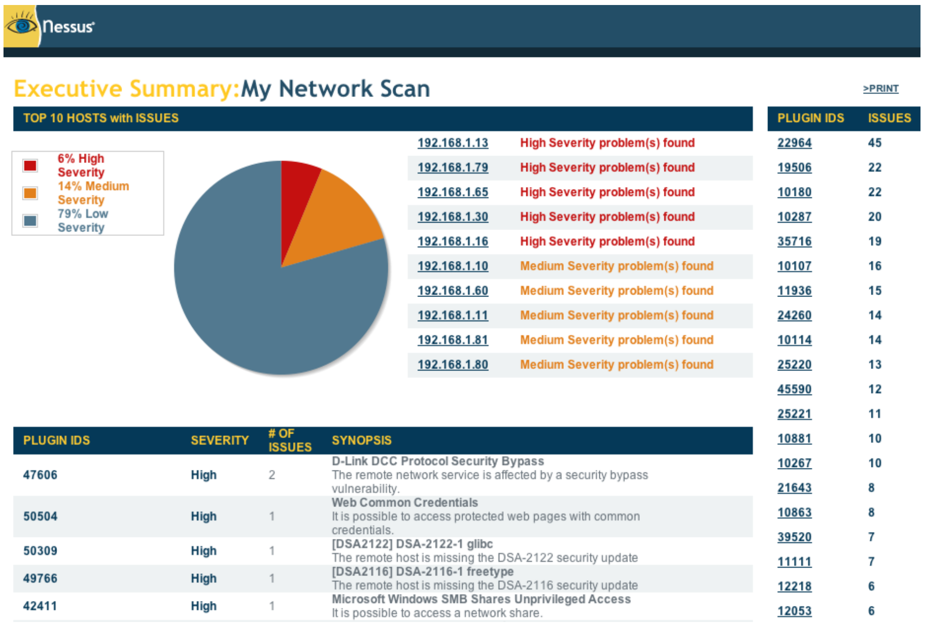
Penetration testing is not a vulnerability scan or a compliance audit.

- **Vulnerability scan:** While a penetration test may involve use of automated tools and process frameworks**, the focus is ultimately on the individual or team of testers**, the **experience** they bring to the test, and the **skills** and **wherewithal** (=the resources/money available to them) leverage in the context of an active attack on an organization. This can't be over-emphasized. Even highly automated, well-resourced, and advanced networks employing sophisticated counter-measure technologies are often vulnerable to the unique nature of the human mind, which can **think laterally** and **outside of the box**, can both **analyze** and **synthesize**, and is armed with motive and determination.
- **Compliance audits** check for the **existence** of required controls and their correct **configurations**. But **not how good** these controls are at **detecting and preventing** actual threats. Even a 100% compliant organization may still be vulnerable in the real world against a skilled human threat agent!

## Out of the Box Thinking / Thinking Outside the Box

- Think about 10 ways to turn off the light in this room

## Is this a Penetration Test?



Nessus

Executive Summary: My Network Scan

>PRINT

TOP 10 HOSTS with ISSUES

| | |
|---|---|
| 🟥 | 6% High Severity |
| 🟧 | 14% Medium Severity |
| 🟦 | 79% Low Severity |

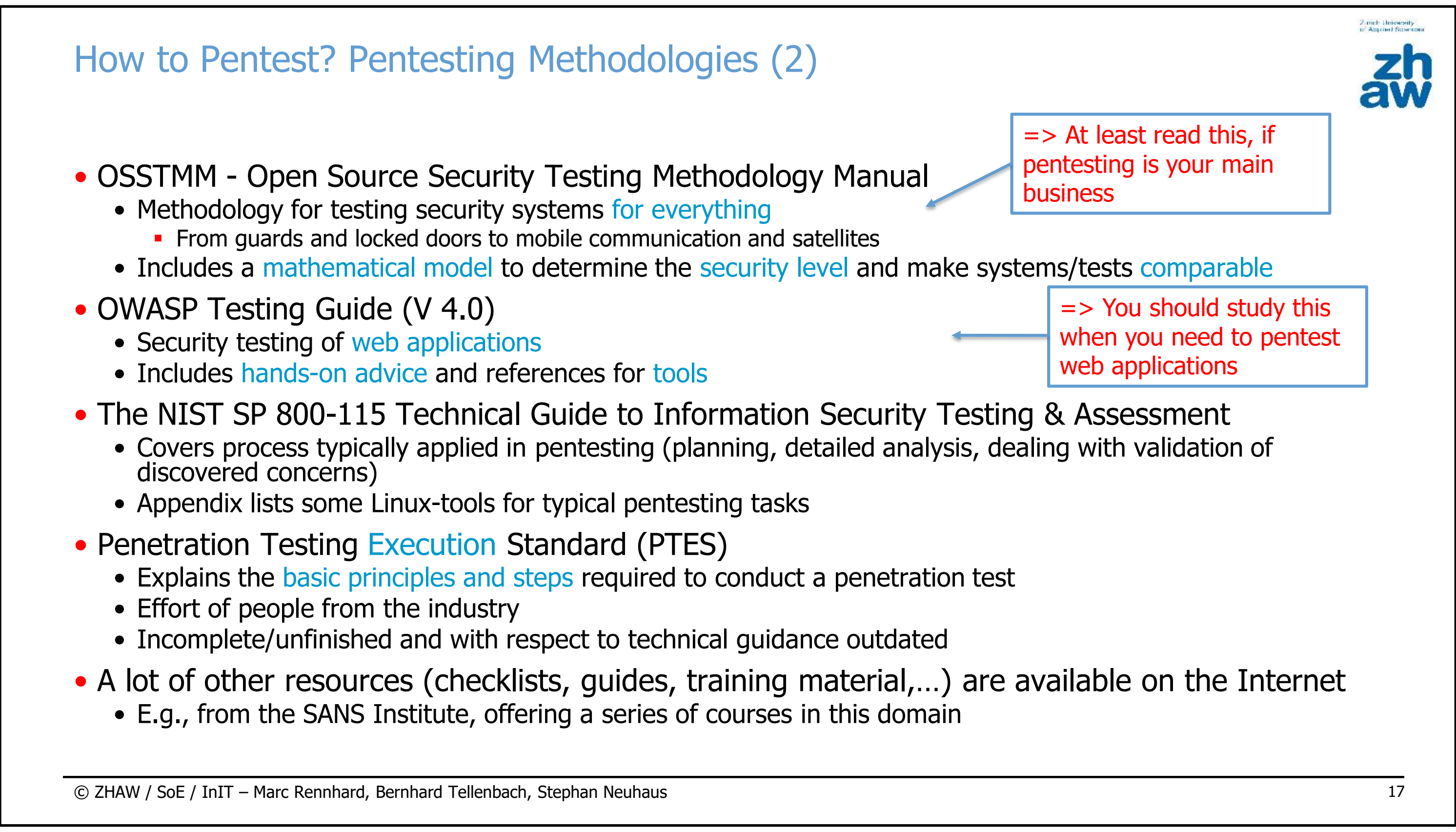
| | |
|---|---|
| 192.168.1.13 | High Severity problem(s) found |
| 192.168.1.79 | High Severity problem(s) found |
| 192.168.1.65 | High Severity problem(s) found |
| 192.168.1.30 | High Severity problem(s) found |
| 192.168.1.16 | High Severity problem(s) found |
| 192.168.1.10 | Medium Severity problem(s) found |
| 192.168.1.60 | Medium Severity problem(s) found |
| 192.168.1.11 | Medium Severity problem(s) found |
| 192.168.1.81 | Medium Severity problem(s) found |
| 192.168.1.80 | Medium Severity problem(s) found |

