TRAINING

CSSLP

2021

AGENDA

Domain 1: Secure Software Concepts

- ➢ Core Concepts of Secure Software
- ➢ Security Design Principles
- ➢ Privacy
- ➢ Governance, Risk, and Compliance (GRC)
- ➢ Software Development Methodologies

“The loftier the building the deeper the foundation must be”.

ISC2 – CSSLP - CBK

QUESTIONS

Few Questions to begin with

What will you do if you find an USB stick/ SD memory card in a public place?

Assume you found a Laptop in a public place, what will you do?

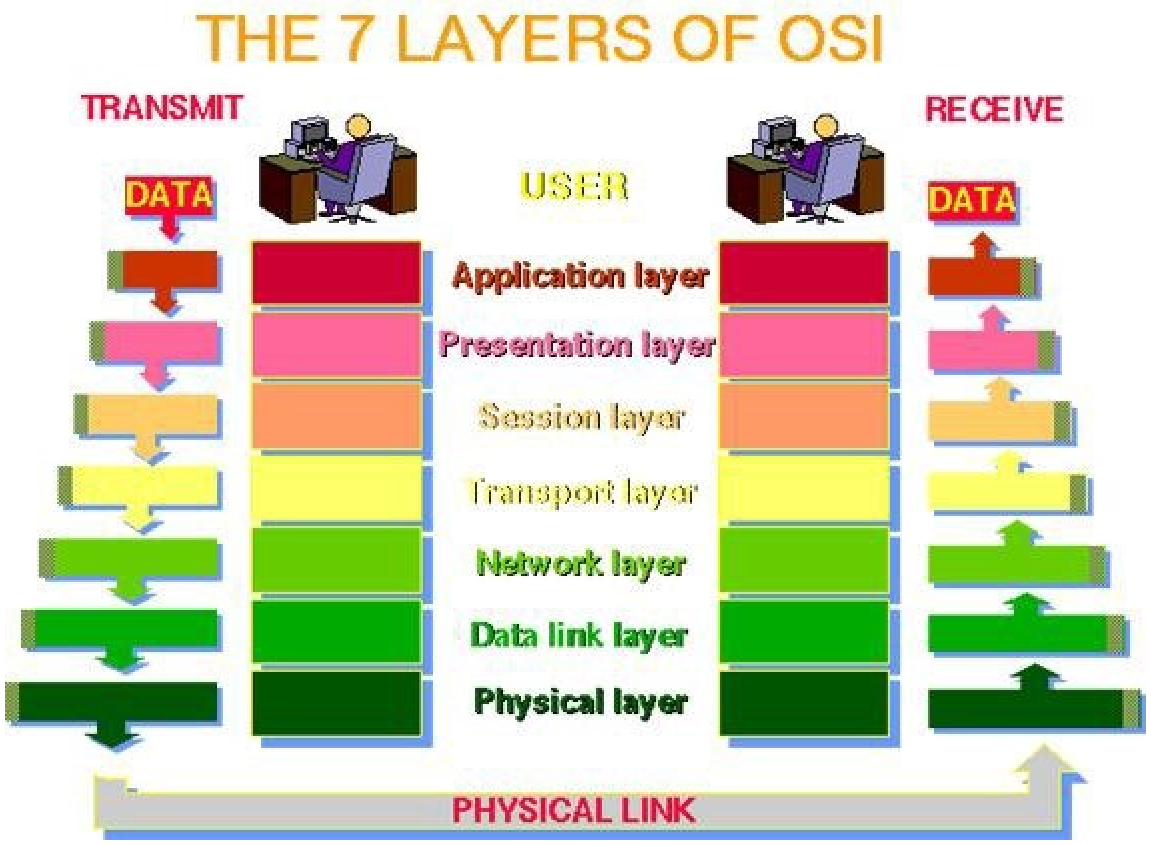
Assume you found out password of your manager, what will you do?

Assume you found out Facebook/Social media password of your Friend/Enemy, what will you do?

What are OSI Model Layers? How many Layers are there?

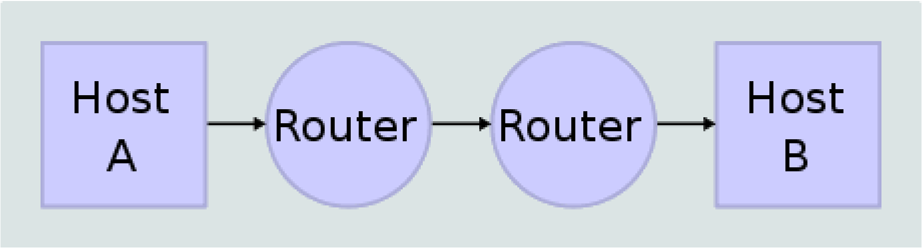
What are TCP/HTTP/Web Sockets?

