BROKEN ACCESS CONTROL

SCENARIO

An employee of a company discovers a flaw in the financial system by modifying the URL, gaining unauthorized access to other users' reports. This exposes confidential data and could enable fraud due to missing permissions.

CORRECTION

Implement permission verification on the back-end, ensuring that the user only accesses their own data.

SECURITY LOGGING AND MONITORING FAILURES

SCENARIO

An attacker exploits a lack of security monitoring, carrying out malicious actions without being detected due to missing logs and proper analysis.

CORRECTION

Implement detailed logs and continuous monitoring to detect suspicious activity in real time.

INJECTION

SCENARIO

A hacker discovered an SQL injection vulnerability in your user login form. This could potentially allow them to access and manipulate your database.

CORRECTION

Add a data input validation mechanism.

CRYPTOGRAPHIC FAILURES

SCENARIO

An attacker intercepts and decrypts sensitive data due to the use of weak encryption algorithms, exposing information such as passwords and financial data.

CORRECTION

Utilize strong and up-to-date encryption algorithms, apply secure hashing for passwords, and ensure the protection of sensitive data in transit and at rest.

SOFTWARE AND DATA INTEGRITY FAILURES

SCENARIO

An attacker exploits flaws in software integrity by modifying code or data, which compromises the reliability of the system.

CORRECTION

Use integrity checks and digital signatures to ensure that software and data are not altered in an unauthorized manner.

SERVER-SIDE REQUEST FORGERY (SSRF)

SCENARIO

An attacker exploits an SSRF vulnerability to forge server requests, accessing internal resources and causing damage or exfiltrating sensitive data.

CORRECTION

Validate user input and restrict access to sensitive internal resources and services.

IDENTIFICATION AND AUTHENTICATION FAILURES

SCENARIO

An attacker exploits authentication flaws, such as weak passwords or lack of multi-factor authentication, to access user data or accounts without authorization.

CORRECTION

Use multi-factor authentication (MFA) and limit login attempts to protect against brute-force attacks.

SECURITY PATTERN

To fix the system flaw, it is essential to implement permissions on the back-end, ensuring that each user can access only their own data, preventing unauthorized use.

SECURITY PATTERN

Key Rotation and the Use of Strong Encryption Algorithms to protect Sensitive Data.

SECURITY PATTERN

Update on Cryptographic Algorithms and Protocols to Prevent Interception of Sensitive Data

SECURITY PATTERN

Implementing Integrity Verification to Protect Code and Data.

SECURITY PATTERN

Segregation of Duties and Access Control to Ensure Integrity and Reliability.

SECURITY PATTERN

Implementing Multi-Factor Authentication (MFA) to strengthen authentication against weak passwords.

SECURITY PATTERN

Using Strong Passwords and password policies to prevent authentication failures and unauthorized access.

SECURITY PATTERN

Continuous Monitoring and Analysis of Security Logs.

SECURITY PATTERN

Centralization and Secure Storage of Security Monitoring Logs

SECURITY PATTERN

URL Validation and filtering to prevent SSRF vulnerabilities and unauthorized requests.

SECURITY PATTERN

Network and Firewall restrictions to mitigate SSRF vulnerabilities and protect internal resources.


RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD

TESTER

SKILL

You can reveal a hidden vulnerability card in play once per game.

SECURITY PATTERN

Validation and Sanitization of input data to prevent SQL Injection attacks on the database.

SECURITY PATTERN

Implementation of Least Privilege principles to control database access and reduce unauthorized access.

SECURITY FIX RELEASED

EVENT

A new security patch has been released to mitigate a critical vulnerability affecting authentication systems.

ACTION

Each player must choose one of the vulnerabilities from their deck to put on the table, and everyone can try to resolve it.

SECURITY PATCH DEPLOYED

EVENT

A security patch is available for a commonly used library.

ACTION

Each player must exchange cards from their deck with the player next to them.

ZERO-DAY EXPLOIT DISCOVERED

EVENT

A zero-day exploration has been discovered.

ACTION

The player must try to solve a question from the question deck that will be read by someone at the table (worth +2 points).

★★☆☆☆

DEVELOPER

SKILL

You can choose a solution card from the person next to you.