| PLUGIN IDS | ISSUES |
|---|---|
| 22964 | 45 |
| 19506 | 22 |
| 10180 | 22 |
| 10287 | 20 |
| 35716 | 19 |
| 10107 | 16 |
| 11936 | 15 |
| 24260 | 14 |
| 10114 | 14 |
| 25220 | 13 |
| 45590 | 12 |
| 25221 | 11 |
| 10881 | 10 |
| 10267 | 10 |
| 21643 | 8 |
| 10863 | 8 |
| 39520 | 7 |
| 11111 | 7 |
| 12218 | 6 |
| 12053 | 6 |

| PLUGIN IDS | SEVERITY | # OF ISSUES | SYNOPSIS |
|---|---|---|---|
| 47606 | High | 2 | **D-Link DCC Protocol Security Bypass** The remote network service is affected by a security bypass vulnerability. |
| 50504 | High | 1 | **Web Common Credentials** It is possible to access protected web pages with common credentials. |
| 50309 | High | 1 | **[DSA2122] DSA-2122-1 glibc** The remote host is missing the DSA-2122 security update |
| 49766 | High | 1 | **[DSA2116] DSA-2116-1 freetype** The remote host is missing the DSA-2116 security update |
| 42411 | High | 1 | **Microsoft Windows SMB Shares Unprivileged Access** It is possible to access a network share. |

**Vulnerability scanning or compliance audits are NOT penetration test!**

Source: http://tenable.typepad.com/tenable_network_security/page/11/



## How to Pentest? Pentesting Methodologies (2)

- OSSTMM - Open Source Security Testing Methodology Manual
  - Methodology for testing security systems for everything
    - From guards and locked doors to mobile communication and satellites
  - Includes a mathematical model to determine the security level and make systems/tests comparable
- OWASP Testing Guide (V 4.0)
  - Security testing of web applications
  - Includes hands-on advice and references for tools
- The NIST SP 800-115 Technical Guide to Information Security Testing & Assessment
  - Covers process typically applied in pentesting (planning, detailed analysis, dealing with validation of discovered concerns)
  - Appendix lists some Linux-tools for typical pentesting tasks
- Penetration Testing Execution Standard (PTES)
  - Explains the basic principles and steps required to conduct a penetration test
  - Effort of people from the industry
  - Incomplete/unfinished and with respect to technical guidance outdated
- A lot of other resources (checklists, guides, training material,…) are available on the Internet
  - E.g., from the SANS Institute, offering a series of courses in this domain

=> At least read this, if pentesting is your main business

=> You should study this when you need to pentest web applications

© ZHAW / SoE / InIT – Marc Rennhard, Bernhard Tellenbach, Stephan Neuhaus 17

The **Open Source Security Testing Methodology Manual** (OSSTMM) [pronunciation: awstem] was originally released by Pete Herzog in 1998 and then become an open source project in 2000. It is is distributed by the Institute for Security and Open Methodologies (ISECOM). Its goal is to improve the quality of enterprise security as well as the methodology and strategy of security testers. Rather of making security testing a black art of mystery, this document aims for repeatability, consistency, and high quality in various kinds of security tests. Earlier versions of the OSSTMM are offered for free. The latest version and drafts of new updates are accessible to subscribing members only.