FUNDAMENTALS

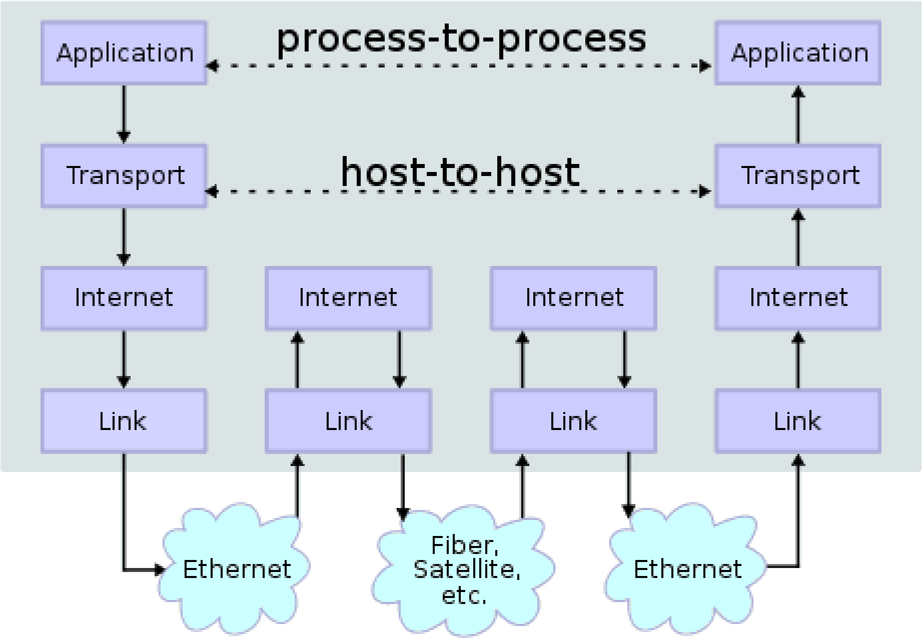


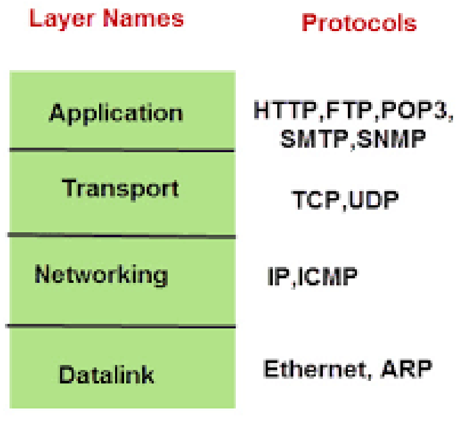
TCP

Network Topology



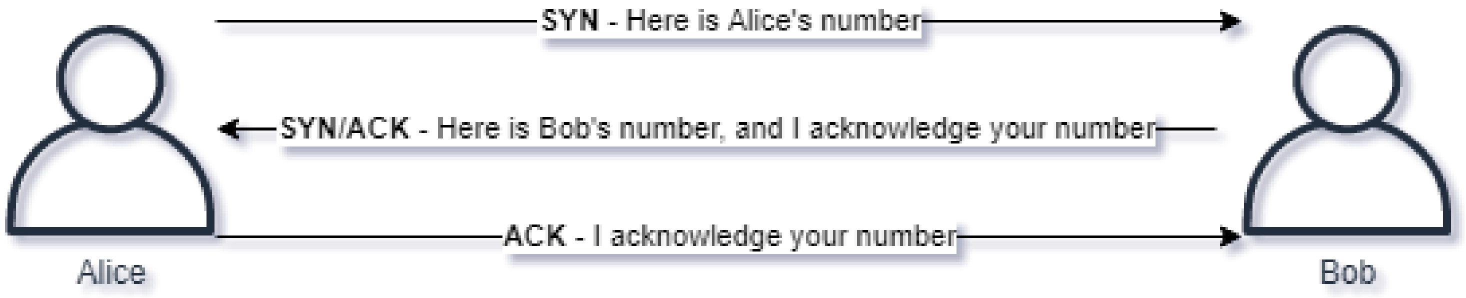
Data Flow





Layer Names	Protocols
Application	HTTP,FTP,POP3, SMTP,SNMP
Transport	TCP,UDP
Networking	IP,ICMP
Datalink	Ethernet, ARP

TCP/IP Networking Model



SYN - Here is Alice's number

SYN/ACK - Here is Bob's number, and I acknowledge your number

ACK - I acknowledge your number

SOFTWARE DEVELOPMENT METHODOLOGIES

Software Development Lifecycle (SDLC) Models

- ➢ Waterfall model
- ➢ Iterative model
- ➢ Spiral model
- ➢ Agile development methodologies

RECENT BREACHES

- https://portswigger.net/daily-swig/data-breach
- https://firewalltimes.com/recent-data-breaches/

Advanced Attack incidents:

- Stuxnet(2010) - Iran's nuclear program https://en.wikipedia.org/wiki/Stuxnet

- Distributed DoS attack on domain name system provider Dyn (Oct 2016)

- https://termly.io/resources/articles/biggest-data-breaches/#top-10-data-breaches-of-all-time-infographic

SECURITY – HUMAN THREAT AGENTS

Ignorant User

The ignorant user is the one that is often the cause of unintentional and plain user error. Plain user error is also referred to sometimes as plain error or simply user error.

Accidental Discoverer

An ordinary user who stumbles upon a functional mistake in the software and who is able to gain privilege access to information or functionality. This user never sought to circumvent security protection mechanisms in the first place.

Curious Attacker

An ordinary user who notices some anomaly in the functioning of the software and decides to pursue it further. Often an accidental discoverer graduates into being a curious attacker.

SECURITY – HUMAN THREAT AGENTS

Insider

One of the most powerful attackers. They enemy inside the firewall. These are potentially disgruntled employees or staff member within the company that has access to insider knowledge.

Ex: The database administrator with unfettered and unaudited access to sensitive information directly is a potential threat source that should not be ignored.

Third Party/Supplier

When software is developed outside the purview of one's company control, then malicious logic and malicious code (malcode) such as logic bombs and Trojan horses can be unintentionally or intentional embedded in the software code, as the software moves through the supply chain. When outsourcing software development, the Foreign

Ownership Control and Influence (FOCI) of the third party or supplier must be determined and code inspection (review) prior to acceptance must be performed by the acquirer.

SECURITY – HUMAN THREAT AGENTS

Script Kiddies

These are those ordinary users who execute hacker scripts against corporate assets without understanding the impact and consequences of their actions. Most elite hackers today were one day script kiddies.

A litmus test to the identification of a script kiddy’s work is that they do not often hide or know how to hide their footprint on the software or systems they have attacked.

Organized Cybercriminals

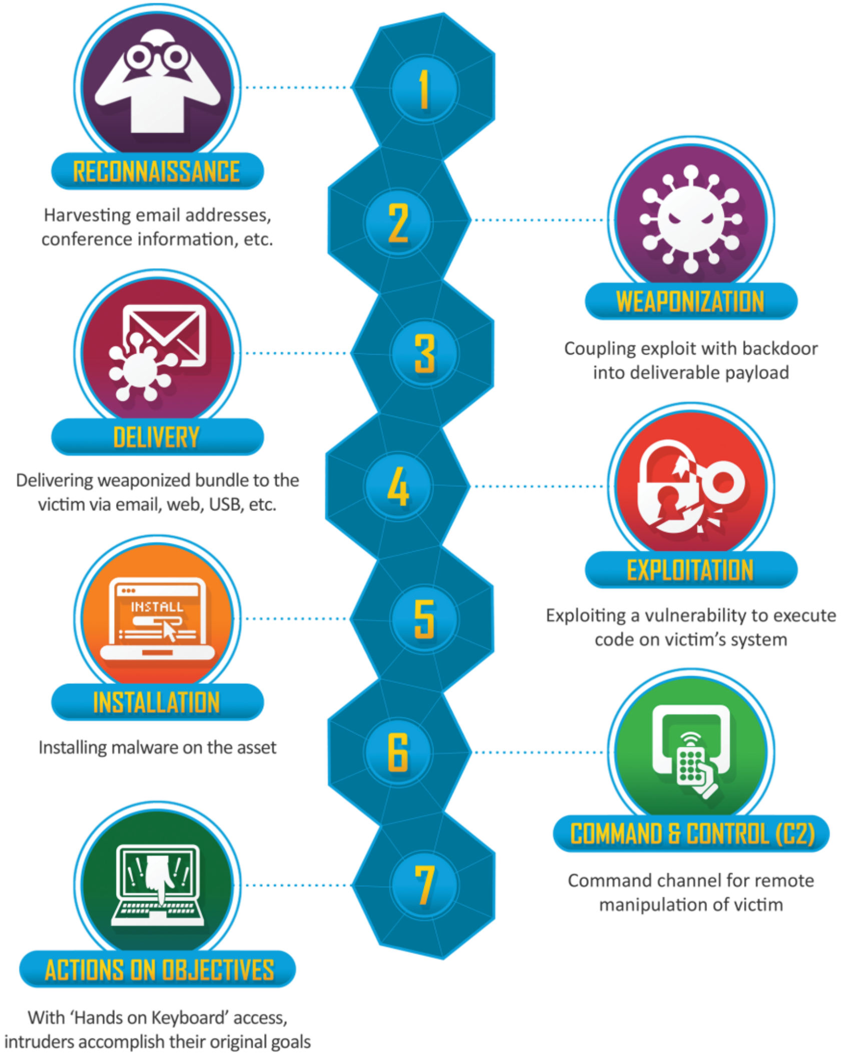
These are highly skilled malefactors that are paid professionally for using their skills to thwart security protection of software and systems, seeking high financial gain.

They not only have a deep understanding of software development, but also of reverse engineering and network and host security controls. They can be used for attacks against corporate assets as well as are a threat to national security as cyber terrorists.

