2022

Class #5 – Domain 3

Security Architecture and Engineering

Ryan Cloutier CISSP®
President of SecurityStudio & vCISO

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

FRSECURE CISSP MENTOR PROGRAM LIVE STREAM

THANK YOU!

Quick housekeeping reminder.

- The online/live chat that's provided while live streaming on YouTube is for constructive, respectful, and relevant (about course content) discussion **ONLY**.
- At **NO TIME** is the online chat permitted to be used for disrespectful, offensive, obscene, indecent, or profane remarks or content.
- Please do not comment about controversial subjects, and please **NO DISCUSSION OF POLITICS OR RELIGION**.
- Failure to abide by the rules may result in disabling chat for you.
- **DO NOT share or post copywritten materials. (pdf of book)**

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

INTRODUCTION

Agenda –

- Welcome
- Introduction
- Security Architecture
- Security Engineering
- Security Models
- Security Controls
- Systems overview
- Cryptography

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION THREE

HELLO, NICE TO MEET YOU

Ryan Cloutier, CISSP, Tonight's Instructor

@cloutiersec
@StudioSecurity

- President of SecurityStudio
- Virtual Chief Information Security Officer
- Serving the underserved is my passion
- Speaking human about tech is my superpower
- Co-host of the Security Shit Show, and Security Simplified podcast
- Infosec Missionary (helper and protector at heart)
- Published by multiple trade magazines
- Co authored academic papers
- Advisor to many

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

GETTING GOING...

Managing Risk!

Study Tips:

- Study in small amounts frequently (20-30 min)
- Flash card and practice test apps help
- Take naps after heavy topics (aka Security Models)
- Write things down, say them out loud
- Use the Slack Channels
- Exercise or get fresh air in between study sessions

Let's get going!

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

GETTING GOING…

Great job last week! We're through Domain 2 (Asset Security)

- Shout Out to Ron Woerner for a great class!

- Ready for more?
  - Study often
  - USE the Slack channel to connect with others
  - Ask questions in #questions in slack
- Check-in.
- How many have read Domain 3 & started on Domain 4?

Let's get going!

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

Warning! (lots to cover, lots to memorize, long class)

We are covering 184 slides tonight this one will run long

You must read the book and memorize most of this content

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DAD JOKE

Before we get too deep into this.

How about a dumb dad joke?

I received a verifiable threat against my Boston cream pie

HAHAHAHA

Moving on…

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

Security Architecture Is

Design and organization of the components, processes, services, and controls appropriate to reduce the security risks associated with a system to an acceptable level.

Security Engineering Is

Implementation of that design

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

Introduction

- The goal is protecting confidentiality, integrity, and availability of the systems or business in addition to Privacy and other important principals
- Conduct a comprehensive risk assessment to gain an accurate idea of the risks to be addressed.
- Once risks are identified and assessed the security architecture can begin.
- Risk treatments
  - Avoid
  - Transfer or share (i.e., insurance or contract)
  - Mitigate (e.g., through security architecture)
  - Accept

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

Risk assessment

- Initial risk assessment identifies the risks to be reduced through the design of a security architecture to incorporate appropriate security controls.
- An assessment must be made to confirm that the resulting system's risks have been reduced to an acceptable level.
- Cost associated with certain controls can be prohibitive related to anticipated benefit.
- Decision to reduce certain risks may need to be reconsidered, and those risks treated in another manner, avoided through a system redesign, or the project simply abandoned.

*Reminder the **cost of a security control, must be less than the cost of the risk** being addressed

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

Introduction

- Security serves to protect the business. The work of the security architect is to ensure the business and its interests at the very least are protected according to applicable standards and laws, as well as meeting any relevant regulatory compliance needs.
- There is a tendency to concentrate on technical security controls and attempt to address all known security issues or requirements
- Security for security's sake, while intellectually satisfying, is a disservice to the organization.
- Always remember we first serve as subject matter experts, aware of relevant regulations or laws and capable of ensuring our organization's compliance wherever change is required.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

Introduction

- Organization's security strategy must align with its mission, goals, objectives, and compliance environment.

- Success in security architecture is much more likely when one is aligned with the business and taking a risk management approach to security architecture.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

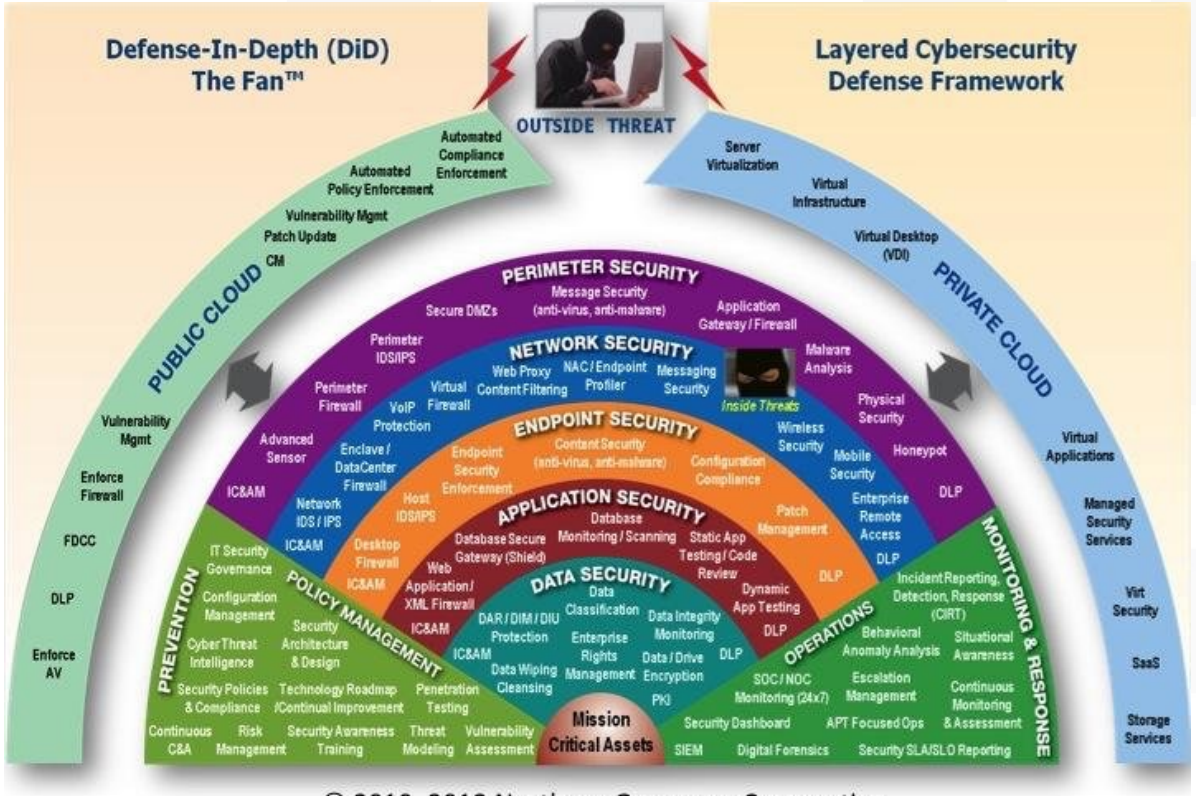
CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

- Design
- Development
- Testing
- Implementation
- Maintenance
- Decommissioning



© 2010, 2012 Northrop Grumman Corporation

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

- It is less expensive to incorporate security when the overall functional system design is developed rather than trying to add it on later (which will often require redesign, if not reengineering, of already developed components).
- The need for security controls is not just to prevent the user from performing unauthorized actions, but to prevent components of the system itself from violating security requirements when acting on the user's requests.
- If security is not intrinsic to the overall design, it is not possible to completely mediate all the activities that can compromise security.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

Fundamental to any security architecture, regardless of the design principles employed, are the basic requirements outlined in 1972 by James Anderson in Computer Security Technology Planning Study (USAF):

- Security functions need to be implemented in a manner that prevents their being bypassed, circumvented, or tampered with.
- Security functions need to be invoked whenever necessary to implement the security control. Security functions need to be as small as possible so that defects are more likely to be found.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Domain separation

- Placing components that share similar security attributes, such as privileges and access rights, in a domain.
- Only permitting separate domains to communicate over well-defined and (completely) mediated communication channels (e.g. application programming interfaces, or APIs).

Real World Examples

- A network is separated into manageable and logical segments. Network traffic (inter-domain communication) is handled according to policy and routing control, based on the trust level and workflow between segments.
- Data is separated into domains in the context of classification, categorization, and security baseline. Even though data might come from disparate sources, if that data is classified at the same level, the handling and security of that classification level (domain) is accomplished with like security attributes.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Layering

- Hierarchical structuring of a system into different levels of abstraction, with higher levels relying upon services and functions provided by lower levels, and lower levels hiding (or abstracting) details of the underlying implementation from higher levels.
- Layering is seen in network protocols, starting with the classic OSI seven-layer model running from physical through to application layers.
- In software systems, one encounters operating system calls, upon which libraries are built, upon which we build our programs. Within the operating system, higher-level functions (such as filesystem functions) are built upon lower-level functions (such as block disk I/O functions).

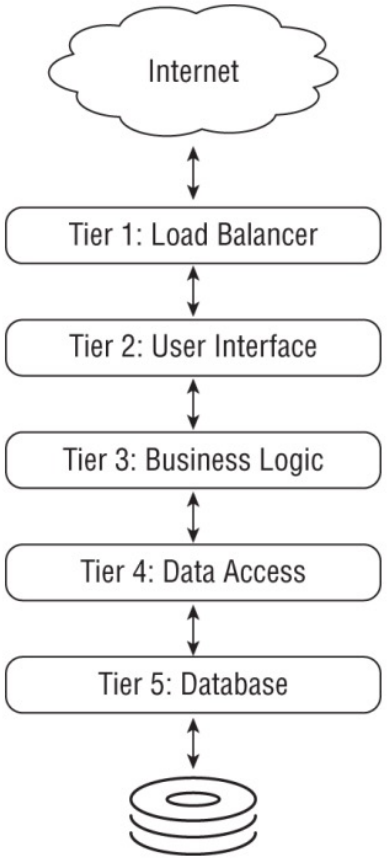
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

The purpose of layering is to do the following:

- Create the ability to impose specific security policies at each layer
- Simplify functionality so that the correctness of its operation is more easily validated

From a security perspective:

- Higher levels always have the same or less privilege than a lower level. If layering to provide security controls, it must not be possible for a higher level to bypass an intermediate level.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Encapsulation

- An architectural concept where objects are accessed only through functions that logically separate functions that are abstracted from their underlying object by inclusion or information hiding within higher level objects.
- Encapsulation functions can define the security policy for that object and mediate all operations on that object.
- Encapsulation requires that all access or manipulation of the encapsulated object must go through the encapsulation functions, and that it is not possible to tamper with the encapsulation of the object or the security attributes (e.g., permissions) of the encapsulation functions.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Encapsulation

- Device drivers can be considered to use a form of encapsulation in which a simpler and consistent interface is provided that hides the details of a particular device.
- Forcing interactions to occur through the abstract object increases the assurance that information flows conform to the expected inputs and outputs.

- An example where encapsulation is used in the real world is the use of the setuid bit. Typically, in Linux or any Unix-based operating system, a file has ownership based on the person who created it, and an application runs based on the person who launched it. A special mechanism, setuid, allows for a file or object to be set with different privileges. Setting the setuid bit on a file will cause it to open with the permission of whatever account you set it to be. The setuid bit controls access, above and beyond the typical operation. That is an example of encapsulation.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Redundancy

- Designing a system with replicated components, operating in parallel, so that the system can continue to operate in spite of errors or excessive load.
- From a security perspective, redundancy is an architectural principle for addressing possible availability and integrity compromises or issues.
- For redundancy to work, it must be possible for the overall system to detect errors in one of the replicated subsystems.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Redundancy examples

- High availability solutions such as a cluster, where one component or system takes over when its active partner becomes inaccessible
- Having storage in redundant array of inexpensive disks (RAID) configurations where the data is made redundant and fault tolerant
- Cloud-based storage, where data is replicated across multiple data centers, zones, or regions

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 architectural principles)

Virtualization

- Is a form of emulation in which the functionality of one real or simulated device is emulated on a different one. (This is discussed in more detail in the "Understand Security Capabilities of Information Systems" section later in this chapter.)
- More commonly, virtualization is the provision of an environment that functions like a single dedicated computer environment but supports multiple such environments on the same physical hardware.
- Virtualization involves abstracting the underlying components of hardware or software from the end user.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Security Architecture

RESEARCH, IMPLEMENT, AND MANAGE ENGINEERING PROCESSES USING SECURE DESIGN PRINCIPLES

ISO/IEC 19249 (5 <

- Least privilege
- Attack surface minimization
- Centralized parameter validation
- Centralized general security services
- Preparing for error and exception handling

The principle of least privilege asserts that access to information should only be granted on an as-needed basis

The more entry points, the greater the attack surface.

Full parameter validation is especially important when dealing with user input, or input from systems to which users input data.

Simplifying your security services interface instead of managing multiple interfaces is a sensible benefit.

Systems must ensure that errors are detected, and appropriate action taken

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling

Process to identify security threats and vulnerabilities, and prioritize mitigations

Used to reduce risk and guide secure development.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling STRIDE (6 categories)

- Design
- Development Testing
- Implementation
- Maintenance
- Decommissioning

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling

Process to identify security threats and vulnerabilities, and prioritize mitigations

Used to reduce risk and guide secure development.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling STRIDE (6 categories)

- **Spoofing** Spoofing is an attack during which a person or system assumes the identity of another person or system by falsifying information.
- **Tampering** Data tampering is an attack on the integrity of data by maliciously manipulating data.
- **Repudiation** Repudiation is the ability of a party to deny that they are responsible for performing an action. Repudiation threat occurs when a user claims that they did not perform an action, and there is no evidence to prove otherwise.
- **Information disclosure** Information disclosure — commonly referred to as a data leak — occurs when information is improperly shared with an unauthorized party
- **Denial of service** A denial-of-service (DoS) attack involves a malicious actor rendering a system or service unavailable by legitimate users.
- **Elevation of privilege** Elevation of privilege (or privilege escalation) occurs when an unprivileged application user can upgrade their privileges to those of a privileged user (such as an administrator).

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling STRIDE (6 categories)

- **Spoofing** [illegible] em assumes the identity of another person or system by falsifying information.
- **Tampering** Data tampering is an attack on the integrity of data by [illegible]
- **Repudiation** [illegible] y are responsible for performing an action. Repudiation threat occurs whe [illegible] n action, and there is no evidence to prove otherwise.
- **Information disclosure** Information disclosure — commonly referre[illegible] information is improperly shared with a[illegible]
- **Denial of service** A denial [illegible] s actor rendering a system or service unavailable by legitimate users.
- **Elevation of privilege** Elevation of privilege (or privilege esca[illegible]tion) o[illegible] user can upgrade their privileges to those of a privileged user (such as an administra[illegible]

Strong passwords, multifactor authentication, and digital signatures are common controls to protect against spoofing.

Strong access controls and thorough logging and monitoring are good ways to prevent and detect data tampering.

Digital signatures and secure logging and auditing are the primary controls to provide nonrepudiation.

Encryption, data loss prevention (DLP), and strong access controls are common controls to protect against information disclosure.

System redundancy, network filtering, and resource limits are common protections against DoS attacks.