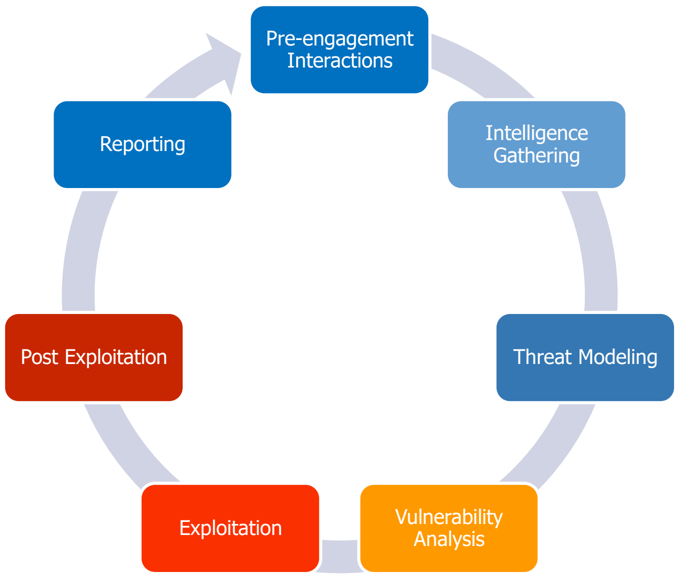
The overall document is quite extensive and covers various kinds of security tests. However, the OSSTMM does not get into depth with specific commands and tools. One of the distinguishing features of the OSSTMM is a mathematical model to determine the security level of the system under review. The rationale of having such a model is to make systems and different tests of the same system comparable. However, the mathematical model is criticised of giving a false sense of accuracy. The model claims to measure “the attack surface that is exposed” but it is questionable whether it captures it fully (also depends on the thoroughness and proficiency of the tester) and whether the way it is measured is reflecting the attack surface that is exposed in a way that is related “how secure” a system is.

The manual is continually being reviewed and modified by industry experts and contains, amongst others, a security testing methodology for all channels (Human, Physical, Wireless, Telecommunications, and Data Networks) and the Rules of Engagement which specify ethical guidelines for security tests. Security gaps are categorized into the five categories Vulnerability, Weakness, Concern, Exposure und Anomaly according to their severity.
Source: http://www.osstmm.org

The **OWASP Testing Guide (V4.0)** The aim of this guide is to help people understand the *what*, *why*, *when*, *where*, and *how* of testing web applications. The guide describes a complete testing framework, not merely a simple checklist or prescription of issues that should be addressed. Readers can use it as a template to build their own testing programs or to qualify other people’s processes. The Testing Guide describes in detail both the general testing framework and the techniques required to implement the framework in practice.

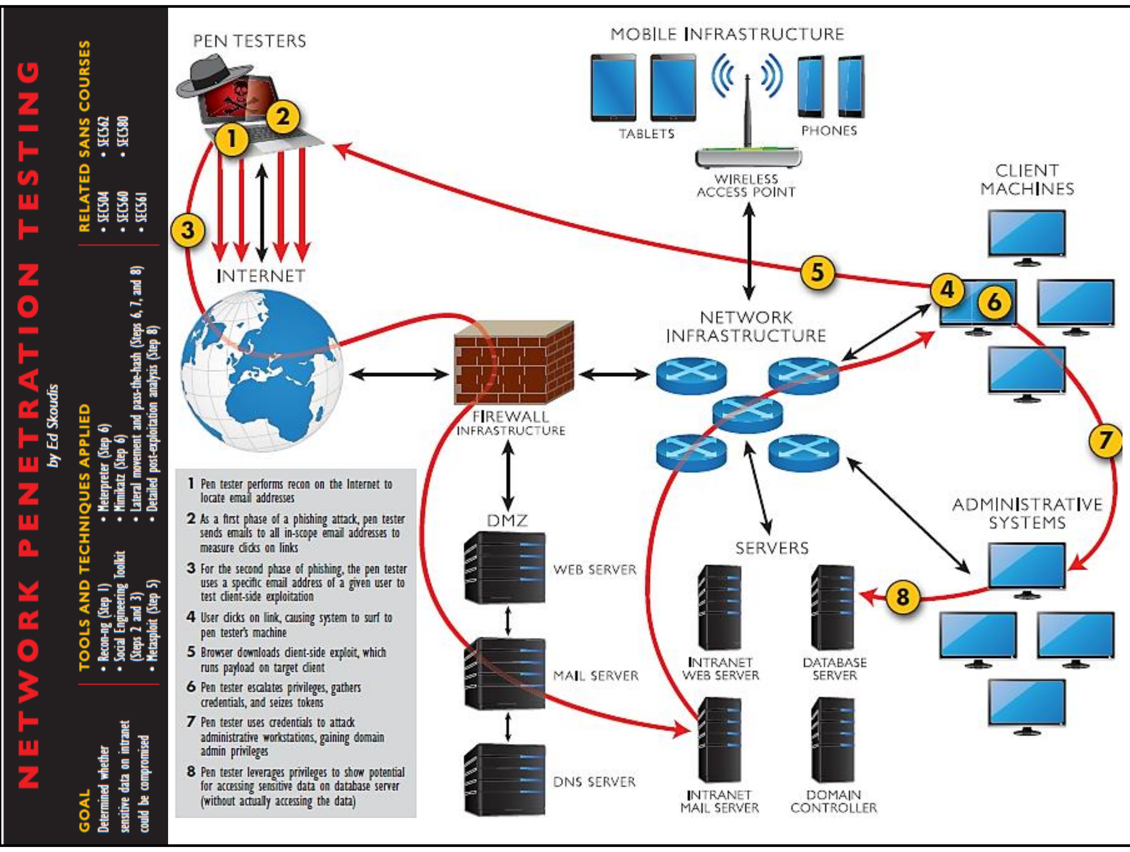
Source: https://www.owasp.org/index.php/Category:OWASP_Testing_Project

## Penetration Test: Phases



Phases borrowed from PTES:

