Trustwave®

QUICK REFERENCE GUIDE:

Penetration Testing

Trustwave SpiderLabs

Information Security is a swift-moving and highly complex field. Practitioners need to walk a fine line between doing what is required and keeping abreast of what is happening.

It would be great to ingest expert-level knowledge about a subject by simply plugging in a memory card or downloading a library of eBooks directly to your brain. Unfortunately, that technology is not quite production-ready; in the interim, here is a quick distillation of the most salient information you need to impress your colleagues when discussing the topic of penetration testing.

This guide aims to rapidly arm you with vital foundational knowledge, a few trade secrets, and some additional resources if you want to take a deeper dive.

Testing and Ethical Hacking

Overview

Testing is a discipline of quality assurance using multiple methodologies and tactics to identify security issues in a given system, process, or software application. Several of the testing terms listed here are often used interchangeably, despite the subtle variances among them, so having a basic understanding of the differences is beneficial.

- Penetration testing and vulnerability assessments
- Infrastructure testing
- Web application testing
- Mobile application testing
- Cloud testing
- Red team engagements
- Purple team exercises
- Blackbox, Greybox, and Whitebox Testing
- Bug Bounties

Penetration Testing and Vulnerability Assessments

Penetration testing is primarily a manual process, reliant on a highly skilled and experienced team using tools and techniques to test a given system to identify, validate, and document security weaknesses.

Conversely, a vulnerability assessment is a mostly automated process that relies on a scanning tool- or collection of tools- configured with a range of IP addresses. It may also be a single URL with settings to determine the speed and aggressiveness of the scan or specifying the vulnerabilities that need identification.

Automated vulnerability assessments also have inherent pitfalls and may include:

- False positives (i.e., the tool thinks there is an issue, but in fact, an issue does not exist.)
- Erroneously rated issues (i.e., the tool reports a finding as high or critical, but due to the target being on an internal and secure network, the real risk is actually lower.)
- Multiple issues resulting from a single root cause (i.e., here is a 1000-page list of 100 servers that require the following 30 patches.)

The primary difference between an automated vulnerability assessment and a manual penetration test is the addition of human intelligence that brings a creative, outside-the-box mindset focused on identifying clues and creating hypotheses to test.

Penetration testing demonstrates how exploiting a vulnerability is possible, whereas a vulnerability assessment would only tell you what is vulnerable and in need of patching. Penetration tests result in a report that details priority recommendations while also considering specific business context and risks.

.

Example

Using observations alone to determine how to break into a house, a vulnerability assessment identifies if you can crack the locks or if a window is ajar. A penetration test would tell you how you can both pick the locks and open the windows-without leaving any evidence.

It is also imperative to understand that the difference between a real attacker and a penetration tester is time. An attacker can take as long as they want and only needs to find a single issue; by contrast, a tester has a very constrained window of time to attempt to discover all of the potential problems (often called the poacher/gamekeeper analogy).

Infrastructure Testing

Infrastructure testing relates to the testing of underlying networks or supporting infrastructure. For example, in a web application test, the underlying web server is typically included in the testing; similarly, in determining vehicle safety, the tester considers the driver's competency to some degree.

Infrastructure testing can encompass everything including hardware, software, and networks. It can span operating systems, access levels, cloud environments, operational technology, wireless systems, internal workstations, printers, fax machines, or SCADA[1] systems. This type of testing requires a clearly defined scope to be valuable.

Web Application Testing

In web application testing, software developers typically construct test cases or use cases to see how their application performs with both valid and invalid requests. For example, if you are building an online website to sell alcohol, there is a need to ask the person placing the order their age. A valid response might be any number 18 or over (21 in the US).

By contrast, a security tester may submit 3,000, -14, Wednesday, or even *^%*$\/*&? to see how the application behaves. Suppose the application rejects all these inputs and responds with sorry age not valid, try again; that is a good thing. If, on the other hand, the application breaks and throws an error, such as Warning: Mysql_age is not a valid number, that might indicate that the developer did not adequately sanitize the input. An attacker could exploit this error by crafting an input that displays the credit card information of others.