Strong access control and input validation are common protections against privilege escalation.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling STRIDE (6 categories)

- **Spoofing** Spoofing em assumes the identity of another person or system by falsifying information.
- **Tampering** Dat
- **Repudiation** R
  Repudiation threat occ
- **Information d**
  information is improper
- **Denial of service** A denial s actor rendering a system or service unavailable by legitimate users.
- **Elevation of privilege** Elevation of privilege (or privilege esca tion) o
  user can upgrade their privileges to those of a privileged user (such as an administra

Strong passwords, multifactor authentication, and digital signatures are common controls to protect against spoofing

controls and thorough toring are good ways to etect data tampering.

ning an action.
dence to prove otherwise.

ta loss prevention (DLP), ess controls are common tect against information disclosure.

System redundancy, network filtering, and resource limits are common protections against DoS attacks.

Strong access control and input validation are common protections against privilege escalation.

QUESTION TO ASK

WHAT CAN GO WRONG??

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling DREAD (5 key points)

- **D**amage What is the total amount of damage the threat is capable of causing to your business?
- **R**eproducibility How easily can an attack on the particular threat be replicated?
- **E**xploitability How much effort is required for the threat to be exploited by an attacker?
- **A**ffected users How many people, either inside or outside of your organization, will be affected by the security threat?
- **D**iscoverability How easily can the vulnerability be found?

Uses a numeric value for rating severity of security threats (1-10)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling DREAD (5 key points)

- **D**amage What is the
- **R**eproducibility
- **E**xploitability Hov
- **A**ffected users
  threat?
- **D**iscoverability

Uses a numeric v

security

D = 4
R = 3
E = 8
A = 5
D = 9
Risk Sum = 29

*There are many opinions on the relative importance of each of the categories within DREAD, and many security professionals disagree with a model that weights each category equally

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling PASTA (7 steps) (Process for Attack Simulation and Threat Analysis)

- **Define objectives** During this first stage, key business objectives are defined, and critical security and compliance requirements are identified.
- **Define technical scope** During this stage, the boundaries of the technical environment and the scope of all technical assets for which threat analysis is needed are defined. In addition to the application boundaries, you must discover and document all infrastructure, application, and software dependencies.
- **Application decomposition** During this stage, an evaluation of all assets (i.e., the application components) needs to be conducted, and the data flows between these assets need to be identified. As part of this process, all application entry points and trust boundaries should be identified and defined. This stage is intended to establish a clear understanding of all data sources, the parties that access those data sources, and all use cases for data access within the application
- **Threat analysis** This stage is intended to identify and analyze threat information from within the system, such as SIEM feeds, web application firewall (WAF) logs, etc., as well as externally available threat intelligence that is related to the system.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling PASTA (7 steps) (Process for Attack Simulation and Threat Analysis)

- **Define objectives** During this first stage, key busine… objec… compliance requirements are identified.
  - a preliminary business impact analysis (BIA) is conducted to identify potential business impact considerations.
- **Define technical scope** During th… …vironment and the scope of all technical assets for which th… …ne application boundaries, you must discover and document all … …pendencies.
  - The goal is to capture a high-level but comprehensive view of all servers, hosts, devices, applications, protocols, and data that need to be protected.
- **Application decomposition** During this stage, an evalu… components) needs to be conducted, and the data flows between these a… this process, all application entry points and trust boundaries should be i… intended to establish a clear understanding of all data sources, the partie… use cases for data access within the application.
  - in other words, who should perform what actions on which components of the application.
- **Threat analysis** This stage is… …ormation from within the system, such as SIEM feeds, web application fir… available threat intelligence that is related to the system.
  - The output of this stage should include a list of the most likely attack vectors for the given application or system.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling PASTA (7 steps) (Process for Attack Simulation and Threat Analysis)

- Vulnerability analysis During this stage, all vulnerabilities within the application's code should be identified and correlated to the threat-attack scenarios identified in step 4. Operating system, application, network, and database scans should be conducted, and dynamic and static code analysis results should be evaluated to enumerate and score existing vulnerabilities
- Attack enumeration During this stage, attacks that could exploit identified vulnerabilities (from step 5) are modeled and simulated. This helps determine the likelihood and impact of each identified attack vector.
- Risk and impact analysis During this final stage, your business impact analysis (from step 1) should be refined based on all the analysis performed in the previous six steps.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Threat Modeling PASTA (7 steps) (Process for Attack Simulation and Threat Analysis)

- **Vulnerability analysis** D…ing thi… …tion's code should be identified and correlated to the threat… …ating system, application, network, and database scans should… …ode analysis results should be evaluated to enumerate and sc…
- **Attack enumeration** Dur…g this … …s (from step 5) are modeled and simulated. Thi… …ntified attack vector.
- **Risk and impact analysis** Du… …m step 1) should be refined based on all the analysis performed in the previous six steps.

The primary output of this stage is a correlated mapping of all threat-attack vectors to existing vulnerabilities and impacted assets.

After this stage, your organization should have a strong understanding of your application's attack surface (i.e., what bad things could happen to which assets within your application environment).

Risks should be prioritized for remediation, and a risk mitigation strategy should be developed to identify countermeasures for all residual risks.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Secure Defaults (secure-by-default) (SBD)

- The concept of secure defaults (or secure-by-default) essentially means that systems should be designed with the best security possible without users needing to turn on security features or otherwise think about security configurations.
- Secure-by-default means that a system's default configuration includes the most secure settings possible, which may not always be the most highly functioning settings.
- Systems and applications should be designed such that the end user (or system admin) must actively choose to override secure configurations based on the business's needs and risk appetite.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Fail Security

- For some systems, a fail-open design, where systems continue to allow access when exceptions occur, may be preferable to ensure that access to important information remains readily available during a system error or exception. Conversely, a fail-secure (also known as a fail-safe or fail-closed) system blocks access by default, ensuring that security is prioritized over availability.

- For systems with sensitive data, security controls should be designed such that in the absence of specific configuration settings to the contrary, the default is to not permit the action. Access should be based on permission (e.g., allowed list), not exclusion (e.g., blocked list)

- *This is the principle behind “deny all” default firewall rules and also relates to the concept of least privileged

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Fail Security

- If an error is detected, the system fails in a deny (or safe) state of higher security rather than failing in an open, less secure state.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Separation of Duties (SOD)

- Requires two (or more) actions, actors, or components to operate in a coordinated manner to perform a security sensitive operation.

- Breaking up a process into multiple steps performed by different individuals or requiring two individuals to perform a single operation together (known as dual control) forces the malicious insider to collude with multiple insiders to compromise the system.
- More robust and less susceptible to failure

- *Separation of duties can also be viewed as a defense-in-depth control; permission for sensitive operations should not depend on a single condition.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Keep it Simple

- "Complexity is the worst enemy of security. The more complex you make your system, the less secure it's going to be, because you'll have more vulnerabilities and make more mistakes somewhere in the system. … The simpler we can make systems, the more secure they are." – Bruce Schneier

- "If complexity is the worst enemy of security, then simplicity must be its ally" – Evan Francen

- "Simple is securable, complex is chaos waiting to happen" –Ryan Cloutier

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Keep it Simple

- Complexity is the enemy of security. The simpler and smaller the system, the easier it is to design, assess, and test. When the system as a whole cannot be simplified sufficiently, consider partitioning the problem so that the components with the most significant risks are separated and simplified to the extent possible. This is the concept behind a security kernel — a small separate subsystem with the security-critical components that the rest of the system can rely upon.
- By separating security functionality into small, isolated components, the task of carefully reviewing and testing the code for security vulnerabilities can be significantly reduced.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Keep

- Complexi … design, assess, and test. Whe … e problem so that the comp … possible. This is the concept b … omponents that the rest of th …
- By separa … eviewing and testing the code …

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Trust, but (then) verify

- “trust, but verify” mantra is widely used to describe situations that require an extra layer of verification and accountability.
- In the information security world, “trust, but verify” has been used to describe the use of perimeter firewalls and other controls that use a party's identity, location, and other characteristics to verify that they are a trusted user or system from a trusted location
- In other words, “trust, but verify” assumes everything behind your corporate firewall is safe and verifies that anything passing through that firewall into your network is safe to allow in essentially, “verify once, trust forever.” This model of security has become less-preferred in recent years, in favor of the zero trust model (discussed in the next section).

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Trust, but (then) verify

- Another way to think of this is, inspect what you expect
- Audits are the bedrock of verify
- 3rd parties (partners, cloud providers, or anyone else outside of your organization)
- Trust has to be earned and maintained (“no set-and-forget”)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Impl… Design Principles

Trust, bu…

- Another way…
- Audits are th…
- 3rd parties (p… r organization)
- Trust has to …

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Zero Trust (coined by John Kindervag)

- **Always verify -** Authenticate and authorize every access request based on user identity, location, system health (e.g., patch levels), data classification, user behavior analytics, and any other available data points.

- **Use least privilege access -** Always assign the minimum rights required for the specific access requested, on a Just in Time (JIT) basis.

- **Assume breach -** Instead of trusting devices on your network, assume the worst-case scenario (i.e., that you've already been breached) and minimize the blast radius to prevent further damage.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Zero Trust (coined by John Kindervag)

- Identities should always be verified and secured with strong multifactor authentication, or MFA, wherever possible
- Devices that access your network should be inspected both for identity verification and also to ensure their health status and compliance with your organization's security requirements prior to granting access.
- Remember that users and devices shouldn't be trusted just because they're on an internal network.
- All internal communications should be encrypted, access should be limited to least privilege, by policy, and microsegmentation should be employed to contain threats.
- **Microsegmentation** is a network security technique that involves dividing large network segments into smaller zones to isolate resources from one another and minimize lateral movement by users.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Zero Trust (coined by John Kindervag)

- Detective controls play a big part in a successful zero trust architecture
- Deploy real-time monitoring to help detect and stop attacks and other anomalous behavior
- Real-time analytics can also help inform access decisions by providing real-time context for access requests and supporting JIT permissions

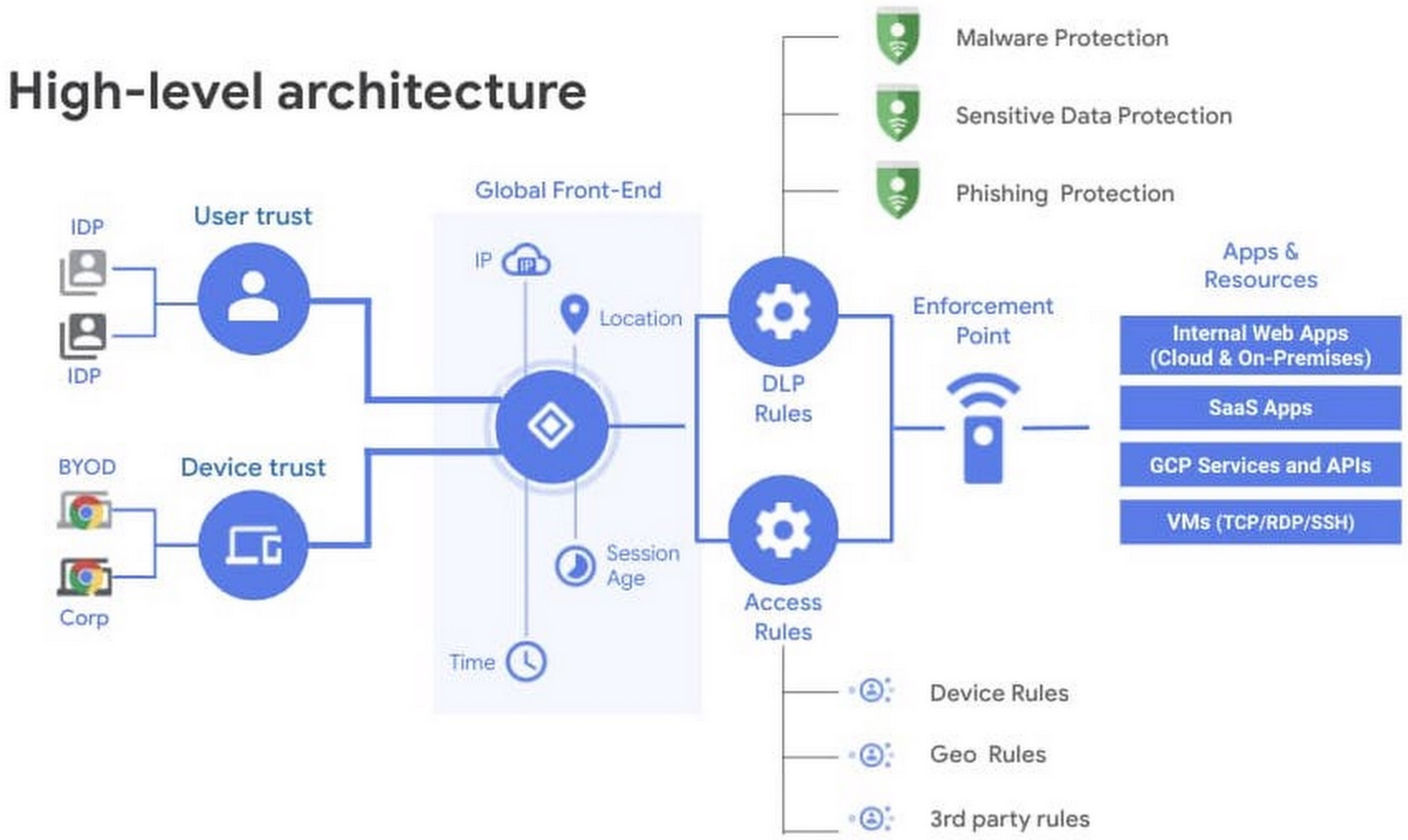
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Zero Trust (coined by John Kindervag)



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Privacy by Design (PbD)

- **Proactive not Reactive; Preventative not Remedial** - Anticipate and prevent invasive privacy events before they happen, rather than relying on detecting and responding to them once they already occur.
- **Privacy as the Default Setting -** Practical examples include anonymizing or masking personal information, restricting access to personal information to those who absolutely need it (i.e., least privilege), and deleting such information when it is no longer needed.
- **Privacy Embedded into Design -** privacy should be treated as a core functionality of the system
- **Full Functionality — Positive-Sum, not Zero-Sum -** PbD encourages a “win-win” approach to all legitimate system design goals and discourages unnecessary trade-offs being made. Both privacy and security are important — both can and should be achieved.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Privacy by Design (PbD)

- **End-to-End Security — Full Lifecycle Protection -** Ensure security and privacy of personal data from cradle to grave; data should be created, managed, and destroyed in a secure fashion. Encryption and authentication are standard at every stage, but you should pay close attention to what security and privacy mechanisms may be required throughout the data lifecycle.

- **Visibility and Transparency — Keep it Open -** This is a “trust, but verify” principle (discussed earlier) that seeks to assure all stakeholders that the system operates securely and maintains data privacy as intended. (e.g., Privacy policy)

- **Respect for User Privacy — Keep it User-Centric -** System architects, developers, and operators must keep the interests of the individual as their utmost priority by providing strong privacy defaults, appropriate notice, and a user-friendly experience. (e.g., by clicking a button or ticking a check box) in order to give consent.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Privacy by Design (PbD)

- FYI-
- In the United States, the Federal Trade Commission (FTC) has recognized PbD as one of three recommended practices for protecting online privacy.
- The European Union General Data Protection Regulation (EU GDPR), the largest privacy regulation around the world to-date, includes “data protection by design” and “data protection by default” as part of its requirements.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Shared Responsibility

- The shared responsibility model is a cloud security framework that describes the obligations of a cloud service provider (CSP) and its customers in keeping cloud systems and data secure.
- In a cloud environment, the CSP takes on much of the operational burden, including a great deal of security responsibility — but not all of it.
- The specific breakdown of responsibility varies by cloud provider and by cloud service type.
- *Your organization is ultimately responsible and accountable for the security of the cloud

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Defense in Depth (layered security)

- Concept of applying multiple, distinct layers of security technologies and strategies to achieve greater overall protection.
  - By using combinations of security controls, the impact from the failure of any single control can be reduced if not eliminated.
  - Layering is another method of separating system components: security controls are placed between the layers, preventing an attacker who has compromised one layer from accessing other layers.
  - Having overlapping security controls such that the failure or compromise of one does not by itself result in an exposure or compromise
  - Related to the concept of assumption of breach, which means managing security on the assumption that one or more security controls have already been compromised.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Research, Implement, and Manage Engineering Processes Using Secure Design Principles

Defense in Depth (layered security)

- The assumption of breach mindset shifts thinking from being simply focused on defending the perimeter (or perimeters) to a balanced approach of establishing multiple defenses so that the compromise of one control does not immediately lead to a successful breach and of considering detection and mitigation to be as important as prevention

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DAD JOKE

Before we get too deep into this.