- **Pre-engagement interactions:** Initial communication and reasoning behind a penetration test
- **Intelligence gathering and threat modeling:** Get a better understanding of the tested organization
- **Vulnerability research, exploitation:** Identify vulnerabilities and demonstrate proof-of-concept or "real" exploits
- **Post exploitation:** Determine the value of the compromised target, maintain control and gain further access to other resources
- **Reporting:** Captures the entire process in a manner that makes sense to the customer and provides the most value to it



NETWORK PENETRATION TESTING

*by Ed Skoudis*

GOAL

Determined whether sensitive data on intranet could be compromised

TOOLS AND TECHNIQUES APPLIED

- Recon-ng (Step 1)
- Social Engineering Toolkit (Steps 2 and 3)
- Metasploit (Step 5)
- Meterpreter (Step 6)
- Mimikatz (Step 6)
- Lateral movement and pass-the-hash (Steps 6, 7, and 8)
- Detailed post-exploitation analysis (Step 8)

RELATED SANS COURSES

- SEC504
- SEC560
- SEC561
- SEC562
- SEC580

1 Pen tester performs recon on the Internet to locate email addresses

2 As a first phase of a phishing attack, pen tester sends emails to all in-scope email addresses to measure clicks on links

3 For the second phase of phishing, the pen tester uses a specific email address of a given user to test client-side exploitation

4 User clicks on link, causing system to surf to pen tester's machine

5 Browser downloads client-side exploit, which runs payload on target client

6 Pen tester escalates privileges, gathers credentials, and seizes tokens

7 Pen tester uses credentials to attack administrative workstations, gaining domain admin privileges

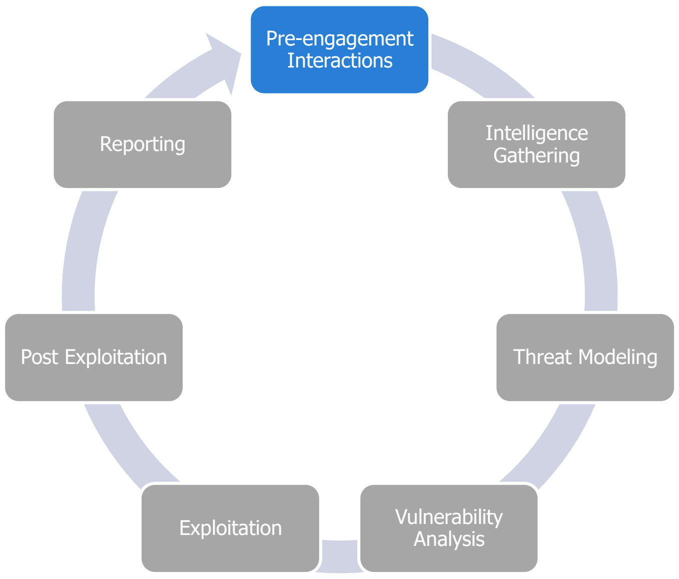
8 Pen tester leverages privileges to show potential for accessing sensitive data on database server (without actually accessing the data)

Questions:

- What phases can you identify?
- Which ones are missing?

http://counterhack.net/Poster_PenTest_2015.pdf

## Pre-engagement Interactions



Phases borrowed from PTES:

- **Pre-engagement interactions:** Initial communication and reasoning behind a penetration test
- **Intelligence gathering and threat modeling:** Get a better understanding of the tested organization
- **Vulnerability research, exploitation:** Identify vulnerabilities and demonstrate proof-of-concept or “real” exploits
- **Post exploitation:** Determine the value of the compromised target, maintain control and gain further access to other resources
- **Reporting:** Captures the entire process in a manner that makes sense to the customer and provides the most value to it

## Pre-Engagement Interactions – Overview

- Scoping – Define what is to be tested
  - An in-depth test of a single application attacked?
  - Over the Internet without the «help» of employees?
  - Test a wide range of IP addresses to find a way into the company network?
- Rules of Engagement – How that the testing is to occur
  - Method, Timeline, locations, evidence Handling,…
- Costs – What are the costs for the test?
  - Estimation, fixed price vs. on time and material basis, …

## Scope – What Does the Client Want?

- Clients are often unaware of exactly what it is they need tested or can't communicate effectively what they're expecting
  - «We need to test whether someone can break into our network»
- What the client is looking to gain from the test
  - Information on vulnerabilities and the associated risk
  - Recommendations on how to fix the problems found
  - A checkmark in a checkbox (⚠)
  - Discredit someone in a different department (⚠)
- Why the client wants/must do a pentest
  - Suspects that they have a serious security problem
  - To meet compliance requirements (⚠)
  - Management wants the new service tested before going live
  - …

- Customer: *«Please lower the rating from HIGH to MEDIUM. If you don't, we cannot go live with the product and must do a full dev and review cycle. The release has already been communicated to customers!»*
- What do you do? Discuss!

The motivation of why a client asks for a pentest and the expected gain can have significant impact on the pentest.

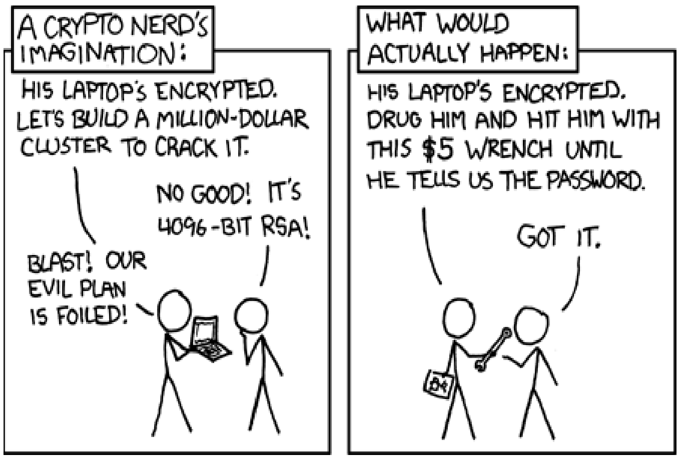
It is quite common, that pentests do have a «political» dimension.