A good web application test will cover all the most common types of risks in web applications using an established framework such as the OWASP Top 10.

Mobile Application Testing

Mobile application testing is often very similar to web application testing because the application tested is often, in practical terms, a web application packaged to appear like a mobile application. The tester will take the application and interrogate the back-end systems that the application talks to, attempting to subvert any controls that have been built in or retrieve any sensitive information stored within the application.

Additional steps in mobile application testing:

- Is the mobile application designed to interact with other devices (often a phone or tablet)?
- Does it contain unauthorised (or unnecessary) access to the device's data?
- Does it introduce other avenues wherein the device itself can be compromised?

[1] Supervisory control and data acquisition (SCADA) is a system of software and hardware elements that allows industrial organizations to: Control industrial processes locally or at remote locations., Monitor, gather, log, and process real-time data., Directly interact with devices such as sensors, valves, pumps, motors, and more through human-machine interface (HMI) software

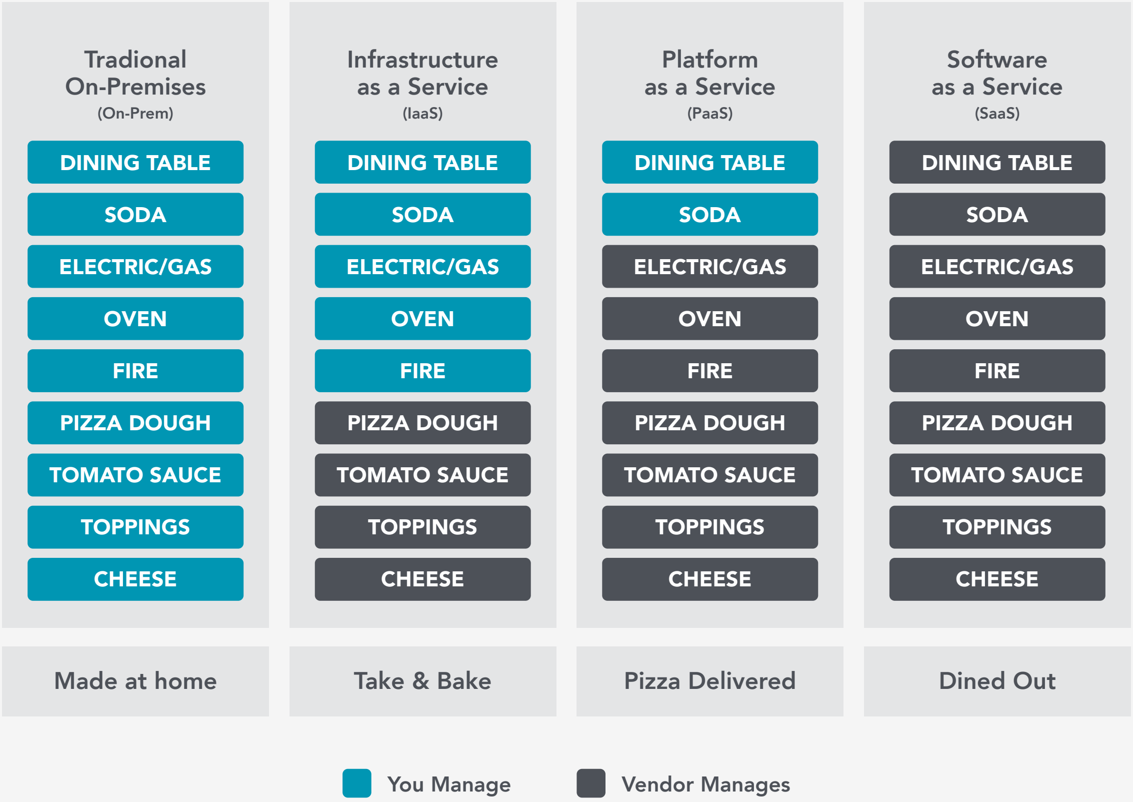
Cloud Testing

Testing in the cloud requires a slightly different approach. Not understanding the nuances between cloud and internal testing can lead to poor outcomes such as:

- Testing efforts scoped incorrectly or overpriced
- Testing unmanaged or unowned components
- Overlooked/missed issues
- A potentially toxic relationship with your cloud service provider

Testing and scoping out a cloud environment requires the solution architecture to be well understood and the shared responsibility model to be in play. The Pizza as a Service Model can help to explain this.