How about a dumb dad joke?

I like telling Dad jokes…

HAHAHAHA

Moving on…

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Model -** Is a hypothetical abstraction of a system, simplified to enable analysis of certain aspects of the system without the complexity and details of the entire system being analyzed
- **Security Model -** Is a model that deals with security policy.
- Can be **formal**, intended for mathematical analysis to assist in the verification that a system complies with a specific policy.
- Can be **informal**, serving to illustrate and simplify the assessment of a system without the rigor of a proof
- Can help **reduce ambiguity** and potential **misunderstanding** as to what, exactly, a security architecture is trying to accomplish

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Finite state machine** (or just state machine) - is a conceptual computer that can be in one of a finite number of states. The computer implements a state transition function that determines the next state, given the current state and the next input and that can, optionally, produce output.

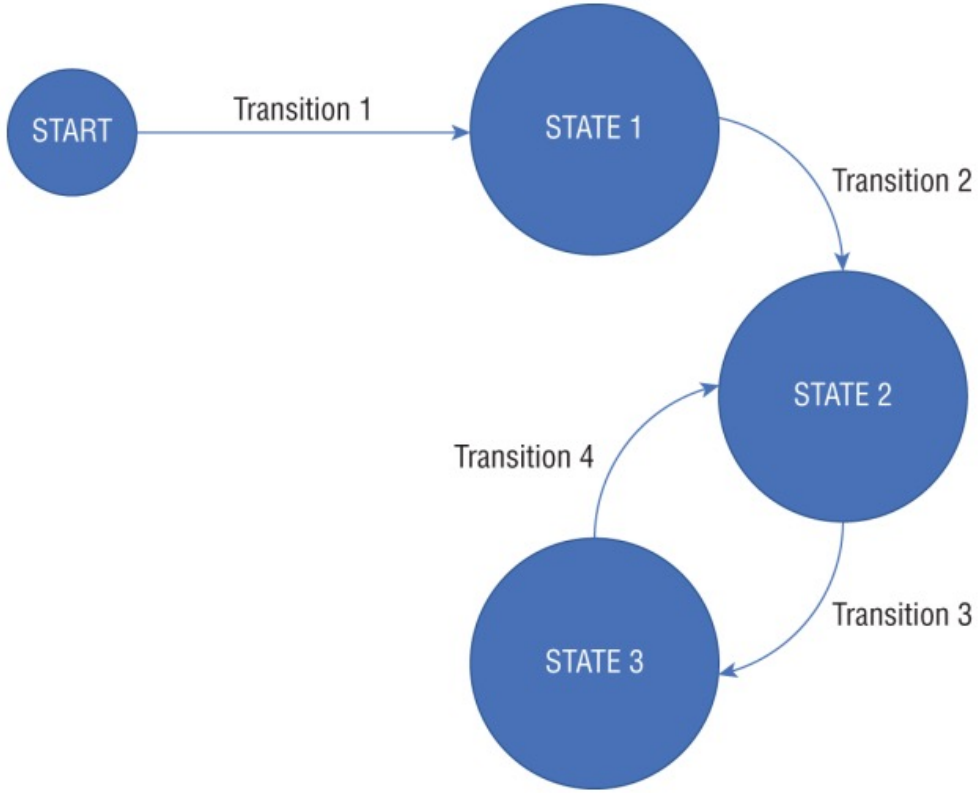
- **A lattice is a finite set with a partial ordering -** partial ordering is a binary relation that is reflexive, anti-symmetric, and transitive. Reflexive means that each item in the set is comparable to itself. Anti-symmetric means that no two different elements precede each other. Transitive means that if a yields b, and b yields c, then a yields c.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Lattice security model does the following

- Defines a set of security levels
- Defines a partial ordering of that set
- Assigns every subject (e.g., user or process) and object (e.g., data) a security level
- Defines a set of rules governing the operations a subject can perform on an object based on the relationship between the security levels of the subject and object

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Information Flow Model

- An information flow model is a type of access control model that defines the flow of information — from one application to another or even one system to another.
- In these models, objects are assigned a security classification, and the direction or type of flow of these objects is controlled by security policy.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Noninterference Model

- A noninterference model is an evolution of the information flow model designed to ensure that objects and subjects on a system don't interfere with other objects and subjects on the same system.
- Under this model, any activities that take place at a higher security level must not impact (or interfere) with activities occurring at a lower level
- The actions of subject A (higher classification) should not affect subject B (lower classification), nor should those actions even be noticed by subject B, in a way that might inform subject B of subject A's actions.
- Without the protection of a noninterference model, subject B might be able to glean the activities of subject A, which may result in information leakage (for example).

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Bell-LaPadula Model (3 rules)

Simple Security Property (ss property) - **No read up**, this rule prevents a subject from reading an object at a higher security level.

Star Property (* property) - **No write down**, this rule prevents a subject from writing to an object at a lower security level.

Discretionary-Security Property - Subject can perform an operation on an object if **permitted by the access matrix**

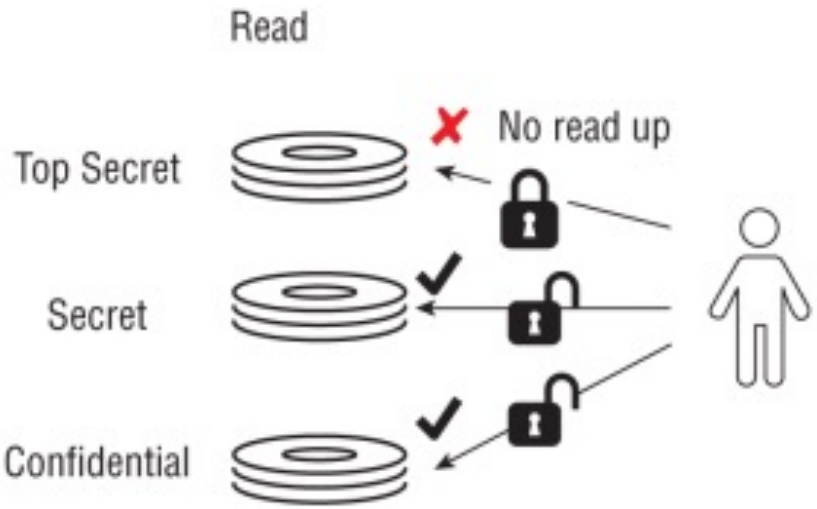
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

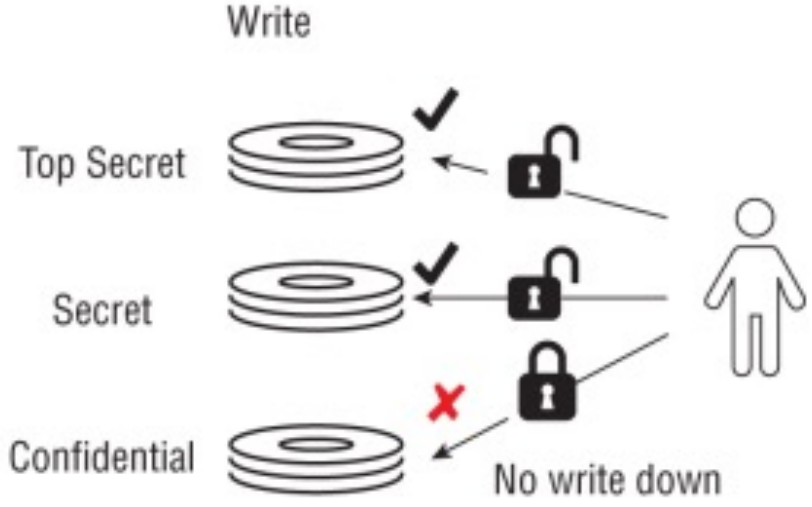
Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Bell-LaPadula Model (3 rules)



Simple Security Property



The Star Property

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Bell-LaPadula Model (3 rules)

Issues not well addressed by the Bell-LaPadula Model

- It does not consider risks to the integrity of information. Protecting the integrity of objects means preventing the unauthorized, possibly malicious, modification of an object.

- Does not deal with covert channels or the possibility of performing permitted operations in a manner that reveals confidential information through side channels

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Bilba Integrity Model**
- Simple Integrity Property - **No read down**, this rule prevents compromising the integrity of more secure information from a less secure source. In other words, higher integrity processes could produce untrustworthy results if they read and use data from lower integrity sources.

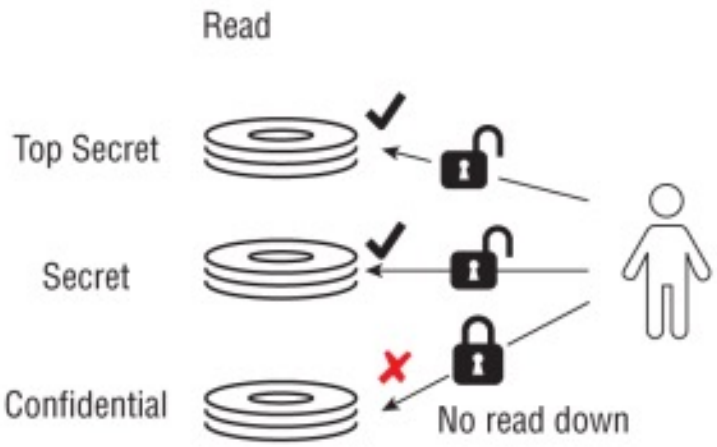
- Star Integrity Property (* integrity property) - **No write up**, this rule prevents the corruption of more secure information by a less privileged subject.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

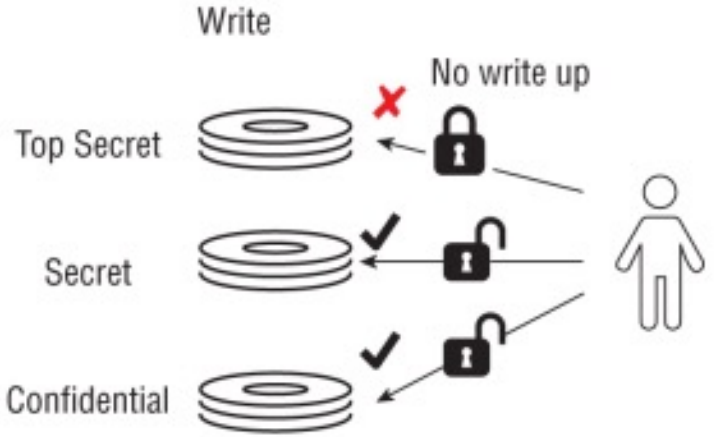
DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components



Simple Integrity Property



Star Integrity Property

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Clark-Wilson Model (2 concepts)

- **Well-formed transactions -** Well-formed transaction is that subjects are constrained to make only those changes that maintain the integrity of the data.
- **Separation of duties -** Aims to make sure that the certifier of a transaction is a different party from the initiator or implementer of the transaction.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Clark-Wilson Model (2 concepts)

- **Constrained data item (CDI) -** This is the key data type in the Clark– Wilson model, and it refers to data whose integrity must be preserved.
- **Unconstrained data item (UDI) -** This includes all data other than CDIs, typically system inputs.
- **Integrity verification procedures (IVPs) -** These procedures check and ensure that all CDIs are valid.
- **Transformation procedures (TPs) -** These procedures enforce a system's integrity policy and maintain the integrity of CDIs.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Brewer-Nash Model**
- Simple Integrity Property - **No read down**, this rule prevents compromising the integrity of more secure information from a less secure source. In other words, higher integrity processes could produce untrustworthy results if they read and use data from lower integrity sources.
- Star Integrity Property (* integrity property) - **No write up**, this rule prevents the corruption of more secure information by a less privileged subject.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Brewer-Nash Model**
- Individual pieces of information related to a single company or client are called objects, in keeping with BLP's usage.
- All objects related to the same company (or client) are part of what is called a company data set.
- All company data sets in the same industry (i.e., that are competitors) are part of what is called a conflict of interest class.

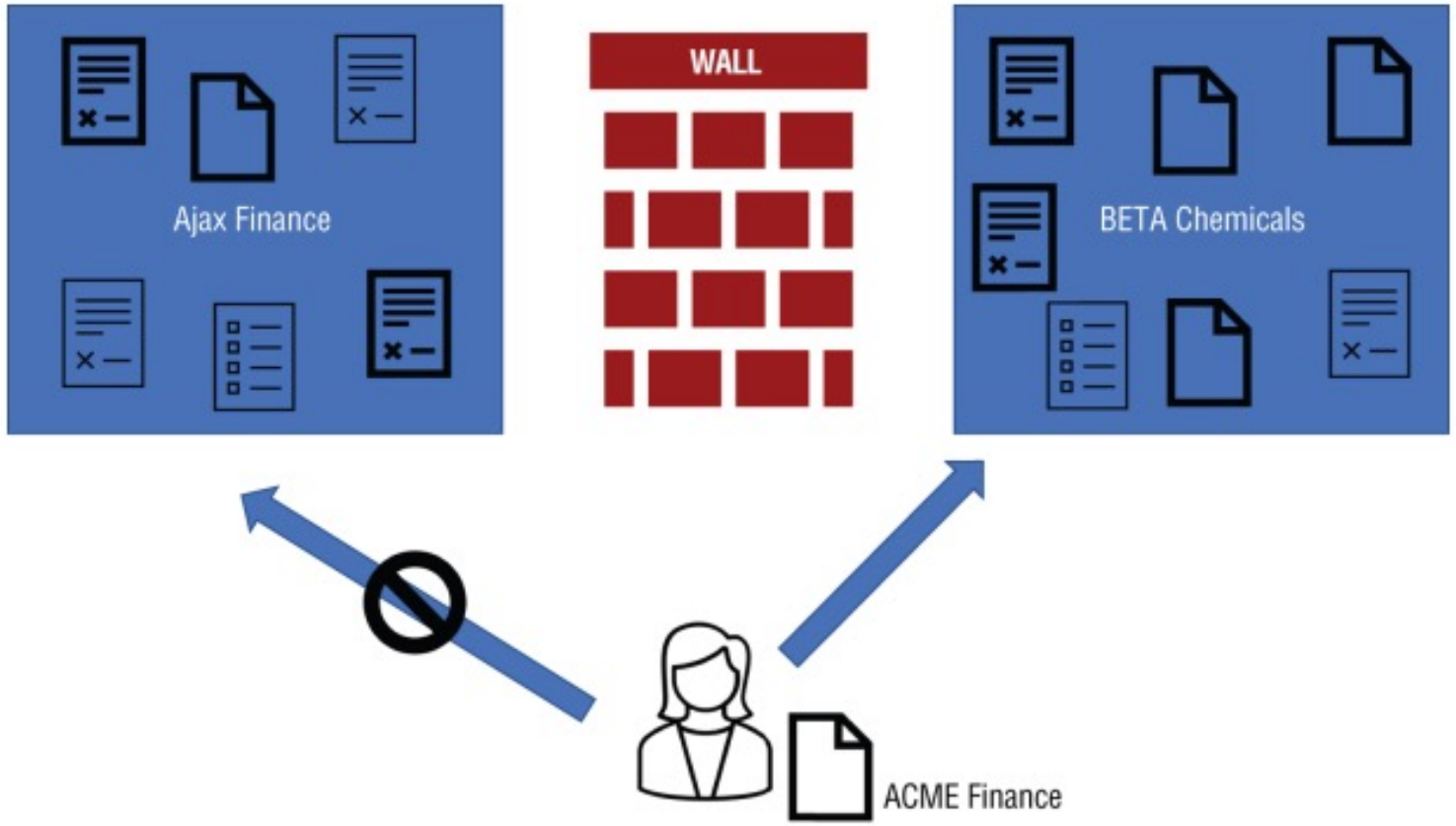
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Bı**
- Inc
  in
- All
  da
- All
  ca



ed objects,

a company

t of what is

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

- **Take-Grant Model**
- **Simple Integrity Property** - **No read down**, this rule prevents compromising the integrity of more secure information from a less secure source. In other words, higher integrity processes could produce untrustworthy results if they read and use data from lower integrity sources.