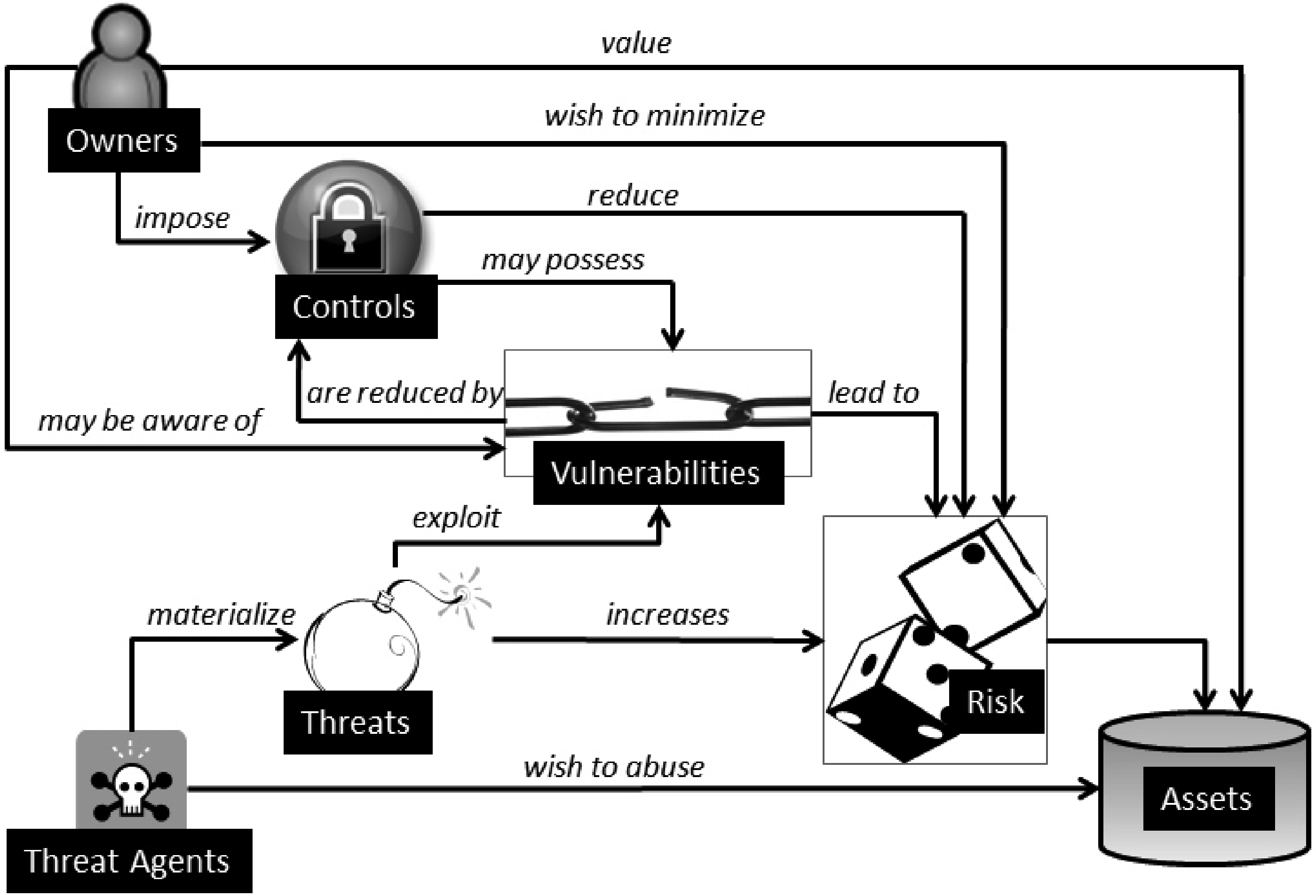
Malware developers and Advance Persistent Threat (APT) hackers usually fall into this category.

APT

The Cyber Kill Chain® framework
https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

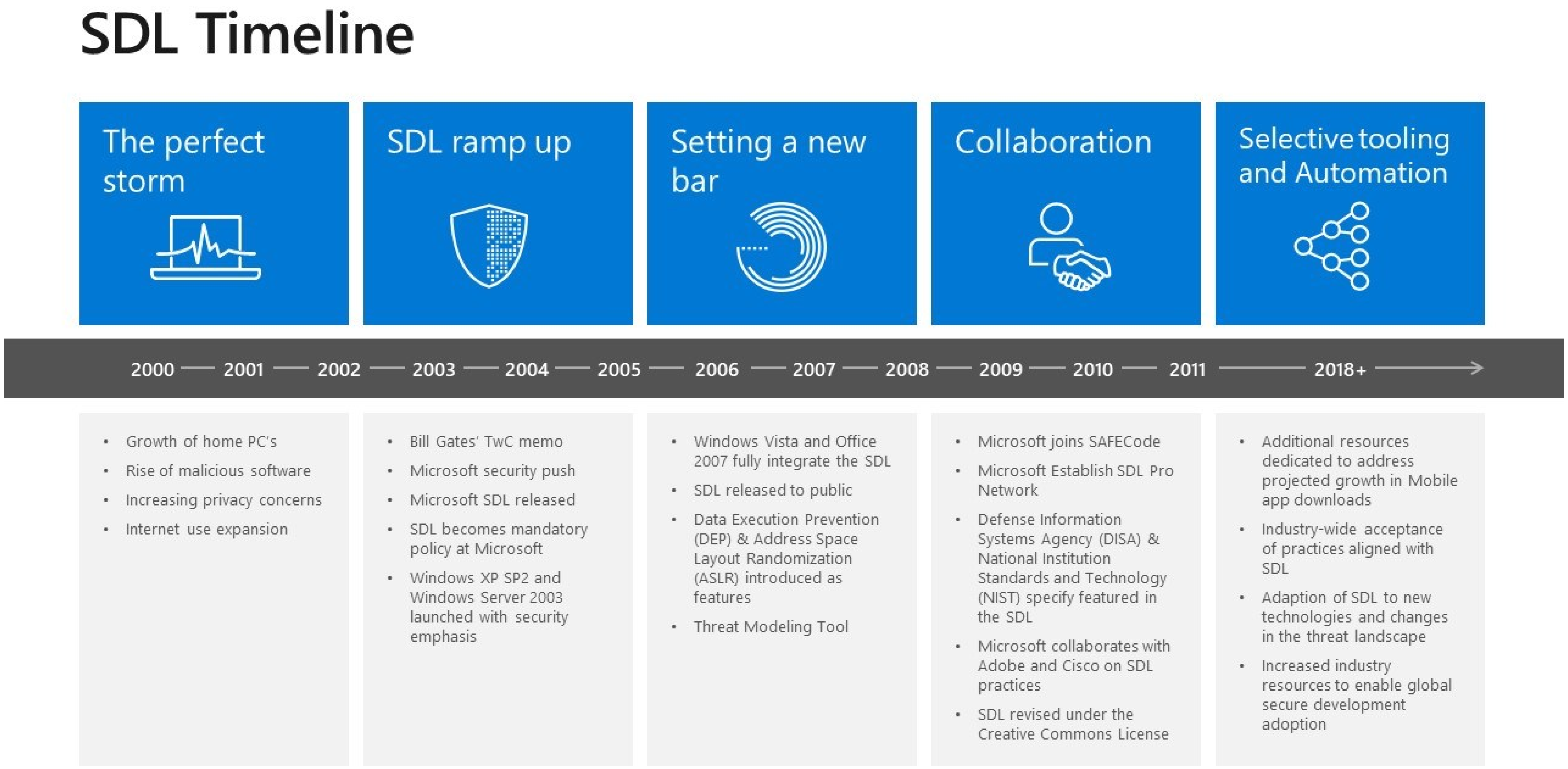

1
RECONNAISSANCE
Harvesting email addresses, conference information, etc.
2
WEAPONIZATION
Coupling exploit with backdoor into deliverable payload
3
DELIVERY
Delivering weaponized bundle to the victim via email, web, USB, etc.
4
EXPLOITATION
Exploiting a vulnerability to execute code on victim's system
5
INSTALLATION
INSTALL
Installing malware on the asset
6
COMMAND & CONTROL (C2)
Command channel for remote manipulation of victim
7
ACTIONS ON OBJECTIVES
With 'Hands on Keyboard' access, intruders accomplish their original goals

RISK MANAGEMENT



INDUSTRY RESPONSE

Example: Microsoft launched its Trustworthy Computing(TwC) initiative – 2002



SDL Timeline

2000 — 2001 — 2002 — 2003 — 2004 — 2005 — 2006 — 2007 — 2008 — 2009 — 2010 — 2011 —— 2018+ ——→