- People responsible for the development and/or the security of the «object» under test might not like that pentesters test it and feel that management does not trust them.
- The pentest was ordered in the hope of **discrediting someone**, e.g. a staff member from a different department
- Another example is when the management wants a system that is being pentested to go live in a few days but serious problems are found, the management might try to make the pentesters **alter the rating** of certain findings or ask them to **not to test certain elements** of the «object» where they suspect such problems.

Special care needs to be taken when hitting one of these situations. Communication and support (provided information etc.) might be challenging in these cases. Avoid engaging in such pentests if possible.

## Scope – What does the Client Want?

- Common pentest types
  - Network *(e.g., get access to the intranet, try to break)*
  - Web application *(e.g. Get access to customer data)*
  - Wireless network *(e.g., get access to the intranet)*
  - Physical *(e.g., enter a secured area and access a specific asset)*
  - Social engineering *(e.g., obtaining credentials)*
- What the means (channels) used to interact with the assets should be
  - Human
  - Physical
  - Wireless
  - Telecommunications
  - Data Networks


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

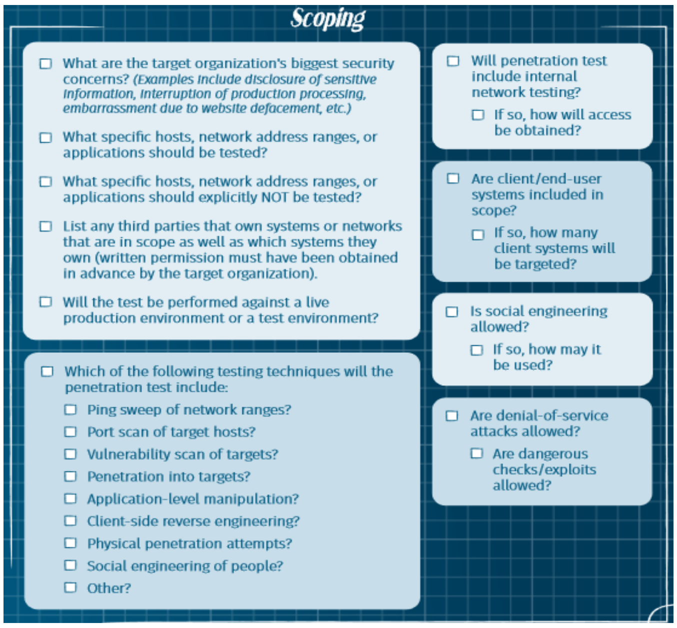
Source: https://xkcd.com/538/

**Channel:** The OSSTMM methodology uses this term to describe the means used to interact with the assets. The channels are: Human, Physical, Wireless, Telecommunications, Data Networks.

- **Human** - Comprises the human element of communication where interaction is either physical or psychological.
- **Physical** - Physical security testing where the channel is both physical and non-electronic in nature. Comprises the tangible element of security where interaction requires physical effort or an energy transmitter to manipulate.
- **Wireless** - Comprises all electronic communications, signals, and emanations which take place over the known EM spectrum. This includes ELSEC as electronic communications, SIGSEC as signals, and EMSEC which are emanations untethered by cables.
- **Telecommunications** - Comprises all telecommunication networks, digital or analog, where interaction takes place over established telephone or telephone-like network lines.
- **Data Networks** - Comprises all electronic systems and data networks where interaction takes place over established cable and wired network lines.

## Scope – What is to be Tested?

- What assets are included/excluded from the test?
  - e.g., the entire web application except for the billing component still under active development
  - e.g., list of IP addresses/networks to be tested
- Are assets from third parties affected? Which?
  - Cloud services, Internet service provider, Web hosting provider, Managed security service provider, …
    => Check terms of service
    => Check impact on test results
    - e.g., filtering of malicious traffic, DoS etc.
- Use questionnaires/checklists to guide the process of defining the scope
  - To avoid pitfalls, you need experience



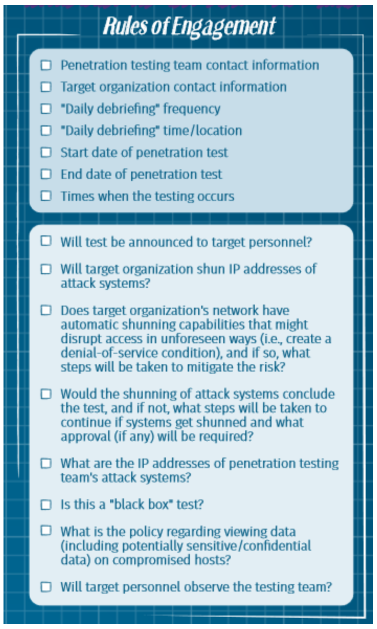
**Scoping**

- ☐ What are the target organization's biggest security concerns? *(Examples include disclosure of sensitive information, interruption of production processing, embarrassment due to website defacement, etc.)*
- ☐ What specific hosts, network address ranges, or applications should be tested?
- ☐ What specific hosts, network address ranges, or applications should explicitly NOT be tested?
- ☐ List any third parties that own systems or networks that are in scope as well as which systems they own (written permission must have been obtained in advance by the target organization).
- ☐ Will the test be performed against a live production environment or a test environment?
- ☐ Which of the following testing techniques will the penetration test include:
  - ☐ Ping sweep of network ranges?
  - ☐ Port scan of target hosts?
  - ☐ Vulnerability scan of targets?
  - ☐ Penetration into targets?
  - ☐ Application-level manipulation?
  - ☐ Client-side reverse engineering?
  - ☐ Physical penetration attempts?
  - ☐ Social engineering of people?
  - ☐ Other?
- ☐ Will penetration test include internal network testing?
  - ☐ If so, how will access be obtained?
- ☐ Are client/end-user systems included in scope?
  - ☐ If so, how many client systems will be targeted?
- ☐ Is social engineering allowed?
  - ☐ If so, how may it be used?
- ☐ Are denial-of-service attacks allowed?
  - ☐ Are dangerous checks/exploits allowed?