Pizza as a Service



Tradional On-Premises (On-Prem)	Infrastructure as a Service (IaaS)	Platform as a Service (PaaS)	Software as a Service (SaaS)
🟦 DINING TABLE	🟦 DINING TABLE	🟦 DINING TABLE	⬛ DINING TABLE
🟦 SODA	🟦 SODA	🟦 SODA	⬛ SODA
🟦 ELECTRIC/GAS	🟦 ELECTRIC/GAS	⬛ ELECTRIC/GAS	⬛ ELECTRIC/GAS
🟦 OVEN	🟦 OVEN	⬛ OVEN	⬛ OVEN
🟦 FIRE	🟦 FIRE	⬛ FIRE	⬛ FIRE
🟦 PIZZA DOUGH	⬛ PIZZA DOUGH	⬛ PIZZA DOUGH	⬛ PIZZA DOUGH
🟦 TOMATO SAUCE	⬛ TOMATO SAUCE	⬛ TOMATO SAUCE	⬛ TOMATO SAUCE
🟦 TOPPINGS	⬛ TOPPINGS	⬛ TOPPINGS	⬛ TOPPINGS
🟦 CHEESE	⬛ CHEESE	⬛ CHEESE	⬛ CHEESE
Made at home	Take & Bake	Pizza Delivered	Dined Out

🟦 You Manage ⬛ Vendor Manages

Here we see the difference between your responsibility (blue) and the cloud service provider's responsibility (green). Where you are deploying a Software as a Service tool such as Office365, you should not need to and will be very unlikely to get permission to test the underlying cloud infrastructure. Instead, a discipline called vendor risk management can assure that the tool is safe by requesting proof from the cloud service provider that they carried out their own penetration testing.

Red Team Testing

Very mature organisations use red teaming exercises to test and improve their blue teams' (defensive) skills and their ability to identify and react to realistic attack scenarios in a controlled manner.

Unlike a penetration test, which might include between five and ten days of effort, it is not uncommon for a red team engagement to last months and include multiple elements. A red team engagement can encompass everything from a targeted phishing campaign and the capture of valid credentials to an orchestrated breach of physical security, such as gaining access to a computer room through lockpicking or social engineering techniques.

A Red team engagement may also include:

- Open-Source Intelligence (OSINT) collection and previous data breach information
- Network discovery, reverse engineering, and exploitation
- Web and mobile application discovery, reverse engineering, and exploitation
- Online password-based attacks for external resources
- Targeted spear-phishing attacks for both endpoint, and credential compromise
- Voice-based and other electronic social engineering attacks
- Internal network compromise via independently established and/or assisted footholds

Rather than attempting to test every aspect of a given system, several flags to capture—or goals to achieve—are assigned, such as gaining domain administrator-level access or access to an executive mailbox. The rules of engagement set out what is and is not permitted. For example, it is uncommon for red teams to target individuals via their personal social media accounts or email addresses, even though this is a common attack vector in the wild.

How a red team exercise is different from a penetration test

Consider the example of a reserve bank that is home to 50 million dollars in Gold Bullion.

A penetration test, being limited in time and scope, could be a test to discover if access can be gained through the bank's front door, simulating an opportunistic attack.

A red team, having a wider and more fluid scope and a large amount of time, could test to see if a highly motivated criminal could perform an Oceans 11 style breach on the bank and clear the vault, simulating an advanced attack or Advanced Persistent Threat.