- **Star Integrity Property** (* integrity property) - **No write up**, this rule prevents the corruption of more secure information by a less privileged subject.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understand the Fundamental Concepts of Security Models

Primer on Common Model Components

Take-Grant Model (four rules)

- **Take**: Allows a subject to obtain (or take) the rights of another object
- **Grant**: Allows a subject to give (or grant) rights to an object
- **Create**: Allows a subject to generate (or create) a new object
- **Remove**: Allows a subject to revoke (or remove) rights it has on an object

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select Controls Based Upon Systems Security Requirements

Selecting Security Controls

- Selecting the security controls appropriate for an information system starts with an analysis of the security requirements.
- Should be defined, repeatable, and consistent.
- Review all of the controls to determine which are appropriate to address the risks you have identified.
- Can demonstrate due care and due diligence in security decision-making.
- Framework is an approach or strategy to take, a standard is a set of quantifiable directions or rules to follow.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select Controls Based Upon Systems Security Requirements

Selecting Security Controls

There are a few things to understand about security frameworks

- They are **not mandatory**.
- They are **not mutually exclusive** of each other.
- They are **not exhaustive** (i.e., they don't cover all security concerns).
- They are **not the same** as a standard or a control list.
- They are **subject to change** or update

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select Controls Based Upon Systems Security Requirements

Selecting Security Controls

- Consider the control and how to implement and adapt it to your specific circumstances (the “Plan” phase)
- Implement the control (the “Do” phase) Assess the effectiveness of the control (the “Check” phase)
- Remediate the gaps and deficiencies (the “Act” phase)

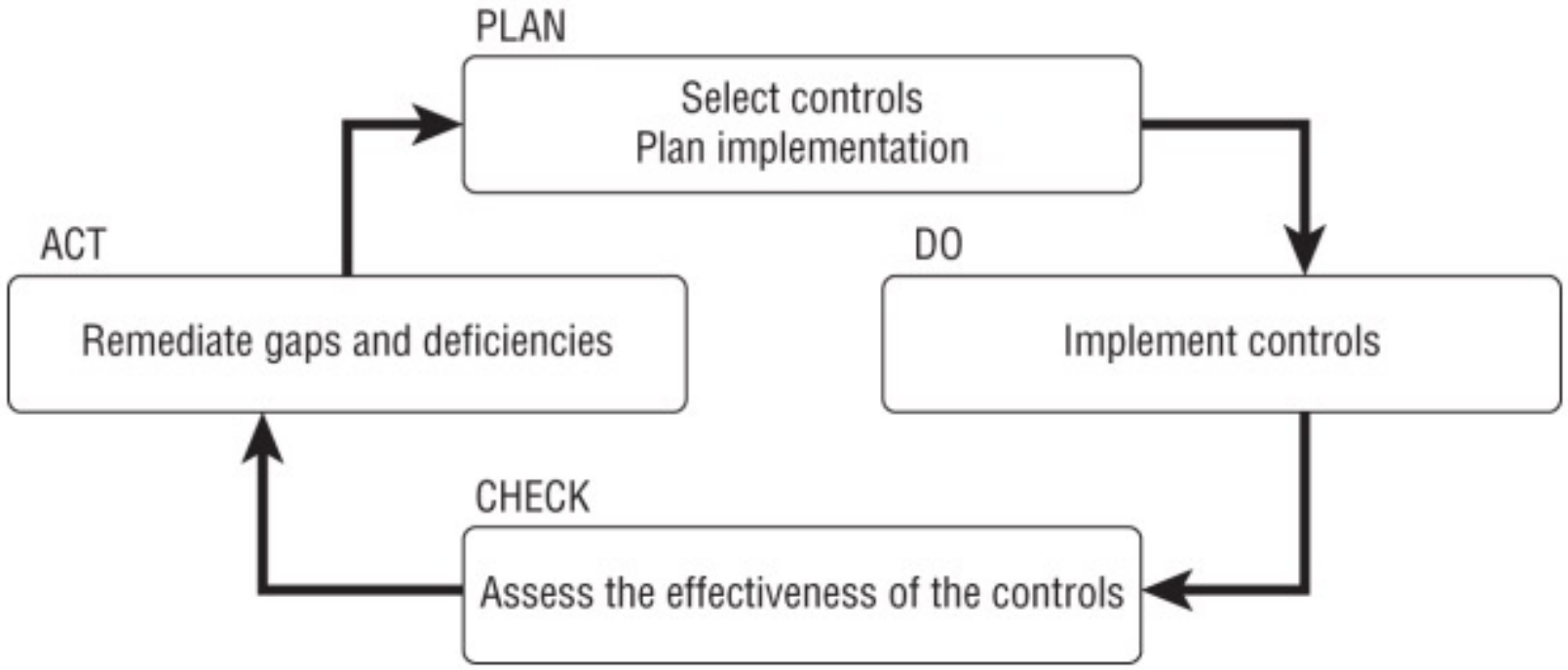
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select Controls Based Upon Systems Security Requirements

Selecting Security Controls

- C dapt
  it e)
- Ir e
- R hase)



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select Controls Based Upon Systems Security Requirements

Selecting Security Controls, when (re-)assess is required

- Security incident or breach
- Significant change in organization structure or major staffing change
- New or retired product or service
- New or significantly changed threat or threat actor
- Significant change to an information system or infrastructure
- Significant change to the type of information being processed
- Significant change to security governance, the risk management framework, or policies
- Widespread social, economic, or political change (e.g., COVID-19)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Foundational Capabilities of Information Systems

- Memory protection Trusted Platform Modules (TPMs)
- Cryptographic modules
- Hardware Security Modules (HSMs)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Memory Protection

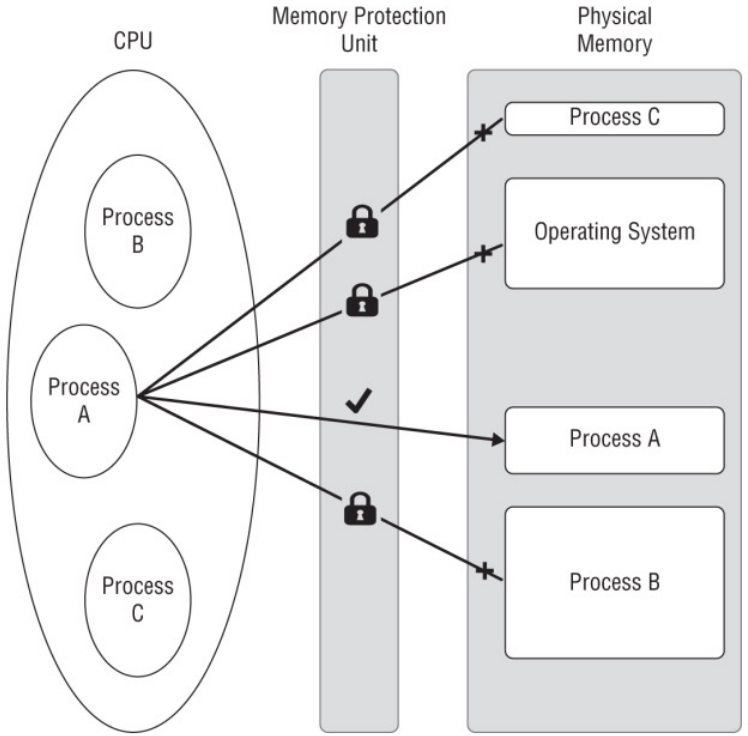
- Foundational security controls on all systems that allows multiple programs to run simultaneously is memory protection.
- Prevents one program from referencing memory not specifically assigned to it.
- If a program attempts to reference a memory address it is not permitted to access, the system blocks the access, suspends the program, and transfers control to the operating system.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Memory Protection



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Memory Protection

- Hardware feature that is required to support memory protection is dual-mode operation.
- Processor can operate in one of (at least) two modes: privileged (or kernel) mode and unprivileged (or user) mode
- The OS runs in privileged mode, which grants it permission to set up and control the memory protection subsystem. Privileged mode also permits the operating system to execute special privileged instructions that control the processor environment
- Address space layout randomization (ASLR), seeks to mitigate the risks of predictable memory address location. (The location in memory for a known instruction becomes a risk when there is a threat of exploiting that location for an attack.)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Memory Protection (Potential Weaknesses)

- Proper memory protection relies upon both the correct operation of the hardware and the correct design of the operating system to prevent programs from accessing memory they have not been given permission to access.
- A defect in either can compromise the security provided by memory protection. Note that this protection prevents the direct disclosure of memory contents that are blocked from an unauthorized program, but does not necessarily prevent side-channel exploits from revealing information about memory that is protected from access.
- Attacks that leverage ineffective isolation and memory protection can have catastrophic effects. Spectre and Meltdown exploits in 2018 revealed, flaws in the design of Intel and some other CPU chips permitted clever programming techniques to deduce the contents of memory locations that those programs were not permitted to access directly.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Secure Cryptoprocessor

- The challenge with standard microprocessors is that code running with the highest privilege can access any device and any memory location.
- The security of the system depends entirely on the security of all the software operating at that privilege level. If that software is defective or can be compromised, then the fundamental security of everything done on that processor becomes suspect.
- To address this problem, hardware modules called secure cryptoprocessors have been developed that are resistant to hardware tampering and that have a limited interface (i.e., attack surface), making it easier to verify the integrity and secure operation of the (limited) code running on the cryptoprocessor.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Secure Cryptoprocessor (Services)

- Hardware-based true random number generators (TRNGs)
- Secure generation of keys using the embedded TRNG
- Secure storage of keys that are not externally accessible
- Encryption and digital signing using internally secured keys
- High-speed encryption, offloading the main processor from the computational burden of cryptographic operations

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Secure Cryptoprocessor (Features)

- Tamper detection with automatic destruction of storage in the event of tampering.
- Chip design features such as shield layers to prevent eavesdropping on internal signals using ion probes or other microscopic devices.
- Hardware-based cryptographic accelerator (i.e., specialized instructions or logic to increase the performance of standard cryptographic algorithms such as AES, SHA, RSA, ECC, DSA, and ECDSA.
- Trusted boot process that validates the initial boot firmware and operating system load.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Secure Cryptoprocessor (Types)

- Proprietary, such as Apple's “Secure Enclave” found in iPhones
- Open standard, such as the TPM as specified by the ISO/IEC 11889 standard and used in some laptops and servers
- Standalone (e.g., separate standalone device with external communications ports)
- Smartcards

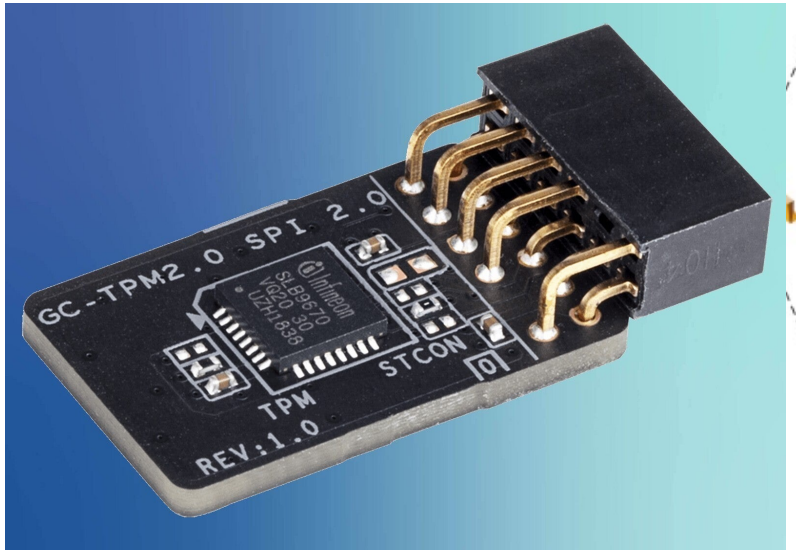
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

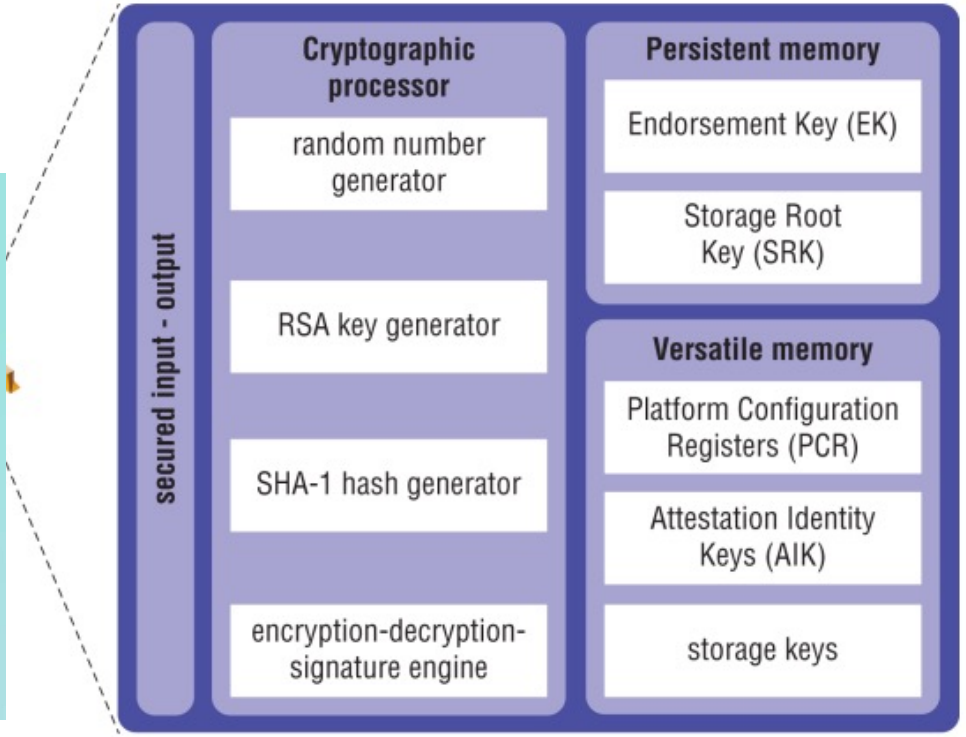
DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Trusted Platform Module

Provides secure storage and cryptographic services as specified by ISO/IEC 11889





This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Trusted Platform Module

- **Attestation**: Creates a cryptographic hash of the system's known good hardware and software state, allowing third-party verification of the system's integrity
- **Binding**: Encrypts data using a cryptographic key that is uniquely associated with (or bound to) the system
- **Sealing**: Ensures that ciphertext can be decrypted only if the system is attested to be in a known good state

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Trusted Platform Module

- Generate private/public key pairs such that the private key never leaves the TPM in plaintext. Increasing the security related to the private key.
- Digitally sign data using a private key that is stored on the TPM and that never leaves the confines of the TPM. Significantly decreasing the possibility that the key can become known by an attacker and used to forge identities and launch man-in-the-middle (MITM) attacks.