The perfect storm

- Growth of home PC's
- Rise of malicious software
- Increasing privacy concerns
- Internet use expansion

SDL ramp up

- Bill Gates' TwC memo
- Microsoft security push
- Microsoft SDL released
- SDL becomes mandatory policy at Microsoft
- Windows XP SP2 and Windows Server 2003 launched with security emphasis

Setting a new bar

- Windows Vista and Office 2007 fully integrate the SDL
- SDL released to public
- Data Execution Prevention (DEP) & Address Space Layout Randomization (ASLR) introduced as features
- Threat Modeling Tool

Collaboration

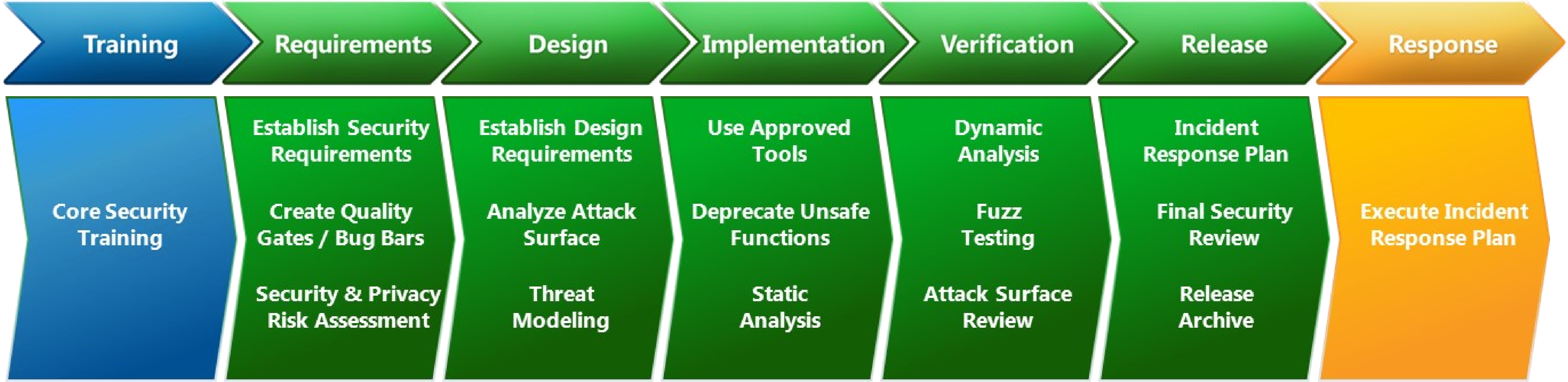
- Microsoft joins SAFECode
- Microsoft Establish SDL Pro Network
- Defense Information Systems Agency (DISA) & National Institution Standards and Technology (NIST) specify featured in the SDL
- Microsoft collaborates with Adobe and Cisco on SDL practices
- SDL revised under the Creative Commons License

Selective tooling and Automation

- Additional resources dedicated to address projected growth in Mobile app downloads
- Industry-wide acceptance of practices aligned with SDL
- Adaption of SDL to new technologies and changes in the threat landscape
- Increased industry resources to enable global secure development adoption

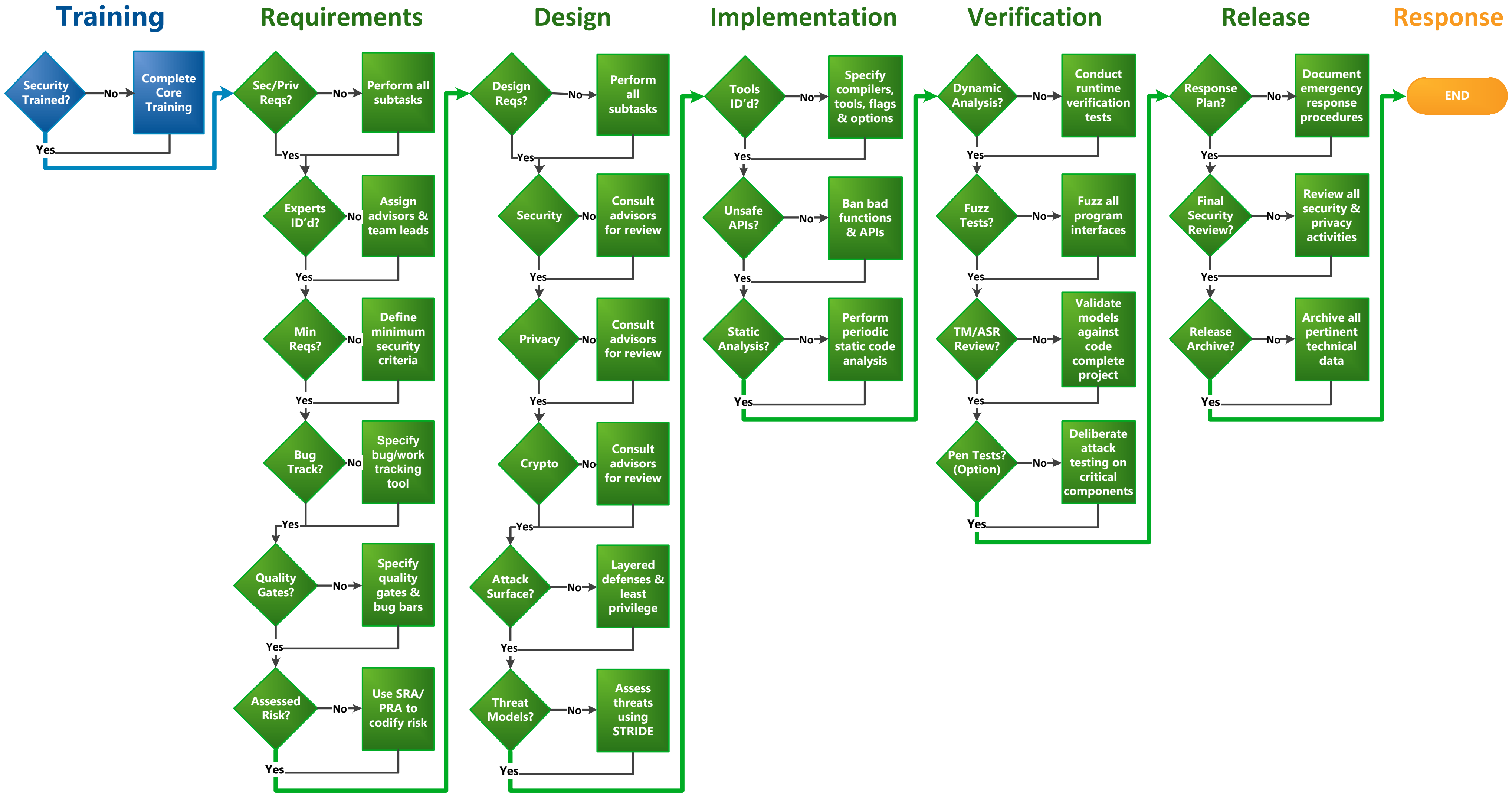
INDUSTRY RESPONSE

Microsoft SDL



Training	Requirements	Design	Implementation	Verification	Release	Response
Core Security Training	Establish Security Requirements Create Quality Gates / Bug Bars Security & Privacy Risk Assessment	Establish Design Requirements Analyze Attack Surface Threat Modeling	Use Approved Tools Deprecate Unsafe Functions Static Analysis	Dynamic Analysis Fuzz Testing Attack Surface Review	Incident Response Plan Final Security Review Release Archive	Execute Incident Response Plan