Source: https://www.sans.org/blog/sans-poster-building-a-better-pen-tester-pdf-download/

## Pre-engagement Interactions – Rules of Engagement

- How the testing is to occur, including the test method
- Timeline – What happens when between start and end date?
- Status meetings - Definition of regular status meetings
  - Agenda: Plan, progress, problems
- Time of Day to Test – Limitations on when the testing should occur
  - E.g., only between 20:00 and 06:00 (outside of business hours) and the backup system should not be attacked between 00:00 and 04:00
- Use of evasion/stealth techniques
  - Depends on test methodology (black-box, white-box, …)
- Evidence handling - How to handle/store results
  - Encrypted and limit access on a need-to-know basis
- Permission to test document - States scope and acknowledges awareness of the tester’s activities

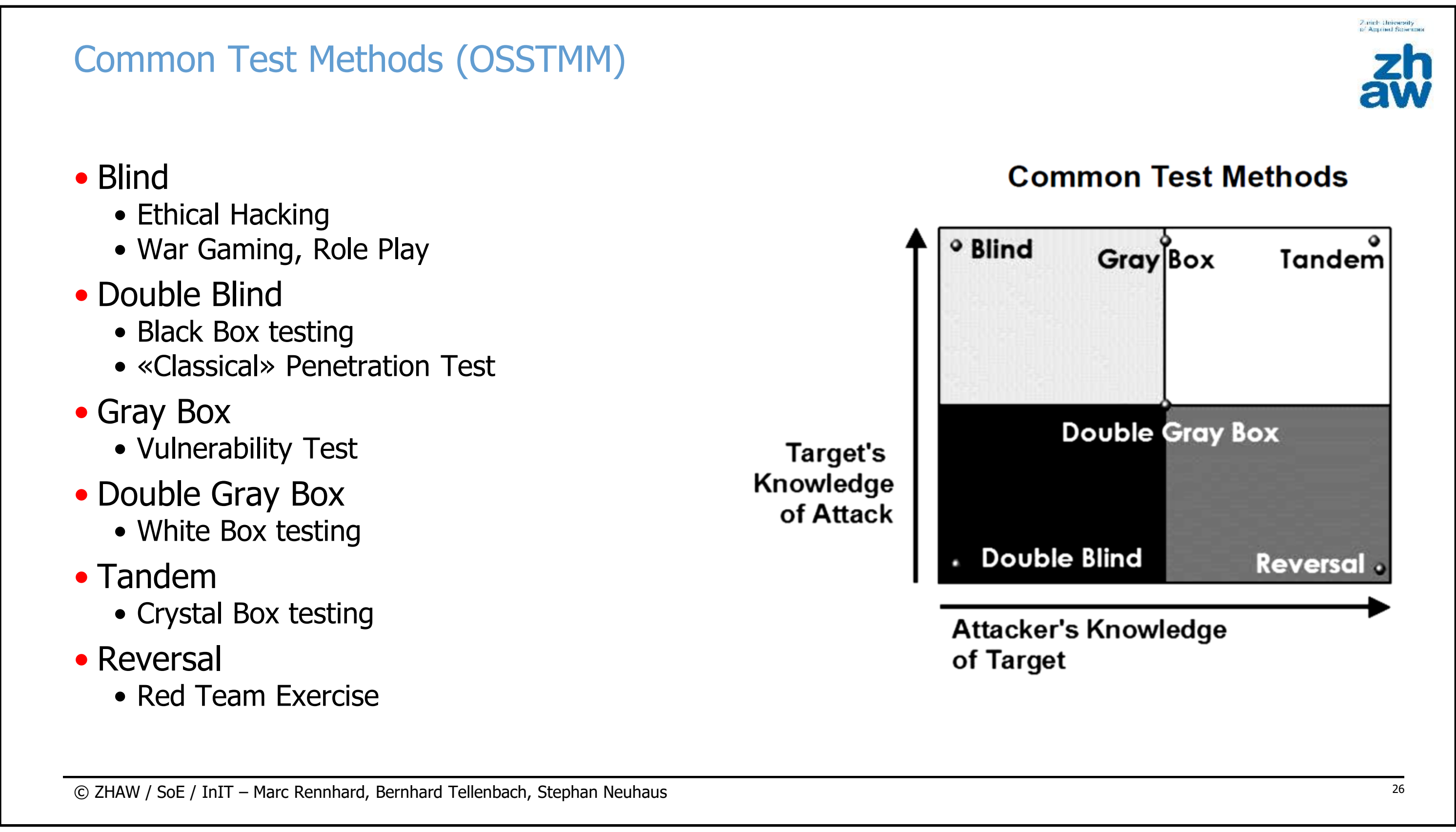


**Rules of Engagement**

- ☐ Penetration testing team contact information
- ☐ Target organization contact information
- ☐ "Daily debriefing" frequency
- ☐ "Daily debriefing" time/location
- ☐ Start date of penetration test
- ☐ End date of penetration test
- ☐ Times when the testing occurs