- Encrypt data such that it can only be decrypted using the same TPM.
- Verify the state of the machine the TPM is installed on to detect certain forms of tampering (i.e., with the BIOS) and ensure platform integrity.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Trusted Platform Module (Potential weaknesses)

- The endorsement key (EK) is a fundamental component of a TPM's security.
- This key is generated by the TPM manufacturer and burned into the TPM hardware during the manufacturing process.
- User/system owner depends upon the security of the TPM manufacturer to ensure that the PEK remains confidential.
- Flaws in software used in the TPM can expose or make easy to deduce the private keys

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Cryptographic Module

Advantages of using a cryptographic module as opposed to a cryptographic software library include.

- Separate device that is dedicated to that purpose.
- Isolating security-sensitive functionality with limited interfaces and attack surfaces, it is easier to provide assurances about the secure operation of the device.
- increased availability of noncryptographic dedicated resources.
- Most secure cryptographic modules contain physical security protections including tamper resistance and tamper detection.
- Some cryptographic modules can enforce separation of duties so that certain sensitive operations, such as manipulating key storage, can be done only with the cooperation of two different individuals who authenticate to the cryptographic module separately.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Understanding Security Capabilities of Information Systems

Hardware Security Module

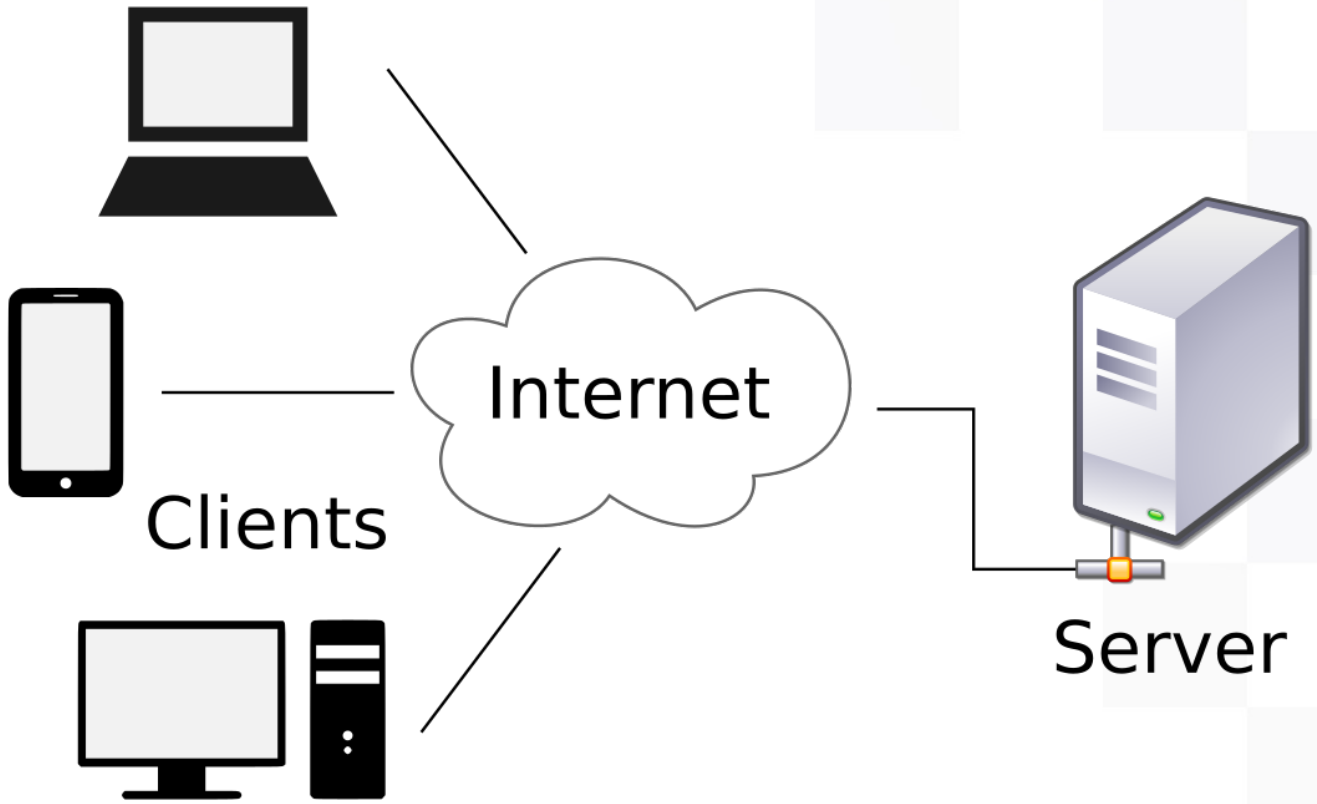
- Stand alone as an appliance to provide cryptographic services over an externally accessible API (typically over a network or USB connection).
- HSMs are frequently found in certificate authorities (CAs) that use them to protect their root private keys, and payment processors.
- HSMs are also used in many national security applications or other environments.
- HSMs are used by enterprise network backbones as part of encryption management of archives, east-west data movement, and even VPN traffic.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Client-Based Systems



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Client-Based Systems

- Client-related vulnerabilities can be grouped into two broad categories, client application itself, and those related to the system on which the client runs.
- Client-based vulnerabilities may fall into the following categories
- Vulnerabilities related to the insecure operation of the client
  - Storing temporary data on the client system in a manner that is insecure (i.e., accessible to unauthorized users through, for example, direct access to the client device's filesystem)
  - Running insecure (e.g., out-of-date or unpatched) software versions

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Client-Based Systems

Vulnerabilities related to communications with the server client software that connects to remote servers but does not take appropriate steps to do the following

- Validate the identity of the server
- Validate or sanitize the data received from the server
- Prevent eavesdropping of data exchanged with the server
- Detect tampering with data exchanged with the server
- Validate commands or code received from the server before executing or taking action based on information received from the server.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Client-Based Systems

To address these vulnerabilities, consider the following

- Evaluate your operating systems and applications for unpatched software or insecure configurations.
- Using a recognized secure protocol (e.g., transport layer security (TLS)) to validate the identity of the server and to prevent eavesdropping of, and tampering with, data communicated with the server.
- Using appropriate coding techniques to ensure that the data or commands received from the server are valid and consistent.
- Using digital signing to verify executable code received from the server prior to execution.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Server-Based Systems

- The server needs to validate the identity of the client and/or the identity of the user of the client. This can be done using a combination of Identity and Access Management (IAM) techniques along with a secure communications protocol such as TLS, using client-side certificates.
- The server also must validate all inputs and not assume that simply because the commands and data coming from the client are originating from (and have been validated by) the corresponding client-side software, they are valid and have been sanitized.
- The client must be considered untrusted, and it must be assumed that the client-end can insert or modify commands or data before being encrypted and transmitted over the secure (e.g., TLS) link.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Server-Based Systems

- A vulnerability management program is needed to ensure that updates and patches are applied in a timely fashion. This holds true regardless of whether the server-side software is developed in-house or is based in part or completely on software obtained from a third party (such as commercial off-the-shelf software, or COTS).
- Best practices include the server using filesystem ownership and permissions to avoid data leakage, logging and monitoring appropriate information (such as successful and failed login attempts, privileged access, etc.), and capturing forensic information to permit analysis of a possible or actual security incident.
- Finally, threats to the server itself need to be addressed. This may include physical and environmental threats, threats to the communications infrastructure, and server hardening as per industry recommendations.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Server-Based Systems (Server Harding Guidelines)

- Installing updates and patches
- Removing or locking unnecessary default accounts
- Changing default account passwords
- Enabling only needed services, protocols, daemons, etc. (conversely, disabling any not needed)
- Enabling logging and auditing Implementing only one primary function per server
- Changing default system, filesystem, service, and network configurations as needed to improve security (including full-disk encryption if appropriate)
- Removing (or disabling) unneeded drivers, executables, filesystems, libraries, scripts, services, etc.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DAD JOKE

Laughter for Levity

How about a dumb dad joke?

What did Baby Corn say to Mama Corn?

HAHAHAHA

Moving on…

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Securing database systems is a special case of the more general server-based system security discussed in the previous section. If the database is accessible over a network, then all the security controls discussed there apply

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Consult the CIS's hardening guidelines for the database system being used. These guidelines include several of the recommendations below, and many others.
- Only install or enable those components of the database system that are needed for your application.
- Place data stores and log files on nonsystem partitions.
- Set appropriate filesystem permissions on database directories, data stores, logs, and certificate files.
- Run database services using a dedicated unprivileged account on a dedicated server.
- Disable command history

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Do not use environment variables, command line, or database configuration files to pass authentication credentials.
- Do not reuse database account names across different applications. Disable “anonymous” accounts (if supported by the database).
- Mandate that all connections use TLS if access or replication traffic travels over untrusted networks.
- Use unique certificates for each database instance. Use restricted views.
- Ensure that all DBMS vendor-provided sample or test databases are removed or not accessible from user endpoints and clients.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Change all default passwords, ensure all accounts have secure passwords, and consider enabling multifactor or certificate-based authentication (where supported).
- Ensure user account permissions have been assigned using the principle of least privilege and in alignment with enterprise-wide access control policies and procedures. Database privileges can be complex and interact in unexpected ways—avoid default roles and define those you need with only the permissions needed for each.
- Disable or remove unneeded accounts, especially those with administrative permissions.
- Manage all accounts according to best practices (see Chapter 5).
- Enable logging of sensitive operations and route logs to your log monitoring and alerting system.
- Use bind variables where possible to minimize injection attack surfaces.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Assign unique admin accounts for each administrator (i.e., do not share admin accounts between more than one admin)
- Enable logging at a sufficiently detailed level to provide the forensic information needed to identify the cause of events related to security incidents (but ensure logging does not include passwords)
- Protect the logs from tampering by database admins, either through permissions on the database system itself or by transmitting the log data in real time to a separate secure logging system.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Consult vendor database documentation for database-specific security controls.

- For databases that are only accessed through application software (e.g., the typical n-tier web server application), run the database on private networks only accessible to the business logic servers that need access.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems (Database encryption)

- Full-disk encryption (FDE) at the lowest level protects all the data on the storage media, protecting against the physical theft or loss of the drive itself. It provides no protection from threat actors who have logical access to the system.
- Filesystem-level encryption allows the encryption to occur at the filesystem level.
- Transparent data encryption (TDE) protects the data from those who have direct access to the filesystem (i.e., the “root” user), but do not have permission to access the database system and the specific database item.
- Cell-level encryption (CLE) encrypts database information at the cell or column level. With this approach, data remains encrypted when read from the database and is decrypted only when requested.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems (Database encryption)

- Full-disk encryption (FDE) at the l ta on the storage media, protecting against the physical th
- Filesystem-level encryption allows the encryption to occur at the filesystem level.
- Transparent data encryption (TDE) protects the data from those who have direct access to the filesystem (i.e., the "root" user), but ess the database system and the specific databas
- Cell-level encryption (CL E) encrypts database information at the cell or column level. With this approach, data s decrypted only when re

Provides no protection from threat actors who have logical access to the system.

Does not protect against malicious database administrators or attacks, such as SQL injection, not intended to be used alone

Key management and handling the decryption/encryption requests can add considerable complexity to the application and depending on the types of queries (and whether they include CLE-protected data), the performance can be affected, sometimes drastically.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Database Systems

- Application-level encryption is a high-level approach that provides protection even if access to the database system is compromised.
- Business-logic or application layer is responsible for encrypting the data to be protected before it is passed to the database and for decrypting it once it has been retrieved.
- This approach is the most complex, but provides greater security (if properly implemented and managed)
- Handle the encryption/decryption as close to the point of use as possible
- The decision as to which combination of database encryption approaches to use will be influenced by considerations such as:
  - Performance, especially if searches reference data encrypted using CLE.
  - Backups, which will be protected using TDE or CLE, but not necessarily when using FDE (unless the backup is on another FDEprotected drive).
  - Compression as encrypted data does not compress, so the use of encryption may significantly increase the size of backups.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

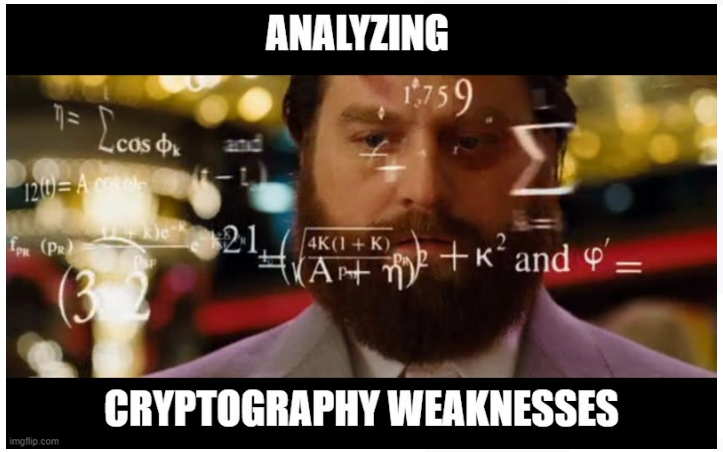
DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems

"All cryptography can eventually be broken the only question is how much effort is required." – Bruce Schneier

- A number of avenues that can be followed to compromise a cryptographic system.
  - Algorithm and protocol weaknesses
  - Implementation weakness
  - Key management vulnerabilities



*There are countries that strictly regulate the use of cryptography, and countries that, while permitting the unrestricted use of cryptography, regulate the export of cryptographic technology

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Algorithm and protocol weaknesses)

- **Cryptology is hard**, and even the experts get it wrong.
- The cryptographic attack surface includes not only the algorithm, but the **people**, **processes**, and **technology** that implement the cryptographic protections, all of which are potentially vulnerable to attack.
- Cryptanalysis becomes more effective over time, owing to advances in computing, mathematical breakthroughs, and other improvements in cryptanalytic methods.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Algorithm and protocol weaknesses)

- Crypt
- The c
peop
prote
- Crypt
comp
cryptana

but the
hic
es in
s in

Time erodes the security of cryptographic protections.

Plan for the Lifecycle

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Implementation Weaknesses)

Use industry-standard and tested algorithms, implemented in published libraries. **Don't invent or implement algorithms yourself.**

Side-channel attack is the analysis of artifacts related to the implementation of the algorithm, such as the time the algorithm takes to execute, the electrical power consumed by the device running the cryptographic implementation, or the electromagnetic radiation released by the device.

The best defense is to **use standard cryptographic libraries** that have been tested over time for side-channel information leakage.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Implementation Weaknesses)

There are also a number of steps one can take to minimize the possibility of leaking information via side channels.

- Compare secret strings (e.g., keys, plaintext, unhashed passwords) using constant-time comparison routines.
- Avoid branching or loop counts that depend upon secret data.
- Avoid indexing lookup tables or arrays using secret data Use strong (i.e., “cryptographic grade”) random number generators.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Implementation Weaknesses)

There are ‹ ‘king
informati(

- Compa ant-time
  compar
- Avoid b
- Avoid ir ɔgraphic
  grade”)

Read the Case Studies in the Book

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Key Management Vulnerabilities)

A number of vulnerabilities that can be introduced through the incorrect use, storage, and management of cryptographic keys.