MICROSOFT SDL



Training

Security Trained? —No→ Complete Core Training

Yes

Requirements

Sec/Priv Reqs? —No→ Perform all subtasks

Yes

Experts ID'd? —No→ Assign advisors & team leads

Yes

Min Reqs? —No→ Define minimum security criteria

Yes

Bug Track? —No→ Specify bug/work tracking tool

Yes

Quality Gates? —No→ Specify quality gates & bug bars

Yes

Assessed Risk? —No→ Use SRA/ PRA to codify risk

Yes

Design

Design Reqs? —No→ Perform all subtasks

Yes

Security —No→ Consult advisors for review

Yes

Privacy —No→ Consult advisors for review

Yes

Crypto —No→ Consult advisors for review

Yes

Attack Surface? —No→ Layered defenses & least privilege

Yes

Threat Models? —No→ Assess threats using STRIDE

Yes

Implementation

Tools ID'd? —No→ Specify compilers, tools, flags & options

Yes

Unsafe APIs? —No→ Ban bad functions & APIs

Yes

Static Analysis? —No→ Perform periodic static code analysis

Yes

Verification

Dynamic Analysis? —No→ Conduct runtime verification tests

Yes

Fuzz Tests? —No→ Fuzz all program interfaces

Yes

TM/ASR Review? —No→ Validate models against code complete project

Yes

Pen Tests? (Option) —No→ Deliberate attack testing on critical components

Yes

Release

Response Plan? —No→ Document emergency response procedures

Yes

Final Security Review? —No→ Review all security & privacy activities

Yes

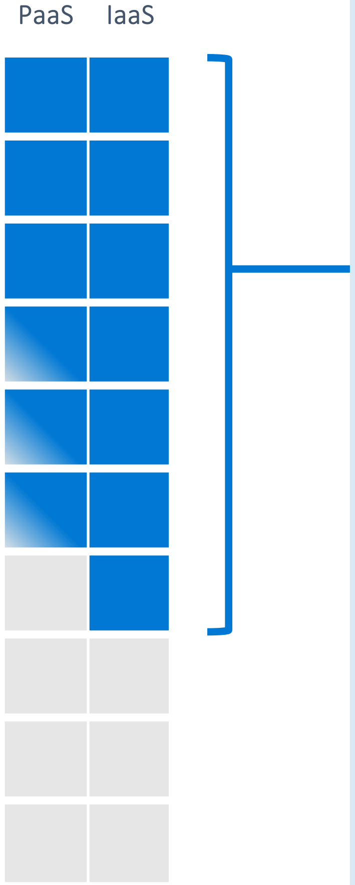
Release Archive? —No→ Archive all pertinent technical data

Yes

Response

END

THREATS - CLOUD APPLICATIONS



EXISTING TECHNIQUES (AT COMPARABLE LEVELS)

EXPLOIT/ENTER	TRAVERSAL	MONETIZATION
SOCIAL ENGINEERING	CREDENTIAL THEFT & ABUSE (HASHES, SSH...)	RANSOMWARE
PHISHING	SCAN & EXPLOIT	TARGETED DATA THEFT
GEO-FILTERING EVASION WITH PROXY		COMMODITY BOTNET/DDOS/ETC
●●●	●●●	●●●

New Techniques (☆) or Very High Usage (⇧)

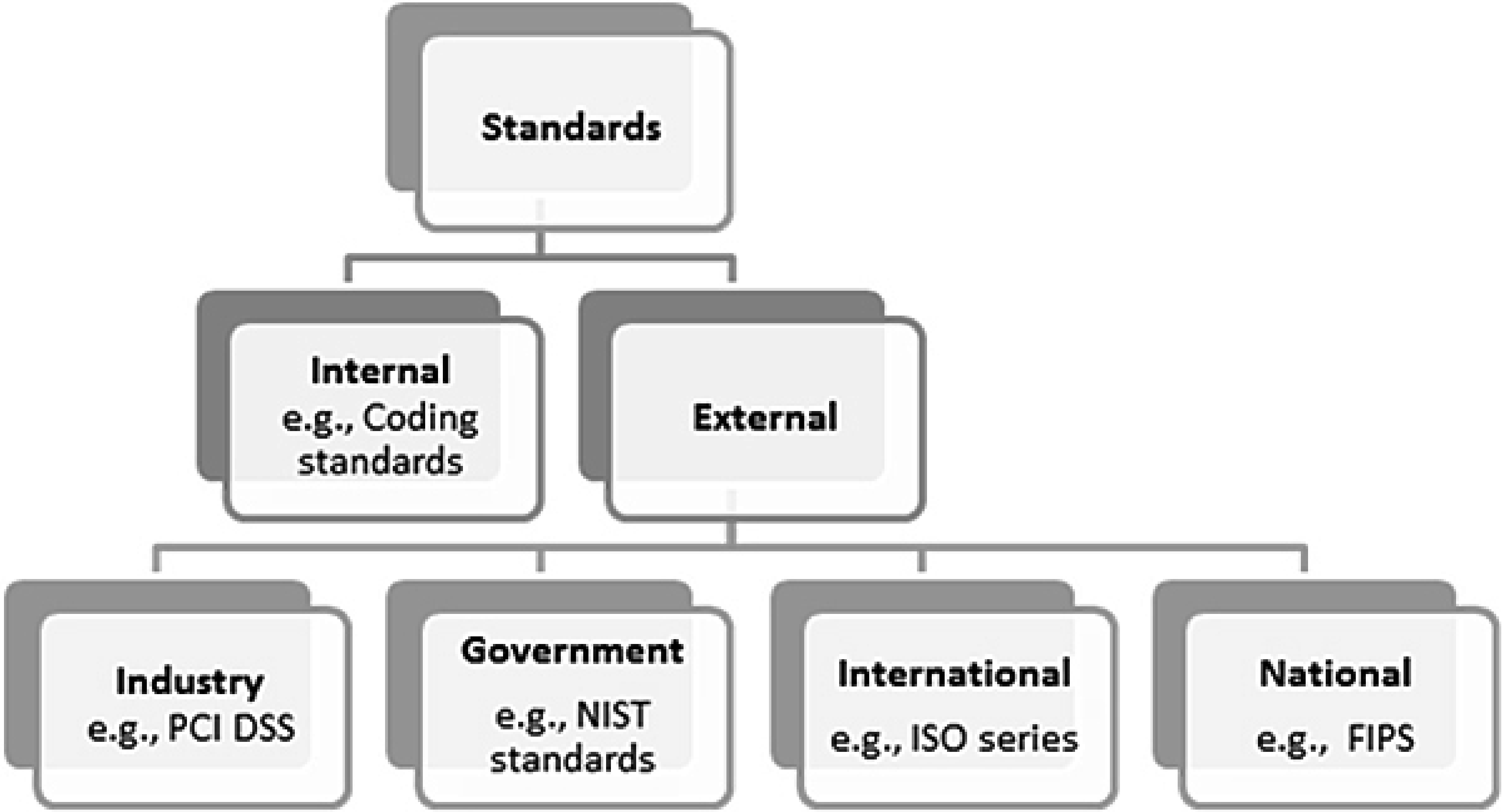
☆ ACQUIRE TENANT KEYS FROM GITHUB/ETC	☆ PIVOT TO ON PREMISES FROM CLOUD	⇧ CRYPTOMINERS – (WEBSERVERS, VISITORS)
⇧ RDP/SSH PASSWORD SPRAY & BRUTE FORCE		

STANDARDS

ISO/IEC 15408 Evaluation Assurance Levels

- EAL1 Functionally tested
- EAL2 Structurally tested
- EAL3 Methodically tested and checked
- EAL4 Methodically designed, tested and reviewed
- EAL5 Semi-formally designed and tested
- EAL6 Semi-formally verified design and tested
- EAL7 Formally verified design and tested