Purple Teaming

Purple teaming is a blend of red and blue teaming. It involves a collaborative approach between the two teams, similar to a relationship between a boxer and a trainer. When sparring with the trainer, who is calling out improvements and refinements as they spar, the boxer hones their skills in real-time.

By shortening the feedback loop between what the red team is doing and what a blue team is or should be observing and reacting to, the amount of information shared and the likelihood that it lands with the best person to receive that information increases.

A successful purple team engagement raises the skills of both the red and blue teams.

Other Kinds of Testing

BLACKBOX, GREYBOX, AND WHITEBOX TESTING

This testing classification identifies how much information the tester receives about the target system before the test itself. Although it may seem counterintuitive to provide a tester with intel that a real-world attacker might not have, time is of the essence. Performing research to understand something that ultimately does not lead to any vulnerabilities is precious time lost.

Additionally, the recommendations resulting from a penetration test can often be far more specific and detailed when testers have a firm working knowledge of the environment they are testing. This knowledge can be the difference between a testing report stating investigate why login field accepts special characters and upgrade the following dependency to version 2.3.7.

Black Box Testing

The testers are given no prior information about the target system and can only rely on what they can determine from their own interrogations.

Based on the principle of "security through maturity," it appears to mimic the approach of a real attacker but in reality, results in a less efficient use of valuable time.

Gray Box Testing

This is the most common scenario in which limited information is provided to the testers. Often, organisations either do not have access to or would prefer not to reveal their source code.

White Box Testing

All information is made available to the tester; this may include:

1. Network diagrams
2. Source code
3. Directory listings
4. Admin credentials
5. Vendor and design documentation

In addition, there are open lines of dialogue between the testers and the teams responsible for building and operating the solution.

Bug Bounties

Bug bounties take a slightly different approach to testing, where many people with varying skill levels compete to find issues in a given system, with cash rewards for the first person to find and report each issue. Typically, the more severe the issue, the higher the bounty or prize.

Running a bug bounty program is a notoriously complicated task, and for this reason, most bug bounties work best in mature environments. An untested system is likely to have many issues that can lead to the bounty pool being exhausted quickly on low hanging fruit, leaving more significant problems undetected. Penetration testing ensures the common issues have been tested and resolved early. Professional bug bounty hunters are then used to identify bugs specific to a particular vertical or technology nuance. For this reason, bug bounties can be a suitable supplement to a mature security programme but are not a substitute for penetration testing.

Smart Questions to Ask

How do I choose the right penetration testing provider?

Use providers that have recognised credentials, such as Offensive Security Certified Professional (OSCP), Offensive Security Certified Expert (OSCE), CREST Registered Penetration Tester (CRT-Pen) or CREST Certified Tester (CCT).

Look for a tester with a large and diverse team; the larger the team is, the more likely the vendor can field a tester with experience in the specific technology stack, in the industry, and in the time frame you need testing. Smaller players may not have the right person available when you need them.

For a larger piece of work, ask for references. A reputable vendor will be willing to provide you with the names of other testing customers who are satisfied with the services.

Consult your peer network and ask the people whom you trust if they have any recommendations.

If you procure a lot of testing, it is both healthy and desirable to have a panel of testing companies to call, rather than being locked into a single provider.

This practice gives you a larger pool of testers and thus more ability to test as and when you need to. Additionally, different vendors testing the same target may uncover differing issues, so cycling your testers is also to your advantage.

Remember not to fall into the trap of choosing the cheapest quote. If you do this, compare both the quoted effort and daily rate. (A low day rate is only less expensive if the number of days is less).

Have we checked the basics?

Don’t pay a vendor to find blank MySQL passwords or tell you that your servers are unpatched; use a vulnerability scanner for these more common issues. Scanners are great at finding blatant network vulnerabilities and other low hanging fruit. Your penetration test vendor can identify the paths to data compromise that a vulnerability scanner never will.

What are the drivers for requesting this testing? Is this a requirement of PCI Compliance, a Contractual obligation, or is this just good practice?