Keys should be generated in a manner appropriate for the cryptographic algorithm being used.

The proper method to generate a symmetric key is different from a public/private key pair.
NIST SP 800-133, "Recommendation for Cryptographic Key Generation," provides specific guidance.

Keys should not be reused and should be rotated (replaced) periodically to ensure that the amount of data encrypted using a single key is limited.

Symmetric and private keys depend upon confidentiality to be effective. This means great care must be taken with how the keys are stored to reduce the possibility of their becoming known to unauthorized entities.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cryptographic Systems (Key Management Vulnerabilities)

Secure key management is paramount to maintaining cryptography benefits and security

- Key management software
- Key management services provided by cloud service providers
- Dedicated hardware devices that keep the keys stored internally in a tamper-resistant secure device
- Keys should have a defined lifetime
- Account for insider threat (Dual control or SOD)
- Must consider availability (CIA Triad)
- Key operations should be logged
- Automate Key management functions when practical

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Industrial Control Systems

Industrial control systems (ICSs) are used to automate industrial processes and cover a range of control systems and related sensors.

Security in this context concentrates mostly on the integrity and availability aspects of the CIA Triad: integrity of the data (e.g., sensor inputs and control setpoints) used by the control system to make control decisions, and availability of the sensor data and the control system itself.

In addition to integrity, safety is a huge consideration for modern industrial control systems, such as those that steer aircrafts, treat our drinking water, and power biomedical systems.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Industrial Control Systems

- Historically, ICSs communicated using proprietary methods and were not connected to local area networks (LANs) or the internet, so security was not a design consideration.

- Today, many industrial control systems have been attached to internet protocol (IP) gateways without much consideration as to the threats such access enables.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Industrial Control Systems

There are a number of organizations that provide guidance or regulations related to ICS security:

- ISA/IEC-62443 is a series of standards, technical reports, and related information that define procedures for implementing electronically secure Industrial Automation and Control Systems (IACS).
- The North American Electric Reliability Corporation (NERC) provides a series of guides referred to as the Critical Infrastructure Protection (CIP) standards. NERC CIP standards are mandatory in the United States and Canada for entities involved in power generation/distribution.
- The European Reference Network for Critical Infrastructure Protection (ERNCIP) is an EU project with similar aims to those of NERC CIP.
- NIST and the UK National Centre for the Protection of National Infrastructure (CPNI). See, for example, NIST publication SP800-82.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Industrial Control Systems

These are the challenges specific to industrial control

- The difficulty of patching device firmware to address vulnerabilities in the software discovered after placing the device into production in the field
- Failure to change factory-default settings, especially those related to access controls and passwords
- The long production lifetime of industrial systems as compared to IT systems
- The reliance on air-gapped networks as a compensating control without proper supervision of network connections

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Industrial Control Systems

With ICSs, patching can be difficult or impossible

- With industrial systems operating nonstop, it may not be feasible to remove an ICS device from operation to update its firmware.
- Similarly, with continuous production being important, the risk of an update breaking something (such as patching the underlying operating system and interfering with the ICS app running on that OS) can be too great (and greater than the perceived risk of running obsolete software). Consider an air traffic control system or an oil and gas pipeline, for example. These systems are hugely important and rely on nearly 100 percent uptime. Often, organizations must weigh the risk of operating an unpatched system with the risk of a patch disrupting service.
- Finally, the location of the ICS device in the field may make the simple matter of reaching the device physically to connect a laptop to install the firmware update a significant undertaking.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Industrial Control Systems

Physical security and compensating controls are key

- Computers used to maintain and manage industrial systems must never be used for any other purpose
- It is essential to limit and screen permitted traffic accessing the ICS network through the use of carefully configured firewalls and network proxies
- For ICSs that must be remotely accessible, compensating controls such as installing a web proxy or VPN should be considered to add an additional layer of security on top of whatever access controls are implemented on the ICS itself.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

- According to NIST, “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” In short, cloud-based systems are remotely located, separately managed systems that are accessible by the internet.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

NIST SP 800-145 and ISO/IEC 17788 define a number of characteristics that describe cloud computing

- Broad network access: Resources (physical and virtual) are accessible and managed over the network.
- Measured service: Users pay only for the services they use.
- On-demand self-service: Users can provision and manage services using automated tools without requiring human interaction.
- Rapid elasticity and scalability: Services can be rapidly and automatically scaled up or down to meet demand.
- Resource pooling and multitenancy: Physical or virtual resources are aggregated to serve multiple users while keeping their data isolated and inaccessible to other tenants.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

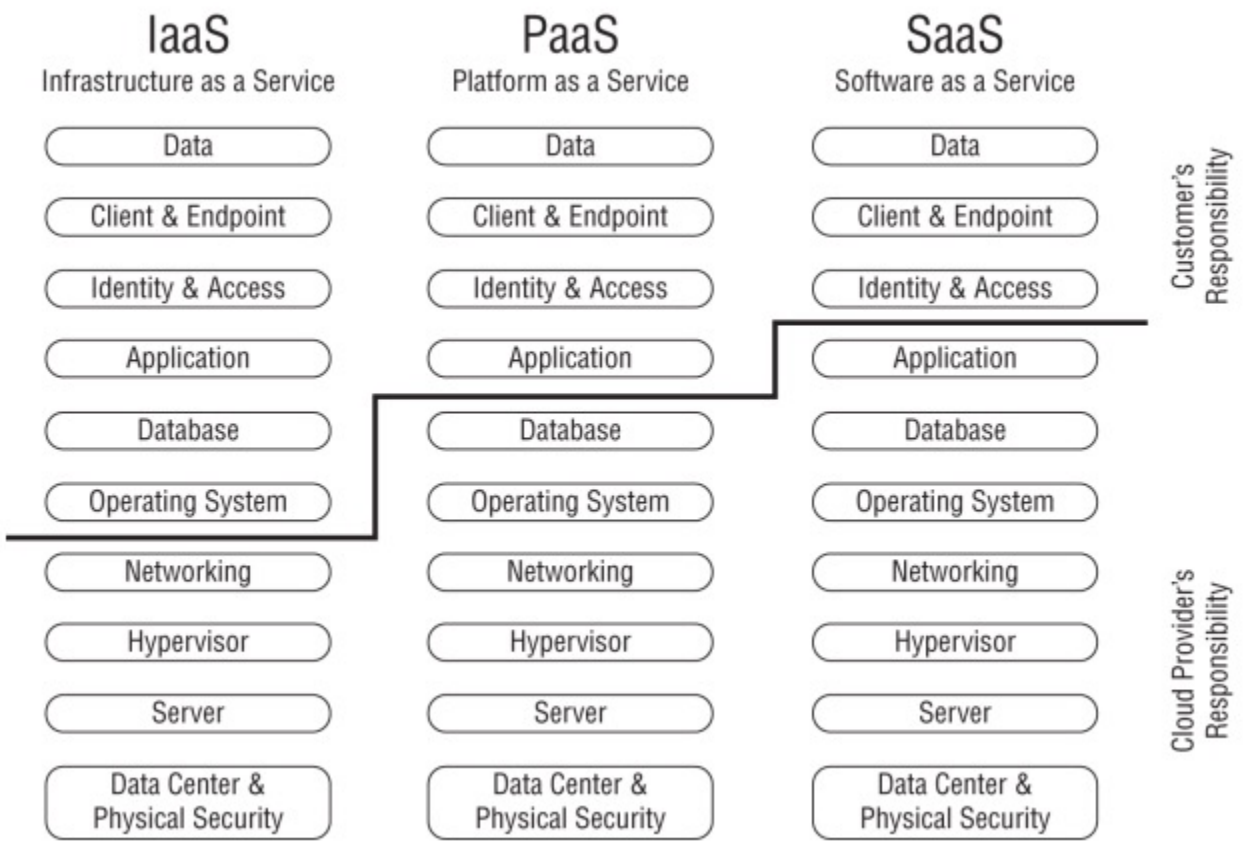
CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

CLOUD SERVICE MODEL	SERVICE PROVIDED	SERVICE PROVIDER RESPONSIBILITIES	CUSTOMER RESPONSIBILITIES
Software as a service (SaaS)	Software application accessible to the customer over the internet (via a browser or API)	Provide and manage all infrastructure from server and network hardware to applications software	Provide the client device and manage user-specific configuration settings
Platform as a service (PaaS)	Web-based framework for developers to create customized applications	Provide and manage all infrastructure from server and network hardware to the libraries and runtime services necessary to run applications	Provide the application and manage the hosting environment
Infrastructure as a service (IaaS)	Infrastructure, including servers, network, storage, and operating systems, delivered through virtualization technology	Provide network and server infrastructure to support VMs and other virtualized resources	Provide and manage all components that run on the VM as well as limited aspects of network services



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

In particular, the cloud service provider is exclusively responsible for the following.

- Physical security Environmental security
- Hardware (i.e., the servers and storage devices)
- Networking (i.e., cables, switches, routers, firewalls, and internet connectivity)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

The cloud service provider and the customer share responsibility for the following

- **Vulnerability and patch management**
- **Configuration management**
- **Training**

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Cloud Based Systems

Cloud service can be deployed in a number of ways (known as deployment models)

- **Public cloud -** Available to any customer
- **Private cloud -** Used exclusively by a single customer (may be in-house or run by a third party, on-premise or off)
- **Community cloud -** Used exclusively by a small group of customers with similar interests or requirements (may be managed by one or more of the customers, or a third party, on-premise or off)
- **Hybrid cloud -** A combination of two or more of the above deployment models

*Since many government agencies share similar interests and requirements, the notion of a government cloud (or "GovCoud") has become an important community cloud concept.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Distributed Systems

A distributed system, by definition, involves multiple subsystems, possibly distributed geographically, and interconnected in some manner, the attack surface is much larger than that of a single system.

It is important to model threats to the overall system and identify the relative risks that need to be addressed.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Distributed Systems (Things to consider)

- The need for encryption and authentication on the connections between the subsystems to ensure attackers cannot intercept, eavesdrop, or spoof communications between subsystems
- The need to protect against DoS attacks against the communications links or the subsystems themselves
- The risks from a lack of homogeneity across subsystems (e.g., different versions and patch levels of operating systems, middleware, and application software; difficulty of maintaining consistent configurations across disparate and distributed systems) and mechanisms to mitigate those risks

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Distributed Systems (Things to consider)

- The need to maintain consistency should communications be disrupted (delayed or interrupted) between groups of (normally) connected subsystems (sometimes referred to as the “split-brain” problem)
- The challenge of ensuring comparable security controls in the case of geographically distributed components (e.g., physical, environmental, and personnel)
- The requirements of privacy and data sovereignty regulations that may limit the transfer of personal data across international borders

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things

The term Internet of Things (IoT) describes a network of physical objects that are embedded with technologies (e.g., sensors and software) that enable them to connect to and exchange data with other devices over the internet.

Examples include household appliances, medical equipment, smart home devices, and so on. Estimates are that the number of such devices in 2020 was somewhere between **20 and 50 billion**, and the rapid expansion of 5G networks is expected to continue to drive IoT growth.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things

The term Internet of Things (IoT) describes a network of physical objects that are embedded with technologies (e.g., sensors and software) that enable them to connect to and exchange data with other devices over the internet.

Examples include household appliances, medical equipment, smart home devices, and so on. Estimates are that the number of such devices in 2020 was somewhere between **20 and 50 billion**, and the rapid expansion of 5G networks is expected to continue to drive IoT growth.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things

The importance of IoT security can be demonstrated through the infamous Mirai distributed denial of service (DDoS) attack.

The Mirai attack (Figure 3.10) involved a worm that searched for vulnerable IoT devices (typically consumer routers and IP-enabled closed circuit television (CCTV) cameras), infected them with a copy of the malware, and then waited for instructions from a command and control (C&C) server as to which target to attack with a DDoS attack.

In late 2016, this botnet took the Krebs on Security blog offline and later attacked the Dyn DNS service, which in turn seriously impacted many of their customers including GitHub, Twitter, Reddit, Netflix, and Airbnb.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things

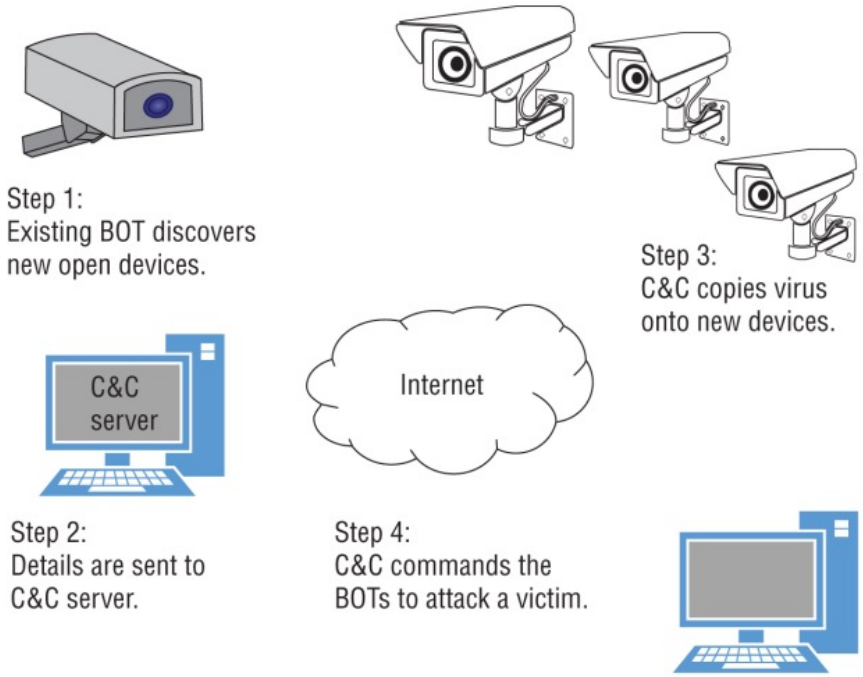


FIGURE 3.10 Components of the Mirai DDoS BotNet attack

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things from a Manufactures Perspective

- During development, you will want to conduct threat modeling to determine likely vulnerabilities and to ensure that appropriate mitigations are deployed
- Review their product's security architecture to determine if the general guidelines outlined earlier in this section have been observed in the design of the firmware.
- Development team will need to pay particular attention to secure software development guidelines such as those from the open web application security project (OWASP) and SANS.
- Quality assurance (QA) team will need to perform active white- and black-box penetration testing.
- Basic security hygiene such as changing default credentials and updating the firmware to patch known vulnerabilities.
- Make implementing the previous two security controls as easy as possible

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things from a Manufacturer's Perspective

- Device should refuse to connect to the internet until the user has changed the default admin credentials. It means that your device should update itself automatically (with the consent of the user), and if auto-update has not been enabled, possibly refuse to operate (after sufficient notice) should a patch be available for a high-severity vulnerability being actively exploited in the wild.
- In some cases, IoT devices may refuse to operate without first connecting to the internet for initialization. In these cases, it's best that you keep potentially insecure IoT devices isolated from critical systems or sensitive areas on your network.
- While ease of use is a key factor in the commercial success of IoT devices, one has to draw the line where ease of use is directly connected to ease of compromise.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things from a User's Perspective

- To start, you can protect yourself (and others that might be a target of your compromised devices) through the same two basic security controls previously mentioned
  - Change default credentials as soon as possible, and before you connect the device to the internet.
  - Keep your device updated with the current firmware release, either by enabling auto-update (if supported by your device) or by periodically checking with the manufacturer's website for firmware updates.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Internet of Things from a User's Perspective

- In addition, you can employ security in depth through additional controls:
  - Do not place IoT devices on the open internet, but rather behind a firewall so that they are not directly accessible externally.
  - Segment your network so that your IoT devices do not have access to other sensitive devices or servers on your internal networks. If you have to be able to access your IoT device externally, then at the very least put the device behind a router that does reverse NAT mapping.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Microservices

- Microservice architecture is a modular software development style that involves developing a single application as a collection of loosely coupled smaller applications or services (microservices), each running its own processes.
- Microservices are built to be independently deployable and work together through lightweight communications protocols.
- Microservice architectures are highly distributed and dynamic and present unique security concerns that must be considered from the first stages of design and throughout the entire development lifecycle. Two key principles to consider when securing microservices are: isolation and defense in depth.
- Monolithic architecture, which involves developing an application as a single, indivisible unit, typically with a large codebase that lacks modularity.