TYPES OF STANDARDS



Standards

Internal
e.g., Coding standards

External

Industry
e.g., PCI DSS

Government
e.g., NIST standards

International
e.g., ISO series

National
e.g., FIPS

STANDARDS

- National Institute of Standards and Technology (NIST)
    - Special Publications
    - Federal Information Processing Standards (FIPS)
- International Organization for Standardization (ISO)
- Payment Card Industry (PCI)
- Organization for the Advancement of Structured Information Standards (OASIS)

STANDARDS

Important NIST Publications

FIPS 200 Minimum Security Requirements for Federal Information and Information Systems

FIPS 199 Standards for Security Categorization of Federal Information and Information Systems

FIPS 197 Advanced Encryption Standard

FIPS 186-3 Digital Signature Standard (DSS)

FIPS 190- 4 Secure Hash Standard (SHS)

FIPS 140 series Security Requirements for Cryptographic Modules

SP 800- 152 A Profile for U.S. Federal Cryptographic Key Management Systems (CKMS)

SP 800- 107 Recommendation for Applications Using Approved Hash Algorithms

SP 800- 100 Information Security Handbook: A Guide for Managers

Continued in next slide…

STANDARDS

Continued from previous slide…

SP 800-64 Security Considerations in the System Development Life Cycle

SP 800-63 Electronic Authentication Guideline

SP 800-53 Security and Privacy Controls for Federal Information Systems and Organizations

SP 800-30 Guide for Conducting Risk Assessments

SP 800-14 Generally Accepted Principles and Practices for Securing Information Technology Systems

SP 800-12 An Introduction to Computer Security: The NIST Handbook

SECURITY – CORE CONCEPTS

Confidentiality (C)

Confidentiality is the security concept that has to do with protection against unauthorized information disclosure

Integrity (I)

1. Ensure that the data that is transmitted, processed and stored is as accurate as the originator intended

2. The software performs reliably as it was intended to.

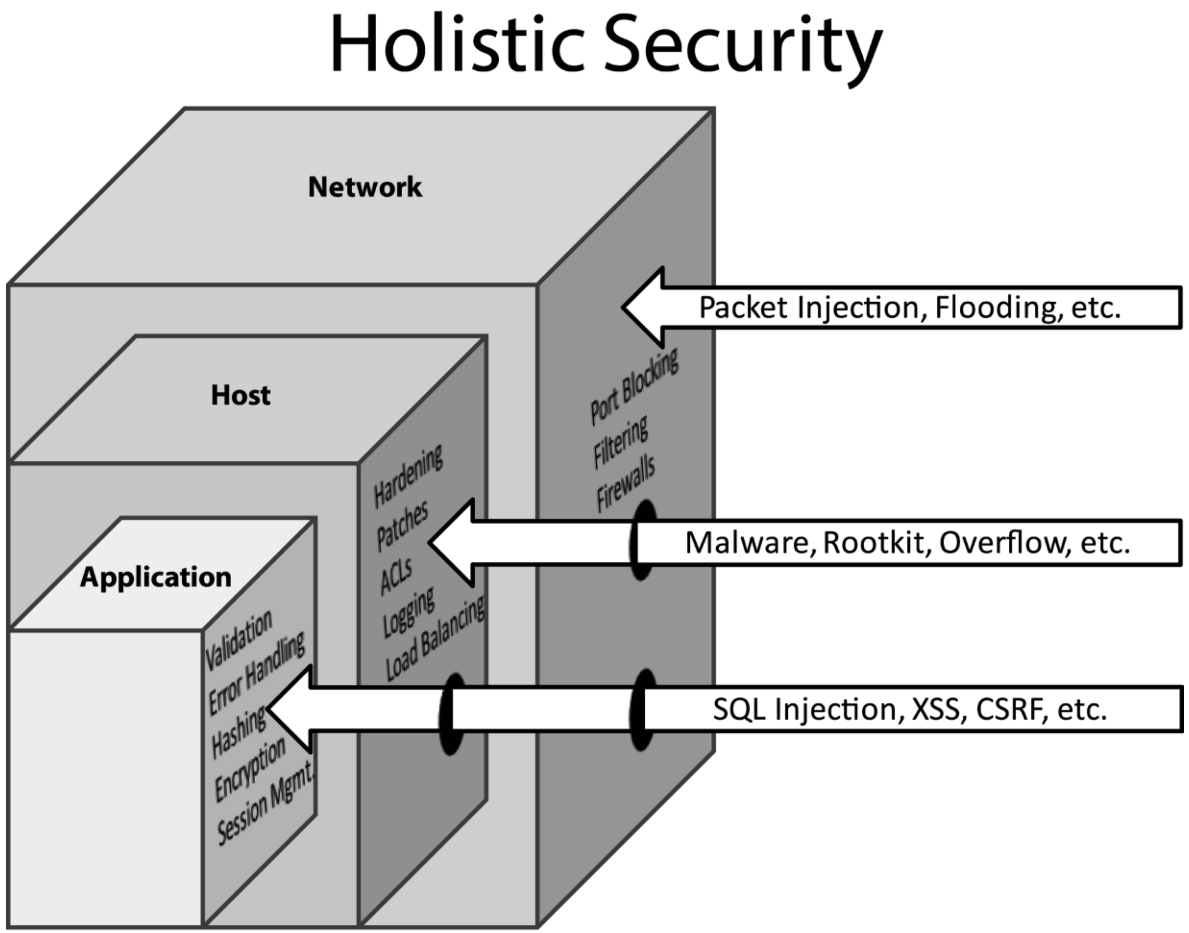
Availability (A)

1. The software or the data it processes must be accessible by only those who are authorized (who)

2. The software or the data it processes must be accessible only at the time (when) that it is required. Data must not be available to the wrong people or at the wrong time.

INTRODUCTION - SECURITY CONCEPTS

What to protect?