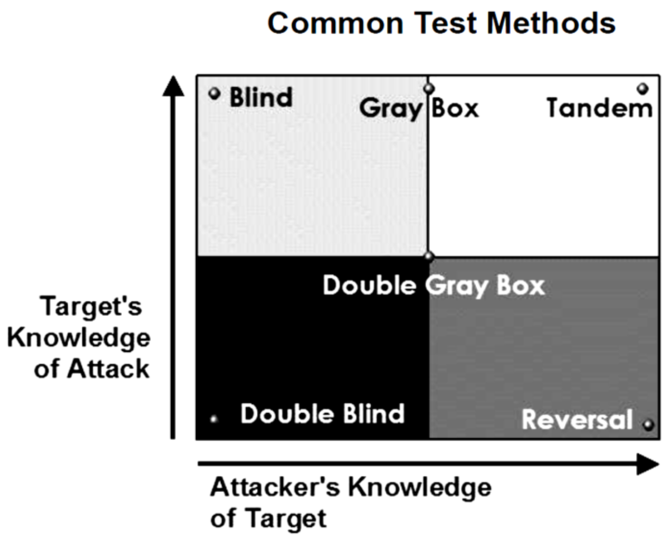
- ☐ Will test be announced to target personnel?
- ☐ Will target organization shun IP addresses of attack systems?
- ☐ Does target organization's network have automatic shunning capabilities that might disrupt access in unforeseen ways (i.e., create a denial-of-service condition), and if so, what steps will be taken to mitigate the risk?
- ☐ Would the shunning of attack systems conclude the test, and if not, what steps will be taken to continue if systems get shunned and what approval (if any) will be required?
- ☐ What are the IP addresses of penetration testing team's attack systems?
- ☐ Is this a "black box" test?
- ☐ What is the policy regarding viewing data (including potentially sensitive/confidential data) on compromised hosts?
- ☐ Will target personnel observe the testing team?

Source: https://www.sans.org/blog/sans-poster-building-a-better-pen-tester-pdf-download/



## Common Test Methods (OSSTMM)

- Blind
  - Ethical Hacking
  - War Gaming, Role Play
- Double Blind
  - Black Box testing
  - «Classical» Penetration Test
- Gray Box
  - Vulnerability Test
- Double Gray Box
  - White Box testing
- Tandem
  - Crystal Box testing
- Reversal
  - Red Team Exercise



© ZHAW / SoE / InIT – Marc Rennhard, Bernhard Tellenbach, Stephan Neuhaus 26

White-box testing:

- Good level of information about the test target – e.g., the source code of the web application under test and information about the protection measures in place

Black-box testing:

- Almost no information about the test target - e.g., only the name of a company or the URL to the web application

Gray-box Testing

- Anything between black-box and white-box testing. Most pentests are in fact gray-box tests.

<u>Common test methods:</u>

- **Blind -** The Analyst engages the target with no prior knowledge of its defenses, assets, or channels. The target is prepared for the audit, knowing in advance all the details of the audit. A blind audit primarily tests the skills of the Analyst. The breadth and depth of a blind audit can only be as vast as the Analyst’s applicable knowledge and efficiency allows. In COMSEC and SPECSEC, this is often referred to as **Ethical Hacking** and in the PHYSSEC class, this is generally scripted as **War Gaming** or **Role Playing**.
- **Double Blind -** The Analyst engages the target with no prior knowledge of its defenses, assets, or channels. The target is not notified in advance of the scope of the audit, the channels tested, or the test vectors. A double blind audit tests the skills of the Analyst and the preparedness of the target to unknown variables of agitation. The breadth and depth of any blind audit can only be as vast as the analyst’s applicable knowledge and efficiency allows. This is also known as a **Black Box test** or **Penetration test**.
- **Gray Box -** The Analyst engages the target with limited knowledge of its defenses and assets and full knowledge of channels. The target is prepared for the audit, knowing in advance all the details of the audit. A gray box audit tests the skills of the Analyst. The nature of the test is efficiency. The breadth and depth depends upon the quality of the information provided to the Analyst before the test as well as the Analyst’s applicable knowledge. This type of test is often referred to as a **Vulnerability Test** and is most often initiated by the target as a self-

assessment.

- **Double Gray Box** - The Analyst engages the target with limited knowledge of its defenses and assets and full knowledge of channels. The target is notified in advance of the scope and time frame of the audit but not the channels tested or the test vectors. A double gray box audit tests the skills of the Analyst and the target's preparedness to unknown variables of agitation. The breadth and depth depends upon the quality of the information provided to the Analyst and the target before the test as well as the analyst's applicable knowledge. This is also known as a **White Box** test.
- **Tandem -** The Analyst and the target are prepared for the audit, *both knowing in advance all the details of the audit*. A tandem audit tests the protection and controls of the target. However, it cannot test the preparedness of the target to unknown variables of agitation. The true nature of the test is thoroughness as the Analyst does have full view of all tests and their responses. The breadth and depth depends upon the quality of the information provided to the Analyst before the test (transparency) as well as the Analyst's applicable knowledge. This is often known as an In-House Audit or a **Crystal Box** test and the Analyst is often part of the security process.
- **Reversal -** The Analyst engages the target with *full knowledge of its processes and operational security*, but the target knows nothing of what, how, or when the Analyst will be testing. The true nature of this test is to audit the preparedness of the target to unknown variables and vectors of agitation. The breadth and depth depends upon the quality of the information provided to the Analyst and the Analyst's applicable knowledge and creativity. This is also often called a **Red Team exercise**.