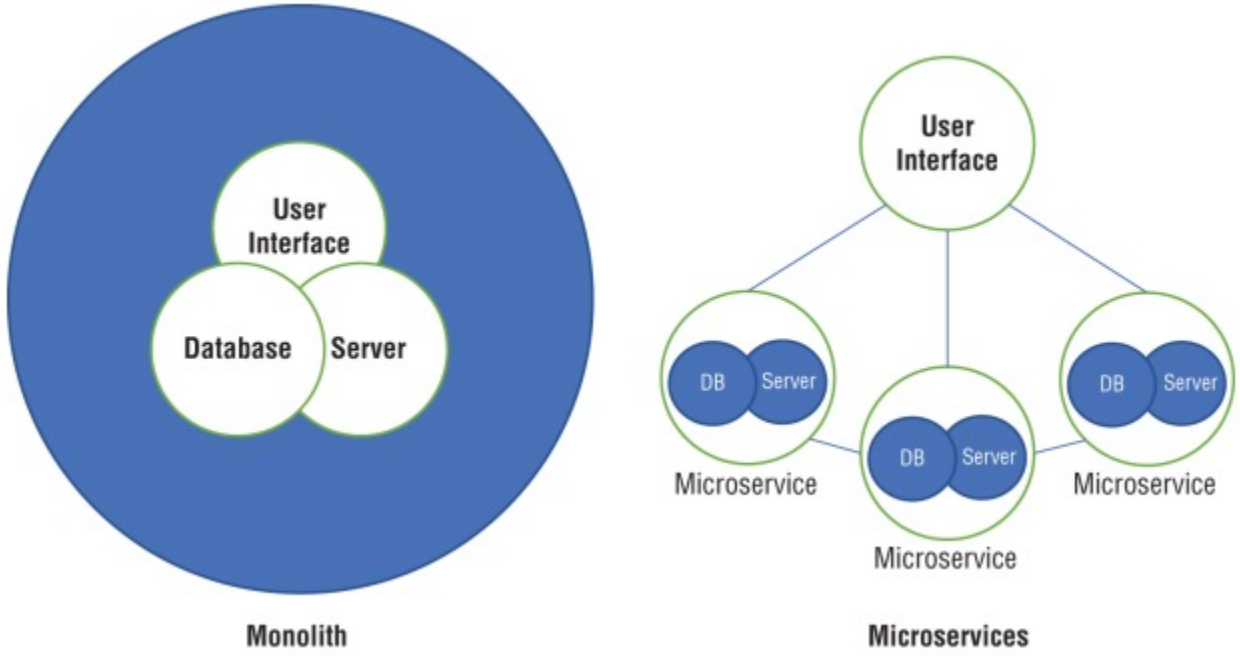
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Microservices



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Microservices

- Isolation is a core principle of microservices, and each microservice must be able to be deployed, modified, maintained, and destroyed without impacting the other microservices around it.
- The principle of defense in depth, while important in any architecture, is particularly critical when dealing with microservices
- Defense in depth is a security strategy that calls for multiple layers of security controls to be implemented throughout an application or system.
- It is essential in a microservice architecture to independently monitor and protect each microservice and the communications between each microservice in the overall environment.
- APIs are the most vulnerable part of microservice architecture

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Containerization

- A container is unit of software that packages up an application and its dependencies so that the application can be decoupled from its environment and developed, deployed, and run consistently across multiple environments.
- A container uses the operating system's kernel and only the resources required to operate the given application.
- Containers were made popular with the development of the open-source Kubernetes platform. Kubernetes and other container platforms are particularly useful in hybrid cloud environments, as they allow developers and users to seamlessly move applications from one cloud to another, or even between cloud and on-prem environments.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Containerization Security Concerns

- Containerization comes with its own set of security challenges, as container technology is inherently flexible and open. Because containers allow you to rapidly scale up and down resources, asset management and configuration management are perhaps even bigger security concerns than traditional systems.
- Container security risks generally fall into two major categories:
  - Compromise of a container image or the entire container repository
  - Misuse of a container to attack other containers or the host OS
- Your base container image is the most important, because it is used as a starting point for derivative images.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Containerization Security Concerns

- In addition to managing secure image baselines, you must also ensure proper access controls to all of your container images. Use role-based access controls, where possible, to manage access to your container images.
- Securing the host OS that runs your containers is a foundational part of securing your containers.
- Host operating systems should run only the minimally required services necessary to operate the containers and exclude applications like web servers, databases, and others that increase the attack surface
- Proper configuration is also important, and host OSs must be included in your configuration management plans. In addition, communications between containers should be restricted based on the principle of least privilege — only allow containers to communicate with those containers that are absolutely required for operation
- Use orchestration and management tools

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Serverless

- Serverless computing is a cloud computing model that involves the cloud provider managing servers, and dynamically allocating machine resources, as needed.
- Infrastructure management tasks like provisioning and patching are handled by the cloud provider
- Serverless computing comes with some notable security benefits. To start, serverless functions are typically ephemeral (i.e., short lived).
- This short-lived nature creates a moving target that adds a high degree of difficulty for attackers to compromise
- Serverless functions are commonly much smaller codebases than even the smallest containers.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Serverless

- Effective serverless security is built on ensuring code integrity, tight access permissions, and proper monitoring.
- You should maintain least privileged access for serverless functions, as you do other services — serverless functions should be granted only the access and permissions necessary to execute their task.
- Code should be routinely scanned for vulnerabilities and configuration issues.
- Runtime protection should be used to detect suspicious events or errors that may lead to unexpected behavior or compromise.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Embedded Systems

Embedded systems are dedicated information processing components built into larger mechanical or electrical systems, intended to provide a limited set of functions.

- Domestic appliances (e.g., dishwashers, clothes washers and dryers, refrigerators, and televisions)
- Office equipment (e.g., printers, scanners, and fax machines)
- Networking devices (e.g., routers, switches, and firewalls)
- Cars and other automobiles
- ATMs
- Medical devices (e.g., heart monitors, glucose meters, and IV infusion pumps)
- Mass transit vehicles, stations, and systems
- Building automation and control systems
- Traffic control and monitoring systems

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Embedded Systems

- Assessing the vulnerabilities in an embedded system ought to start with an enumeration of the attack surfaces available and then examining each.
- This examination can be done in a number of ways, including code inspection, threat modeling, and white- or black-box penetration testing.
- Generally, these attack surfaces will fall into the following categories:
  - User interface (UI, which are buttons or other methods of user input)
  - Physical attacks
  - Sensor attacks
  - Output attacks
  - Processor attacks

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Embedded Systems

- UI attacks involve manipulating the controls of the device in a manner that causes the device to malfunction
- Physical attacks involve the compromise of the embedded system's packaging, either to directly compromise the device or to gain access to parts of the embedded system in order to expose other attack surfaces that may be vulnerable.
- Sensor attacks involve manipulating, or intercepting data from, the sensors the embedded system uses to detect external conditions that are relevant to its operation.
- Output attacks involve manipulating the actuators controlled by the embedded system to bypass the controls imposed by the system.
- Processor attacks involve compromising the processor directly, through means that can range from connecting directly to the processor or memory chips to carefully removing the tops of integrated circuits and using ion beams to probe the chip to obtain or manipulate information within the processor. Processor attacks are normally preceded by a physical attack to gain access to the processor.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Embedded Systems

- Embedded systems that support firmware updates may be vulnerable to accepting rogue or unauthorized firmware.
- As with IoT devices, a problem is that it is difficult, if not impossible, to upgrade the software in many embedded systems.
- Vulnerabilities that are discovered after the product has shipped may be difficult or impossible to patch.
- The result may be the need for compensating controls to mitigate the risk from the unpatched vulnerability or the need to replace the unit entirely.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

High-Performing Computing Systems

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

High-Performing Computing Systems

- High-performance computing (HPC) refers to the use of one or more supercomputers, generally for the purpose of highly complex computational science and other mathematically involved applications.
- Generally speaking, HPC systems experience many of the same security concerns as traditional systems and other cloud-based systems.
- HPC's are subject to software vulnerabilities, configuration issues, and compromised credentials.
- Any customized hardware and software present an added threat vector that must be secured.
- As a best practice, HPC systems should be moved to their own physical enclave or logical security zone that is separate from traditional systems.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Edge-Computing Systems

- Edge computing is a distributed computing model that brings compute and storage resources closer to the location where it is needed, improving response times and reducing bandwidth.
- The concept of edge computing dates back to the content delivery networks (CDNs) of the 1990s and now extends into the world of cloud computing. CDNs are covered in detail in Chapter 4, “Communication and Network Security.”
- Edge computing allows pseudo-local data processing to minimize data sent over the internet.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Edge-Computing Systems Security Concerns

- Devices located at the edge, rather than centrally managed in a data center or other tightly managed facility, may not always receive the same diligence as their peers.
- You must be sure to apply the same security rigor to edge devices as your centrally managed devices.
- This includes hardening, patching, and providing the right level of physical security for edge computing systems.
- Data must be encrypted when in transit between edge devices and centralized systems (or the cloud), and VPN tunneling may also be advisable for sensitive data when managing remote systems.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems

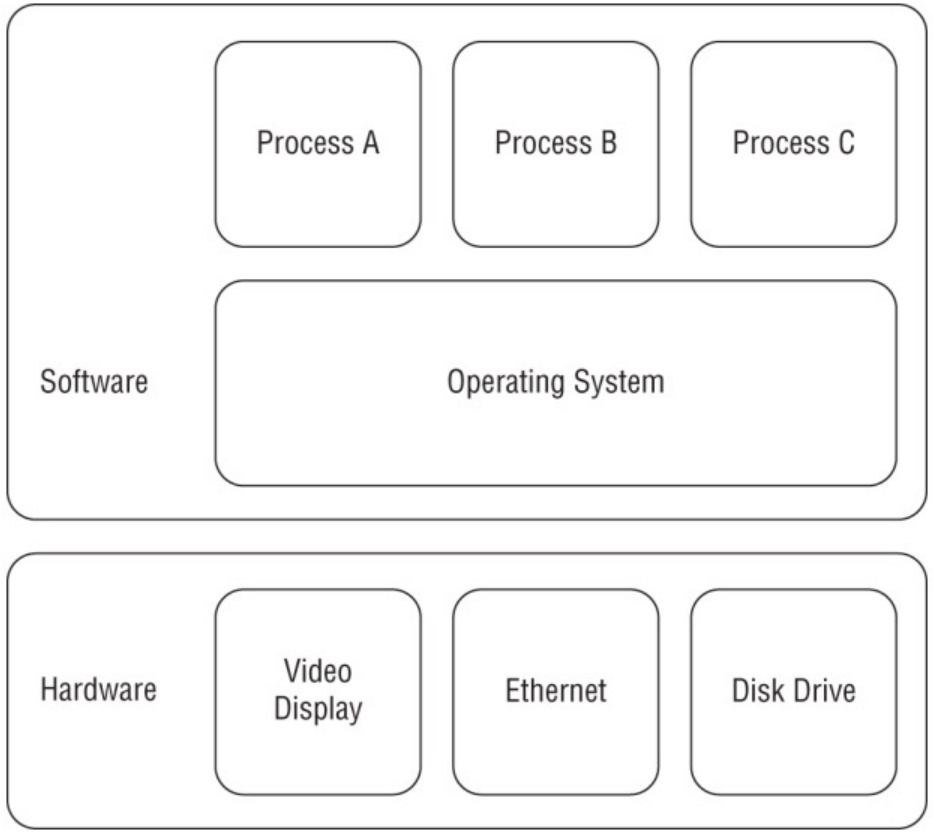
- Operating systems provide programs with a set of services to enable them to operate more efficiently (and to be more easily designed and run) than if the program had to run on the computer directly.
- The operating system provides a level of abstraction that manages the details of files and directories.
- Virtualization is the act of creating virtual (i.e., not real) compute, storage, and network resources, virtualization allows you to create software versions of hardware.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems

- VMs, for instance, are software instances of actual computers. Likewise, software-defined networks (SDNs) are software instances of physical networks.
- Virtualization enables multiple operating systems to run on the same computer, each unaware of and unable (in a properly designed system) to affect the other operating systems.
- Virtualization is the primary technology behind cloud computing.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems

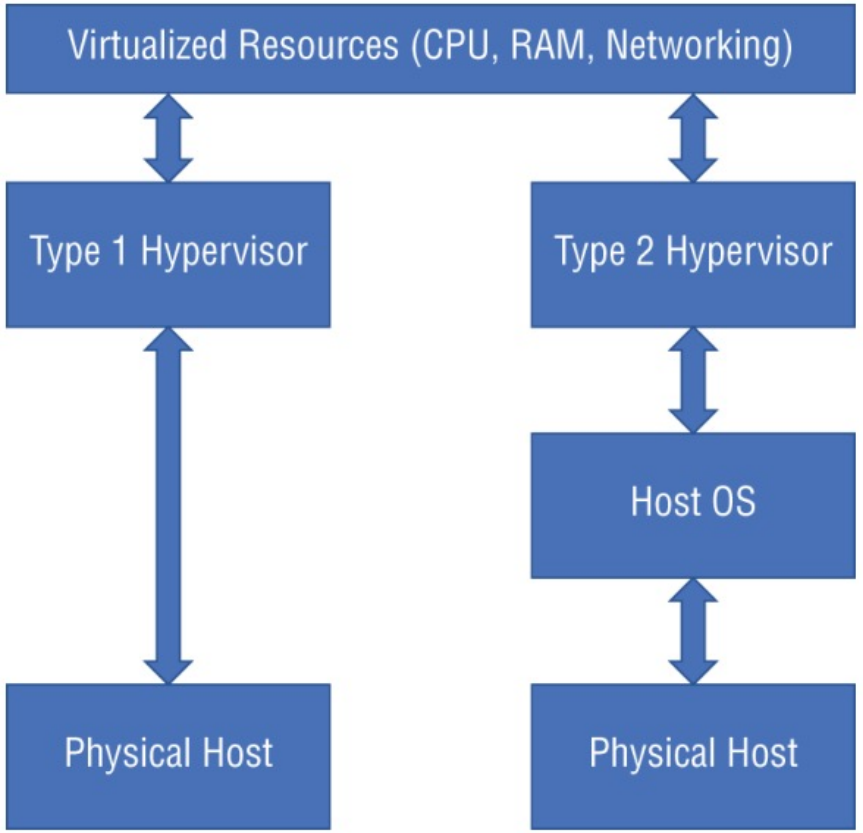
- A hypervisor is a computing layer that allows multiple operating systems to run simultaneously on a single piece of hardware.
- There are two types of hypervisors, commonly referred to as Type 1 and Type 2 hypervisors.
- A Type 1 hypervisor is the sole installation, acting as a bridge between hardware components and VMs.(bare-metal hypervisors)
- Type 2 hypervisor, relying on a host operating system installed on the hardware.
- Virtualized machines running within the host OS are then called guest machines.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems Advantages

- More efficient use of the underlying hardware (just as operating systems permitted a single computer to be shared by multiple programs and users)
- Dynamic scaling of infrastructure in response to demand
- Additional separation and isolation between programs and applications running on different operating systems (as opposed to running on the same OS) — supporting the security principles of defense in depth and layers of security outlined earlier

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Assess and Mitigate the Vulnerabilities of Security Architectures, Designs, and Solution Elements

Virtualized Systems Potential Weaknesses

- As with memory protection, virtualization depends on the correct operation of both the hardware and the hypervisor.
- Software defects in hypervisors can improperly permit software running on one VM to access data on a different VM on the same computer.
- An exploit known as virtual machine escape, for example, occurs when a program is able to break out of its VM and directly interact with the underlying host operating system.
- Type 2 hypervisors generally have a greater attack surface because of the additional vulnerabilities associated with the host operating system and associated software.
- Type 1 hypervisors generally have embedded operating systems that are hardened and tightly controlled by the vendor.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DAD JOKE

Before we get too deep into this.