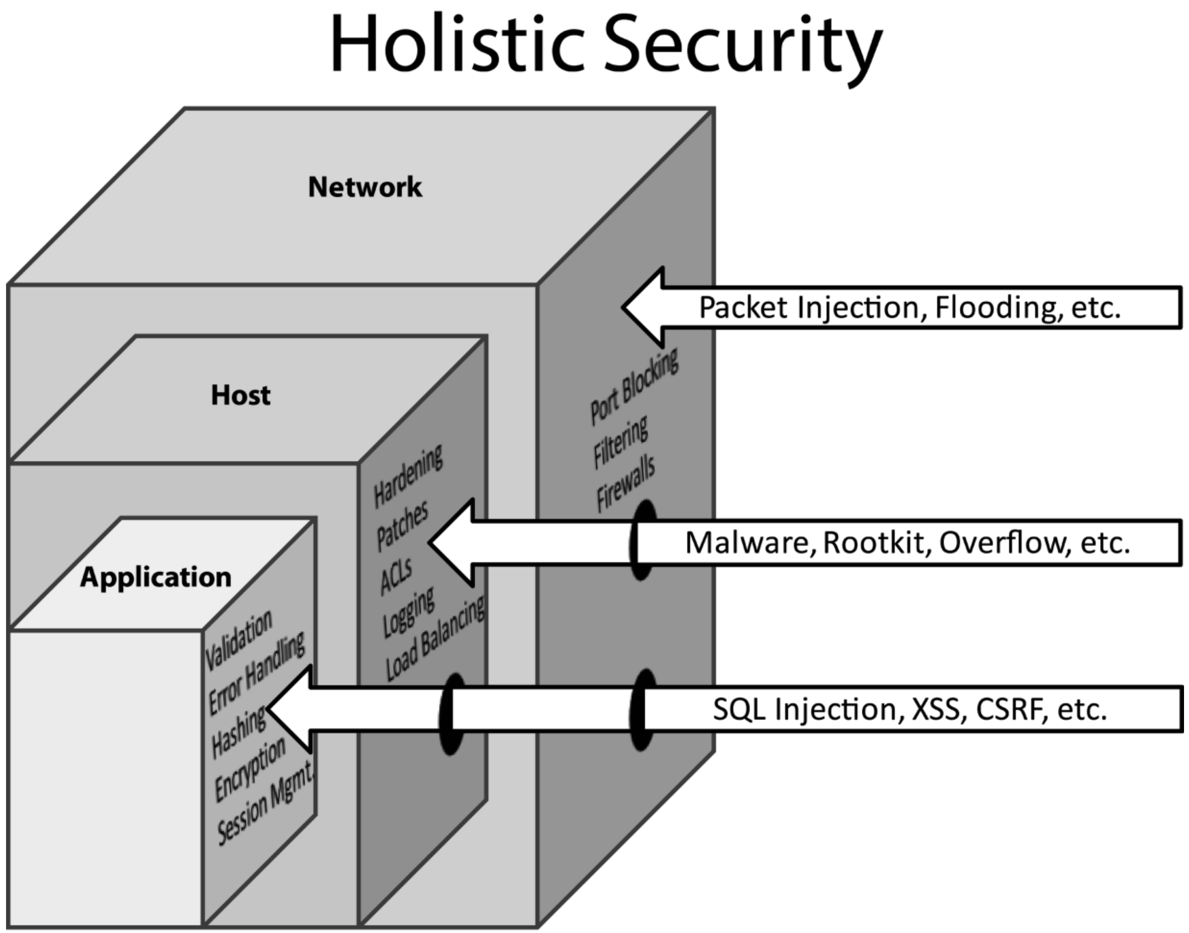


Implementation Challenges

- Project constraints – Time (Schedule), resources (scope) and cost (budget)
- Security as an Afterthought
- Security vs Usability

INTRODUCTION - SECURITY CONCEPTS

What to protect?



Implementation Challenges

- Project constraints – Time (Schedule), resources (scope) and cost (budget)
- Security as an Afterthought
- Security vs Usability

SECURITY – CORE CONCEPTS

Authentication
Identify Human Users, Processes etc

Identity

- Something you know – username and password
- Something you have – Physical lock/key, token
- Something about you (something that you are) – biometrics (Finger prints, voice, retinal scan)

SECURITY – CORE CONCEPTS

Authorization

- **Subject** access **Object**
- Determines access levels of the (**Subject**)
  - Example - User, Process
- **Object** is any resource that is protected
  - Example - a file, a program, an item of data, or any other resource, the authorization system makes the access determination as to grant or deny access.

SECURITY – CORE CONCEPTS

Accountability

- Audit logs (Audit Trials): Build historical record of user actions
- Auditing is a detective and deterrent control

Auditing requirements at the bare minimum must include the following elements

- the identity of the subject (user or process) performing an action (who)
- the action (what)
- the object on which the action was performed (where)
- the timestamp of the action (when)

Repudiation

Denial of a previous action performed

- Accountability with audit trails helps in Non-Repudiation.

Challenges

- Performance impact
- Information overload
- Capacity limitation
- Configuration interfaces protection
- Audit log protection

SECURITY – DESIGN CONCEPTS

Least Privilege

- ✓ The minimum level of access rights (privileges) that is necessary for that person or process to complete an assigned operation.
- ✓ And right must be given only for a minimum amount of time that is necessary to complete the operation.

Separation of Duties or Compartmentalization Principle or Separation of Privilege

- ✓ The successful completion of a single task is dependent upon two or more conditions that need to be met and just one of the conditions will be insufficient in completing the task by itself.

Defense in Depth or Layered Defense

- ✓ Single points of complete compromise are eliminated or mitigated by the incorporation of a series or multiple layers of security safeguards and risk-mitigation countermeasures

Fail Secure or fail safe

- ✓ Aims to maintaining confidentiality, integrity and availability by defaulting to a secure state, rapid recovery of software resiliency upon design or implementation failure

SECURITY – DESIGN CONCEPTS

Economy of Mechanisms or Keep It Simple principle
The likelihood of a greater number of vulnerabilities increases with the complexity of the software architectural design and code. By keeping the software design and implementation details simple, the attackability or attack surface of the software is reduced.

Complete Mediation
A security principle that ensures that authority is not circumvented in subsequent requests of an object by a subject, by checking for authorization (rights and privileges) upon every request for the object. That means, the access requests by a subject for an object is completed mediated each time, every time.

Open Design
The implementation details of the design should be independent of the design itself, which can remain open.

Unlike in the case of security by obscurity wherein the security of the software is dependent upon the obscuring of the design itself.

When software is architected using the open design concept, the review of the design itself will not result in the compromise of the safeguards in the software

SECURITY – DESIGN CONCEPTS

Least Common Mechanisms

The sharing of mechanisms that are common to more than one user or process if the users and processes are at different levels of privilege.

For example, the use of the same function to retrieve the bonus amount of an exempt employee and a non-exempt employee will not be allowed. In this case the calculation of the bonus is the common mechanism.

Psychological Acceptability

- Maximizing the usage and adoption of the security functionality in the software by ensuring that the security functionality is easy to use and at the same time transparent to the user.
- Ease of use and transparency are essential requirements for this security principle to be effective.

Weakest Link

The resiliency of your software against hacker attempts will depend heavily on the protection of its weakest components, be it the code, service or an interface. (Achilles heel)

Leveraging Existing Components

Ensure that the attack surface is not increased and no new vulnerabilities are introduced by promoting the reuse of existing software components, code and functionality.

SECURITY – DESIGN CONCEPTS

Some principles are contradictory to others.
Example :

1. **Economy of Mechanism *vs* Complete Mediation**
2. **Leveraging Existing Components *vs* Leveraging Existing Components**

Architectural decisions must be taken to address these conflicts without compromising the security of the software.

SECURITY - RISK MANAGEMENT

- **Asset** - Assets are those items that are valuable to the organization, the loss of which can potentially cause disruptions in the organization’s ability to accomplish its missions. (Tangible and Intangible assets)
- **Vulnerability** - A weakness or flaw that could be accidently triggered or intentionally exploited by an attacker, resulting in the breach or breakdown of the security policy is known as a vulnerability.
- **Threat** - Vulnerabilities pose threats to assets. A threat is merely the possibility of an unwanted, unintended or harmful event occurring. When the event occurs upon manifestation of the threat, it results in an incident.
- **Threat Source/Agent** - Anyone or anything that has the potential to make a threat materialize is known as the threat-source or threat-agent. Threat agents may be human or non-human.
- **Attack** - an intentional action attempting to cause harm is the simplest definition of an attack.
- **Exploit** - When an attack happens as a result of an attacker taking advantage of a known vulnerability, it is known as an ‘exploit’. The attacker exploits a vulnerability causing the attacker (threat agent) to cause harm (materialize a threat).
- **Probability** - probability is the chance that a particular threat can happen.
- **Impact** - The extent of how serious the disruptions to the organization’s ability to achieve its goal is referred to as the impact.
- **Exposure Factor** - Exposure Factor is defined as the opportunity for a threat to cause loss.

SECURITY - RISK MANAGEMENT

- **Controls** - Security controls are mechanisms by which threats to software and systems can be mitigated.
- **Total Risk** - Total risk is the likelihood of the occurrence of an unwanted, unintended or harmful event. This is traditionally computed using factors such as the asset value, threat, and vulnerability. This is the overall risk of the system, before any security controls are applied. This may be expressed qualitatively (e.g., High, Medium or Low) or quantitatively (using numbers or percentiles).
- **Residual Risk** - Residual risk is the risk that remains after the implementation of mitigating security controls (countermeasures or safeguards).
- **Calculation of Risk**
  - Single Loss Expectancy(SLE) = Asset Value ($) x Exposure Factor (%)
  - Annual Rate of Occurrence (ARO) = ALE = Single Loss Expectancy (SLE) x Annualized Rate of Occurrence (ARO)
- **Risk Handling**
  - Ignore the risk
  - Avoid the risk
  - Mitigate the risk
  - Accept the risk
  - Transfer the risk

STRIDE AND DREAD

STRIDE threats can be grouped and categorized into the following six categories.