The reasons you are testing can have a bearing on things like the level of reporting required. For instance, you may not need formal reporting if you perform testing solely to confirm issue remediation; however, if you are conducting testing as a requirement of your PCI DSS Certification, some components require documentation.

What is the scope of this testing?

Understanding the scope of a test is the key to understanding how much effort will be required.

A testing scope can be as narrow as a single URL, or IP Address, or even a specific functionality within an application following an update. Conversely, the test’s scope can be as broad as your entire internet-facing attack surface and supporting infrastructure. Work with your vendor to determine the best type of test for your requirements; they can help you decide between a vulnerability assessment versus something more detailed.

Post-penetration testing

Did the report results deliver actionable information?

The final report should contain details of issues, including a clear description of the potential to exploit each and, when applicable, either a proof of concept or instructions to recreate the issues. This information will give your teams a better understanding of the problems and how to remediate them.

Your testers should also be willing and able to jump on a call with your technical teams and explain their methodology and findings where necessary. A good tester will relish the opportunity to explain to a developer how they can find and prevent similar issues in their future work.

Ask the tester did they have enough time for this test?

Asking a tester if they had enough time is a great question to ask at the conclusion of every test to be better prepared for the next round of testing.

The most effective way to determine if the estimated effort is sufficient would be to carry out the testing and ask the tester if they would have preferred more time. Because of tight deadlines, sparse budgets, or the system simply not being ready during the testing window, a tester may feel that the time they had was insufficient, which may imply a need for additional testing.

Most testing organisations use a concept called Timeboxing when scoping a test. Timeboxing means the testers will attempt to cover all common issues and then use their experience to maximise any remaining time by applying a risk-based approach to identify crucial vulnerabilities first. Prioritising the time remaining after identifying any common issues acknowledges that the testing time is finite, and poorly scoped tests will not come close to identifying any remaining issues that may exist.

Did your SIEM log any alerts?

A good penetration test is a simulation of an actual attack or breach. If you have a SIEM or central logging solution, verify if the testing generated any alerts. If the tester tried common passwords for every one of your domain users and you were not aware of this until you read the report, you will want to discuss this. Why was this not found previously? What improvements are needed to detect and, where possible, react in real-time to similar future attacks?

Do you have a plan to implement the recommendations?

Heed the advice of all penetration tests and implement it as soon as possible. The longer an issue exists, the more likely it will be found and exploited by a real attacker. If your tester identified any points last year that remain unaddressed, they will, without a doubt, surface again this year. Do not pay for what you already know!

Pitfalls for new players

Not budgeting for testing, both in terms of time and money.

Knowing what it will cost, how long the testing will take, and how the time needed to fix and re-test any findings is difficult to estimate, so approach your testing providers as soon as possible and ask for help with this.

Not ensuring that implementers or developers are ready to address any security issues discovered.

Your suppliers (for example of the mobile app or the cloud system) need to confirm – in writing - that they will promptly fix any issues identified by testing; this could save you from unplanned and costly variations.

Not including subsequent re-testing in the statement of work.

Always ensure that any testing you procure includes at least one round of re-testing.

Leaving penetration testing to the eleventh hour.

A fairly common scenario on the eve before an application or system deployment is a team member asking if they had completed penetration testing, leading to a mad dash to find someone to test it and often suboptimal or incomplete results.

Common Misconceptions

Blackbox testing is more realistic.

Because of the probable limitations of your testing budget, this is not often the case.

There is no such thing as a secure system.

This trope is commonplace and will probably be uttered at least once by someone during every project. The reality is that secure and not secure is a false dichotomy, and these things exist on a broad spectrum that must also accommodate cost and useability. However, developing a secured system is costly, and, it can be said, at some point, security encroaches on useability. The problem with the no such thing as a secure concept is that it leads to defeatism and a why bother mindset. It is much more advisable to address the obvious security flaws and sleep better at night, knowing your system is secure.

Penetration testers are the same as hackers.