Source: The **Open Source Security Testing Methodology Manual** (OSSTMM) v 3.0

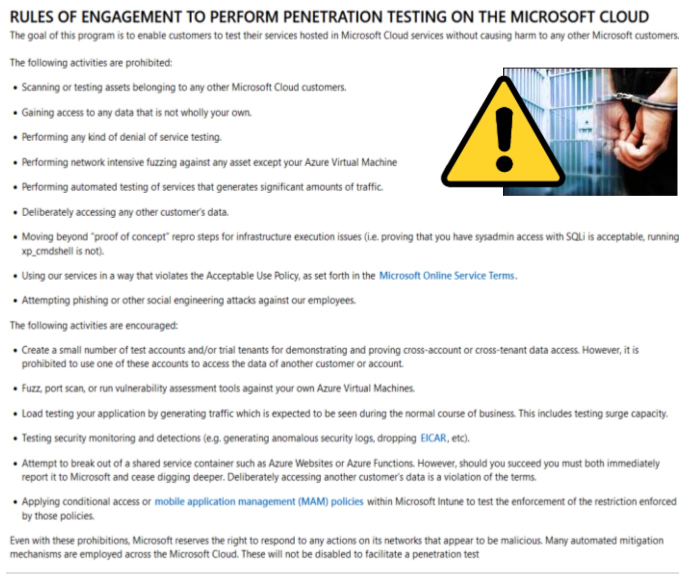
## Evidence Handling – Sensitive Information

- Do not store/access personally identifiable information (PII)
- Do not store/access sensitive personally identifiable information (PII)
  - Religion, race, political opinions and activities,…
- No personal health information (PHI)

But how to prove that access to this information could be obtained?

## Rules of Engagement – Permission to Test Document

- States scope and acknowledges awareness of the tester's activities
- Includes ALL permissions required + granted (customer, 3rd parties, …)
  - For example, testing a service deployed in the Microsoft cloud is permitted (some limits apply)
- «Online» tests: Liability in case of system instabilities/crashes
  - Testing can lead to system instability / crash despite all due care => tester shall not be hold liable for this



**RULES OF ENGAGEMENT TO PERFORM PENETRATION TESTING ON THE MICROSOFT CLOUD**

The goal of this program is to enable customers to test their services hosted in Microsoft Cloud services without causing harm to any other Microsoft customers.

The following activities are prohibited:

- Scanning or testing assets belonging to any other Microsoft Cloud customers.
- Gaining access to any data that is not wholly your own.
- Performing any kind of denial of service testing.
- Performing network intensive fuzzing against any asset except your Azure Virtual Machine
- Performing automated testing of services that generates significant amounts of traffic.
- Deliberately accessing any other customer's data.
- Moving beyond "proof of concept" repro steps for infrastructure execution issues (i.e. proving that you have sysadmin access with SQLi is acceptable, running xp_cmdshell is not).
- Using our services in a way that violates the Acceptable Use Policy, as set forth in the Microsoft Online Service Terms.
- Attempting phishing or other social engineering attacks against our employees.

The following activities are encouraged:

- Create a small number of test accounts and/or trial tenants for demonstrating and proving cross-account or cross-tenant data access. However, it is prohibited to use one of these accounts to access the data of another customer or account.
- Fuzz, port scan, or run vulnerability assessment tools against your own Azure Virtual Machines.
- Load testing your application by generating traffic which is expected to be seen during the normal course of business. This includes testing surge capacity.
- Testing security monitoring and detections (e.g. generating anomalous security logs, dropping EICAR, etc).
- Attempt to break out of a shared service container such as Azure Websites or Azure Functions. However, should you succeed you must both immediately report it to Microsoft and cease digging deeper. Deliberately accessing another customer's data is a violation of the terms.
- Applying conditional access or mobile application management (MAM) policies within Microsoft Intune to test the enforcement of the restriction enforced by those policies.

Even with these prohibitions, Microsoft reserves the right to respond to any actions on its networks that appear to be malicious. Many automated mitigation mechanisms are employed across the Microsoft Cloud. These will not be disabled to facilitate a penetration test

Check out the PDF distributed with these slides for an example for a permission to test document.

**Image:** Rules of Engagement / Permission to Test from Microsoft for YOUR services on the Microsoft Cloud

Source: https://www.microsoft.com/en-us/msrc/pentest-rules-of-engagement?SilentAuth=1&wa=wsignin1.0

## Pre-engagement Interactions - Communication

- Establish lines of communication - How often you interact with the customer and how?
- Emergency contact information
  - List with contacts from all relevant parties *(incl. full name, title and operational responsibility)*
  - Authorization to discuss details of the testing activities
  - Two forms of 24/7 immediate contact if possible *(e.g., instant messenger and cell phone)*
  - How to exchange (large) files *(e.g., secure sharing platform, must be tested and ready)*
- Incident reporting process - Target organization is aware* of when the tests are being conducted
  - Or do you want the security operations centre to call upper management in the middle of the night?
- Status report frequency
  - An ignored customer is a former customer
- Communication channels - How to communicate with the customer securely

***** for some test methods (e.g., reversal-style) this might not be true for the security operations centre staff but the people that would get called by the SOC should be informed.

**Recommendation for who should be on the contact list (minimum):**

- All penetration testers in the test group for the engagement
- The manager of the test group
- Two technical contacts at each target organization
- Two technical contacts at the customer
- One upper management or business contact at the customer

## Challenge - Scope Creep

- Extension/alteration of the scope requiring more time and effort to complete the engagement
- Client: «You did great work by discovering this problem; please verify that this can indeed be exploited»
- Pentester: «Thanks a lot! Sure, we can do this!»

=> What to do in such a case? Discuss!