How about a dumb dad joke?

What's either a really gross animal issue OR an impressive, magical school?

HAHAHAHA

Moving on…

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Terms

- Information Flow Model
- Noninterference Model
- Bell-LaPadula Model
- Bilba Integrity Model
- Clark-Wilson Model
- Brewer-Nash Model
- Take-Grant Model

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptography Basics

Cryptography is the mathematical manipulation of data so as to protect its confidentiality and/or integrity.

- **Confidentiality** (and privacy): One of the main uses of cryptography is to protect the confidentiality of information, both at rest and in transit. This offers the critical feature of "privacy" when applied to personally identifiable information (PII) and protected health information (PHI).
- **Integrity**: Another common application of cryptography is the use of hashing algorithms and message digest to provide assurance of data integrity (or accuracy). These cryptographic applications help ensure that data being accessed is intact and as expected.
- **Authenticity** (and nonrepudiation): Cryptography can also be used for authentication services as well as nonrepudiation through digital signatures and digital certificates.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptography Basics

- **Plaintext**: The message in its natural format, which has not been turned into a secret.
- **Cleartext**: The message in readable, usable form that is not intended to be obscured by cryptographic means.
- **Ciphertext**: The altered form of a plaintext message, so as to be unreadable for anyone except the intended recipients (in other words, something that has been turned into a secret).
- **Encryption**: The process of converting the message from its plaintext to ciphertext.
- **Decryption**: The reverse process from encryption — it is the process of converting a ciphertext message back into plaintext through the use of the cryptographic algorithm and the appropriate key that was used to do the original encryption.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptography Basics

- **Cryptographic algorithm**: A mathematical function that is used in the encryption and decryption process.

- **Key**: The input that controls the operation of the cryptographic algorithm; it determines the behavior of the algorithm and permits the reliable encryption and decryption of the message. Symmetric/private keys (discussed later in this chapter) must be kept private, while public keys (also discussed later in this chapter) are shared to enable authentication and other use cases.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

The cryptographic lifecycle involves algorithm selection, key management, and the management of encrypted data at rest, in transit, and in use.

- Algorithm selection involves a number of choices:
  - The **type of cryptography** appropriate for the purpose (e.g., symmetric, public key, hashing, etc.)
  - The **specific algorithm** (e.g., AES, RSA, SHA, etc.)
  - The **key length** (e.g., AES-256, RSA-2048, SHA-512, etc.) The operating mode (ECB, CBC, etc.)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

The cryptographic lifecycle involves algorithm selection, key management, and the management of encrypted data at rest, in transit, and in use.

- Algorithm selection involves a number of choices:
  - The **type of cryptography** appropriate for the purpose (e.g., symmetric, public key, hashing, etc.)
  - The **specific algorithm** (e.g., AES, RSA, SHA, etc.)
  - The **key length (*strength)** (e.g., AES-256, RSA-2048, SHA-512, etc.)
    The operating mode (ECB, CBC, etc.)

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

- Symmetric encryption is best for storing data at rest that has to be recovered (decrypted) before being used.
- Symmetric encryption is more efficient than public key cryptography
- For symmetric cryptography to be used for data in transit, both ends of the communications link must have knowledge of the secret key.

There are a number of methods for securely exchanging keys over an insecure channel, the most widely used of which uses public key cryptography (also discussed later).

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

- There are some types of data that need to be protected but that do not need to be decrypted.
- For security, it is best that **some data never be able to be decrypted**. For example, to protect the confidentiality of passwords used for user authentication
  - To verify an entered password against the previously stored password, all one need do is encrypt
  - For this purpose, we use a one-way encryption function, otherwise known as cryptographic hashing. the entered password and compare the two ciphertexts.

Cryptography is **not exclusively** encryption and decryption.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

- Cryptography uses an algorithm that, regardless of its level of complexity, deals with an input and an output.
- Cryptographic hashing inputs a string to produce an output, which is generally assumed to be an output of fixed length, regardless of the input length.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

- A technique to protecting the PAN (Primary Account Number) is to disassociate the number from the account holder and any information that can identify that account holder. In and of itself, the PAN is like a token
- Only when the PAN can be linked to PII does the credit card number become sensitive information.
- Translation vaults are being used to generate a random token with which the PAN is securely linked.
- This process (known as **tokenization**) helps separate account numbers from the PII by providing an encrypted relationship between the two.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

- Select the appropriate algorithm. In some cases, the set of algorithms is constrained by protocol standards.
- To protect data in transit, then there is a finite list of supported algorithms, and your choice is limited by this list as well as a desire to remain compatible with as wide a range of browsers as reasonably possible.

- For IoT and edge devices, you'll generally want algorithms that are lightweight and can be supported by devices that are usually resource constrained.
  - Example of how performance and other system factors make cryptography an important factor in overall system architecture and design.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

- Symmetric cryptography is the use of a **single key**, shared between two or more parties.
- Symmetric algorithms are divided into block and stream ciphers.
  - Block ciphers take a block of data (typically 8, 16, or 32 bytes) at a time, typically used in bulk encryption, such as with data at rest.
  - Block ciphers, in certain chaining modes, are unable to resynchronize, and the loss or corruption of the data stream will make the remainder of the transmission unable to be decrypted.
  - Stream ciphers take either a single bit or a single byte at a time, optimized for encrypting communications links, able to quickly resynchronize in the face of dropped or corrupted bits.
  - It is possible, at the cost of some (or considerable) efficiency to employ a block cipher as a stream cipher and vice versa.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle Considerations

- Efficiency of the algorithm
- Support in the processor instruction set * most Intel, AMD, and ARM processors include instructions to speed up the operation of the AES algorithm
- The longer the key, the more secure the cipher
- Balance the security of long (strong) keys with the impact on system performance.. longer keys mean more processing time.
- Lifetime of the encrypted data.
- Optimal at a point in time, cryptographic keys have finite lifetimes, so do cryptographic algorithms.
- Quantum computing may make certain algorithms obsolete.

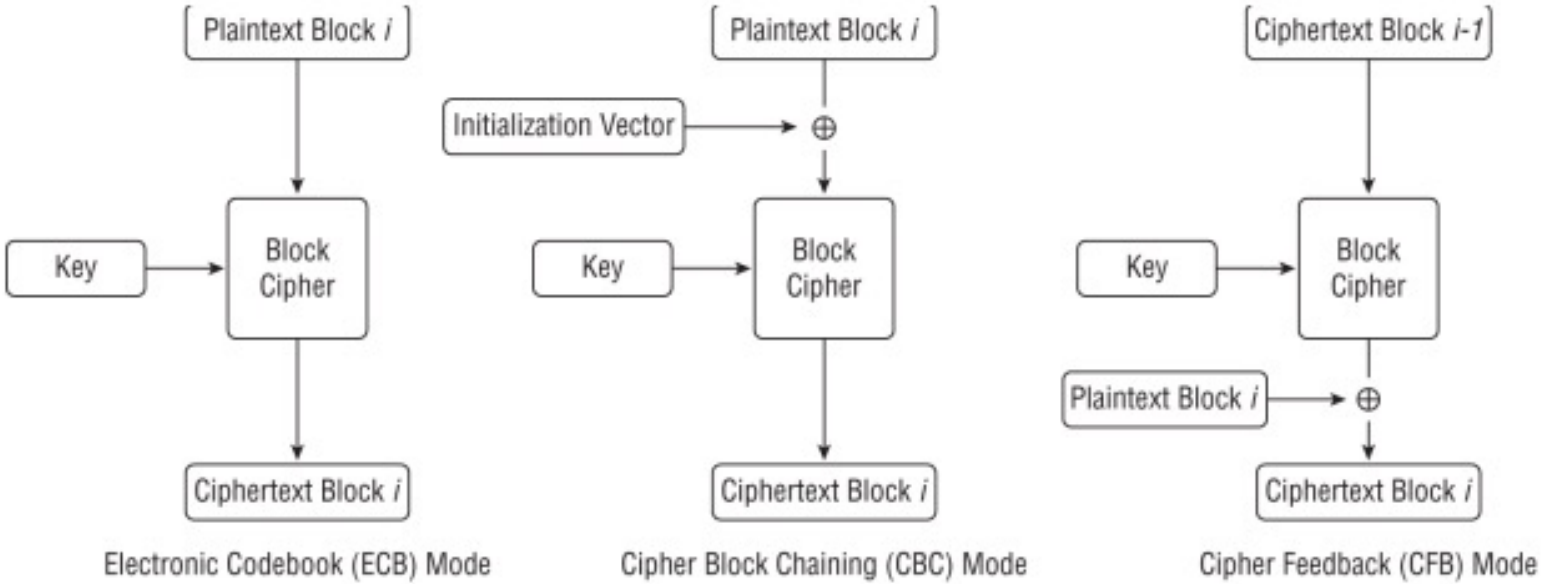
*There have been countless examples in the past of previously considered secure algorithms being deprecated because of advances in cryptanalysis, such as DES, RC4, SHA-1, and others.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

DOMAIN 3: SECURITY ARCHITECTURE AND ENGINEERING

Select and Determine Cryptographic Solutions

Cryptographic Lifecycle

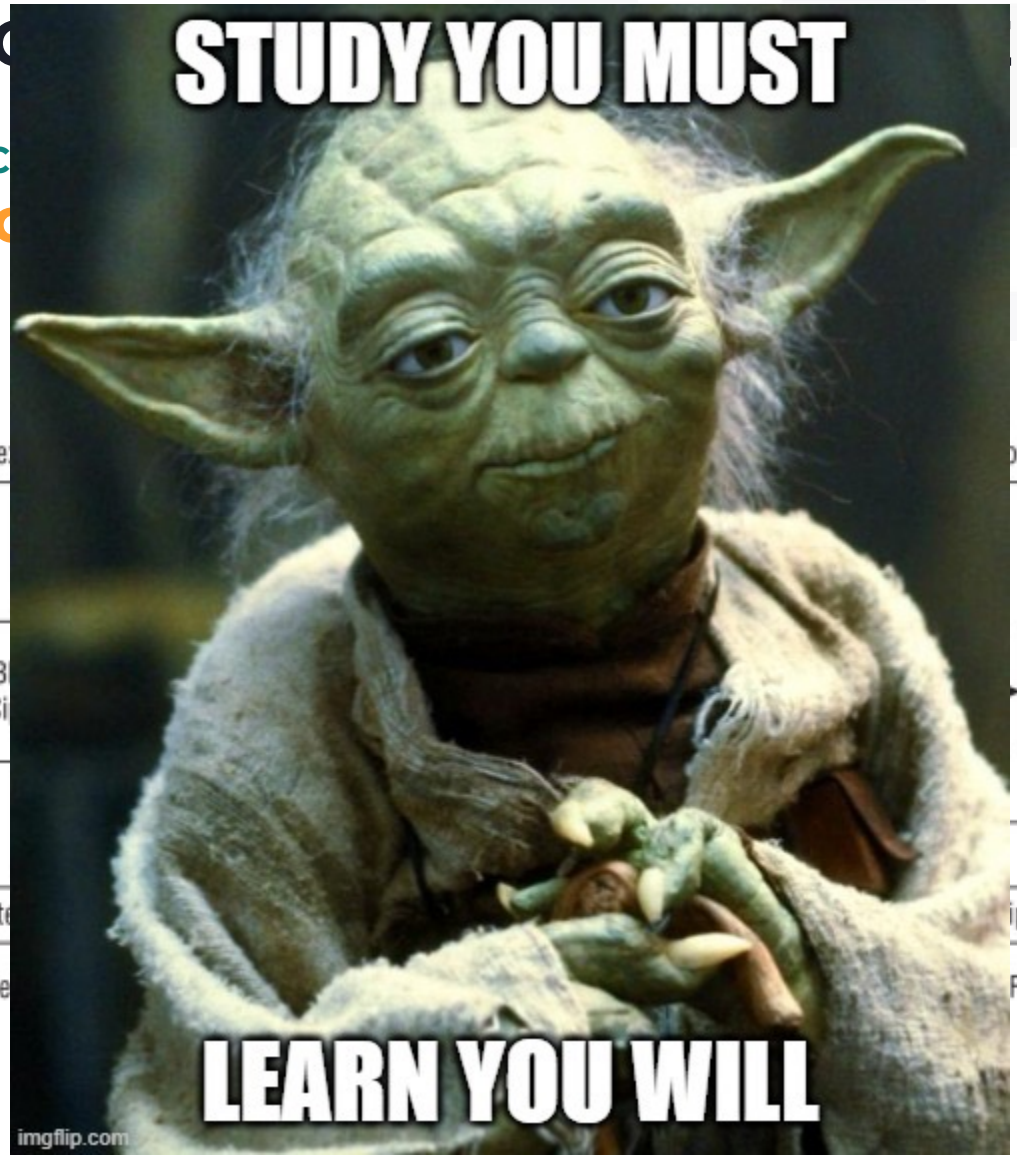


This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION FIVE

DOMAIN 3: SEC NGINEERING

Select and Determine C
Cryptographic



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION TWO

SESSION 5 - FIN
YOU MADE IT!

Domain 3 is 1/2 done WHOOT HECK YA!! YALL!

Domain 3 can be a challenge because it's so dense.

Next Session - Domain 3 part 2– Christophe

- Identification and classification Information and Assets
- Asset handling requirements
- Provision and inventory
- Management
- Roles
- Data Lifecyle and controls

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION TWO

SESSION 5 - FIN
YOU MADE IT!

Domain 3 is 1/2 done WHOOT HECK YA!! YALL!

Domain 3 can be a challenge because it's so dense.

Homework:

- Finish reading Domain 3.
- Take practice tests.
- Review at least two of the references we provided in this class (download for later use).
- Post at least one question/answer in the Slack Channel.

See you next Monday!

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

CISSP® MENTOR PROGRAM – SESSION THREE

HELLO, NICE TO MEET YOU

Ryan Cloutier, CISSP, Tonight's Instructor

@cloutiersec
@StudioSecurity

- President of SecurityStudio
- Virtual Chief Information Security Officer
- Serving the underserved is my passion
- Speaking human about tech is my superpower
- Co-host of the Security Shit Show, and Security Simplified podcast
- Infosec Missionary (helper and protector at heart)
- Published by multiple trade magazines
- Co authored academic papers
- Advisor to many

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

2022

Class #5 – Domain 3

Security Architecture and Engineering

Ryan Cloutier CISSP®
President of SecurityStudio & vCISO

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.