Firstly, penetration testers have permission, but, equally important, they need to excel at much more than the handful of tricks someone who considers themselves a hacker might possess. Primarily though, unlike hackers, penetration testers need the ability and discipline to document and explain what they conducted and found concisely. Penetration testers are far more than just hackers turned good.

Penetration testing is just someone running a tool.

While a penetration tester might use various tools, it is the human element that makes the difference. The ability to understand nuances, think creatively, and both make and test hypotheses are the difference between automated tools and a human-led penetration test. It is unlikely that many people possess Michelangelo's ability to create stunning sculptures out of marble, regardless of how many cool power tools they may own. Without human ingenuity directing them, the power tools are nearly useless.

We can only test in production, so it needs to happen outside regular work hours.

Mature organisations often have test, development, and production environments, allowing testing to happen in a safe sandbox without risk to production users or data.

When a sandbox environment is not available, it is still possible to test in production during working hours with a manageable level of risk, provided you discuss this with your tester ahead of time and be mindful additional time may be needed.

Testers will adjust their approach in a production environment by rate-limiting their tools, treating the environment with kid gloves, and not changing or retrieving an excessive amount of production data.

We have a Nimbus 2000 Web Application Firewall (or other threat protection), so we are secure.

Firewalls and web application firewalls are simply hurdles that can slow an attacker down. Whilst firewalls are, for the most part, useful tools, they should never be considered impenetrable or non-circumventable. A better approach would be to test without these controls in place, and then you can assess if their addition affects the outcome of any issues discovered. This method allows you to inform the risk and likelihood of your findings.

Our IT guys have already tested this.

Many organisations fall under the impression that IT is a single subject and that if you manage one aspect of it, you can manage them all, which is not the case.

If someone, for example, plays the guitar well, can they also play the banjo? What about a sitar or a harp?

You may get lucky and find that you have an IT professional on staff that is a natural musician and can pick up any instrument and play it beautifully. If you do, pay them well, else they will run away to join a superior orchestra the first chance they get. An expert opinion on your security posture through objective eyes is essential for solid security practices.

Other Resources

Trustwave Security Testing Services
https://www.trustwave.com/en-us/services/security-testing/security-testing-services/

Once and Future Threats: What Security Teting is and Will Be
https://www.trustwave.com/en-us/resources/library/documents/once-and-future-threats-what-security-testing-is-and-will-be/

Trustwave Blog: Spotlight On Trustwave Spiderlabs
https://www.trustwave.com/en-us/resources/blogs/trustwave-blog/spotlight-on-trustwave-spiderlabs-part-1-proactive-threat-intelligence/

Trustwave Blog: How to Use Penetration Testing to Improve Your Remote Work Force Security
https://www.trustwave.com/en-us/resources/blogs/trustwave-blog/how-to-use-penetration-testing-to-improve-your-remote-work-force-security/

Trustwave Infographic: Five Ways Attackers Get to Databases
https://www.trustwave.com/en-us/resources/library/documents/five-ways-attackers-get-to-databases/

Trustwave Video: A Red Team Simulation Synopsis – How Trustwave SpiderLabs Conduct a Simulated Attack
https://www.trustwave.com/en-us/resources/library/videos/a-red-team-simulation-synopsis/

The OWASP Top 10
https://owasp.org/www-project-top-ten/

The MITRE ATT&CK Framework
https://attack.mitre.org/

The OSSTMM methodology
https://www.isecom.org/OSSTMM.3.pdf

The NIST Cybersecurity Framework
https://www.nist.gov/cyberframework

The PTES Standard
http://www.pentest-standard.org/index.php/Main_Page

The Council for Registered Ethical Security Testers
https://www.crest-approved.org/

Trustwave is a leading cybersecurity and managed security services provider focused on threat detection and response. Offering a comprehensive portfolio of managed security services, consulting and professional services, and data protection technology, Trustwave helps businesses embrace digital transformation securely. Trustwave is a Singtel company and the global security arm of Singtel, Optus and NCS, with customers in 96 countries.
For more information about Trustwave, visit *www.trustwave.com.*

Trustwave®