- **Spoofing** – Impersonating another user or process
- **Tampering** – Unauthorized alterations that impact integrity
- **Repudiation** – Cannot prove the action; deniability of claim
- **Information Disclosure** – Exposure of information to unauthorized user or process that impact confidentiality
- **Denial of Service** – Service interruption that impacts availability
- **Elevation of privilege** – Unauthorized increase of user or process rights

DREAD is a risk calculation or rating methodology that is often used in conjunction with STRIDE.

- ***Damage potential*** – What will be the impact upon exploitability?
- ***Reproducibility*** – What is the ease of recreating the attack/exploit?
- ***Exploitability*** – What minimum skill level is necessary to launch the attack/exploit?
- ***Affected users*** – How many users will be potentially impacted upon a successful attack/exploit?
- ***Discoverability*** – What is the ease of finding the vulnerability that yields the threat?

ENTERPRISE APPLICATION AND SECURITY FRAMEWORKS

- Zachman Framework
- Control Objectives for Information and related Technology (COBIT®)
- Committee of Sponsoring Organizations (COSO)
- Sherwood Applied Business Security Architecture (SABSA)

BEST PRACTICES

The **Open Web Application Security Project® (OWASP)** is a nonprofit foundation that works to improve the security of software.

OWASP Top 10

- ➢ 10 most critical security concerns for web application security
- ➢ https://owasp.org/www-project-top-ten/

  - ✓ The OWASP Development Guide :
    - ✓ https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/migrated_content

  - ✓ The OWASP Code Review Guide
    - ✓ https://owasp.org/www-project-code-review-guide/

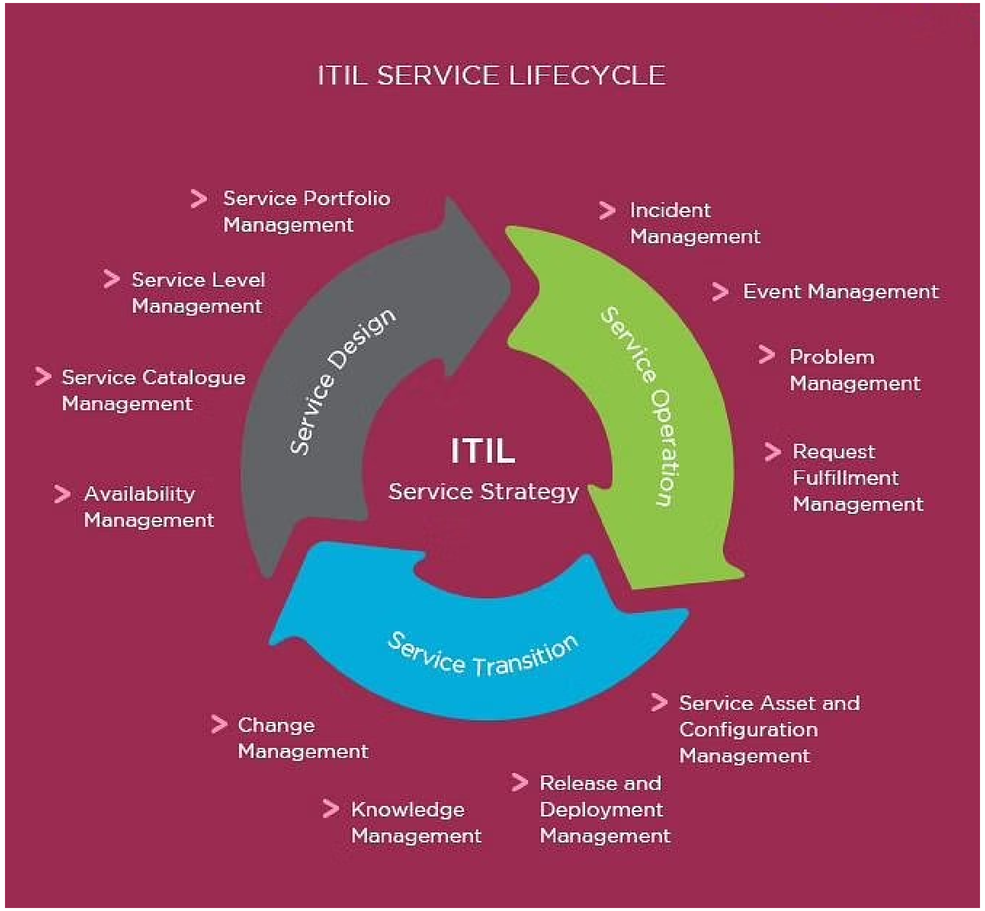
  - ✓ The OWASP Testing Guide
    - ✓ https://owasp.org/www-project-web-security-testing-guide/stable/

  - ✓ Other OWASP Projects
    - ✓ https://owasp.org/projects/

BEST PRACTICES

ITIL - Information Technology Infrastructure Library

The Information Technology Infrastructure Library (ITIL) defines the organizational structure and skill requirements of an information technology organization and a set of standard operational management procedures and practices to allow the organization to manage an IT operation and associated infrastructure. The operational procedures and practices are supplier independent and apply to all aspects within the IT Infrastructure.



SAMPLE QUESTIONS

The PRIMARY reason for incorporating security into the software development life cycle is to protect

A. the unauthorized disclosure of information.
B. the corporate brand and reputation.
C. against hackers who intend to misuse the software.
D. the developers from releasing software with security defects.

==================================

Answer : B

When security is incorporated in to the software development life cycle, confidentiality, integrity and availability can be assured and external hacker and insider threat attempts thwarted. Developers will generate more hack-resilient software with fewer vulnerabilities, but protection of the organization’s reputation and corporate brand is the primary reason for software assurance.

SAMPLE QUESTIONS

The resiliency of software to withstand attacks that attempt modify or alter data in an unauthorized manner is referred to as

A. Confidentiality.
B. Integrity.
C. Availability.
D. Authorization.

==================

Answer : B

When the software program operates as it is expected to, it is said to be reliable or internally consistent. Reliability is an indicator of the integrity of software. Hack resilient software are reliable (functioning as expected), resilient (able to withstand attacks) and recoverable (capable of being restored to normal operations when breached or upon error).

SAMPLE QUESTIONS

The MAIN reason as to why the availability aspects of software must be part of the organization’s software security initiatives is:

A. software issues can cause downtime to the business.

B. developers need to be trained in the business continuity procedures.

C. testing for availability of the software and data is often ignored.

D. hackers like to conduct Denial of Service (DoS) attacks against the organization.

==================

Answer : A

One of the tenets of software assurance is ‘availability’. Software issues can cause software unavailability and downtime to the business. This is often observed as a denial of service (DoS) attack.

SAMPLE QUESTIONS

Developing the software to monitor its functionality and report when the software is down and unable to provide the expected service to the business is a protection to assure which of the following?
A. Confidentiality.
B. Integrity.
C. Availability.
D. Authentication.
==================

Answer : C

Confidentiality controls assures protection against unauthorized disclosure.
Integrity controls assures protection unauthorized modifications or alterations.
Availability controls assures protection against downtime/denial of service
and destruction of information.
Authentication is the mechanism to validate the claims/credentials of an entity.
Authorization has to do with rights and privileges that a subject has upon requested objects.

Questions

THANK YOU