SECURITY ANALYST

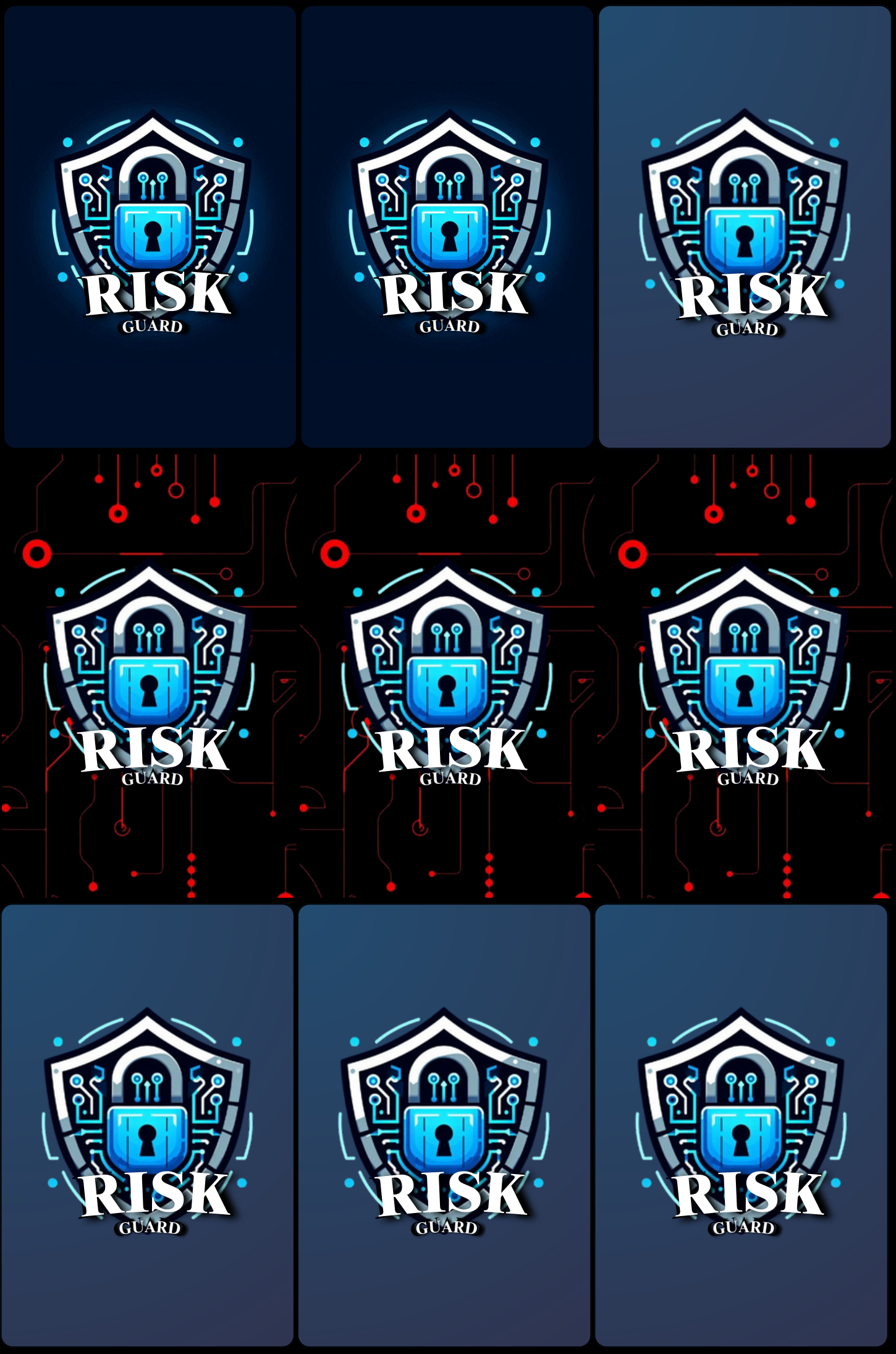
SKILL

You can see which card you will draw from the hand of the player next to you.

PROJECT MANAGER

SKILL

You are allowed to buy a card from any player at the table once per turn.


RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD
RISK
GUARD

TIME FOR THE QUESTION ?

WHAT CHARACTERIZES A CRYPTOGRAPHIC FLAW?

A. The use of strong and up-to-date cryptographic algorithms

B. The improper or outdated use of cryptographic algorithms, or the failure to implement these algorithms correctly.

C. Implementing cryptography in protocols without necessity

D. The use of public encryption keys in private data.

CORRECT ITEM: B

TIME FOR THE QUESTION ?

WHICH OF THE FOLLOWING IS AN EXAMPLE OF A COMMON CRYPTOGRAPHIC FLAW?

A. Encrypting passwords with weak or outdated algorithms, such as MD5 or SHA1.

B. Using AES encryption with a 256-bit key in a secure system.

C. Store passwords in plain text format, without encryption.

D. Use TLS 1.3 for secure communication between servers.

CORRECT ITEM: A

TIME FOR THE QUESTION ?

WHAT ARE THE CONSEQUENCES OF A CRYPTOGRAPHIC FLAW IN A WEB APPLICATION?

A. Leakage of sensitive data, such as passwords and personal information.

B. Improved system performance due to the use of faster algorithms.

C. Increased user confidence due to effective encryption.

D. A security flaw that can be fixed without compromising data.

CORRECT ITEM: A

TIME FOR THE QUESTION ?

WHAT CHARACTERIZES AN INJECTION VULNERABILITY IN A WEB APPLICATION?

A. When data is entered into the system without integrity verification.

B. When an attacker manages to insert malicious commands into data inputs to manipulate the application's behavior.

C. When the system prevents the entry of any type of data.

D. When data is encrypted before being processed.

CORRECT ITEM: B

TIME FOR THE QUESTION ?

WHAT IS AN EFFECTIVE WAY TO PREVENT SQL INJECTION VULNERABILITIES?

A. Allow users to enter any type of data into any field of the application.

B. Use dynamic SQL queries with data entered directly into the SQL statements.

C. Use parameterized queries and stored procedures to interact with the database.

D. Store passwords in plain text format to facilitate authentication.

CORRECT ITEM: C

TIME FOR THE QUESTION ?

WHAT ARE THE CONSEQUENCES OF A SUCCESSFUL SQL INJECTION?

A. The application can be made faster and more efficient.

B. The attacker can obtain, modify, or delete data from the database without permission.

C. The application becomes completely immune to other types of attacks.

D. None of the above.

CORRECT ITEM: B

TIME FOR THE QUESTION ?

WHICH OF THE OPTIONS BELOW IS AN EXAMPLE OF AN AUTHENTICATION FAILURE?

A. Require the user to provide contact information.

B. Validate the user's identity with a unique code sent via email.

C. Authenticate a user with a temporary security token.

D. To allow access to a system without requiring a password or other form of authentication.

CORRECT ITEM: D

TIME FOR THE QUESTION ?

WHAT CHARACTERIZES A SOFTWARE AND DATA INTEGRITY FAILURE?

A. The unauthorized alteration or corruption of data or code in a system.

B. Implementing strong encryption across all communications.

C. Securely storing passwords in databases.

D. The use of digital certificates for authentication.

CORRECT ITEM: A

TIME FOR THE QUESTION ?

HOW CAN AN ATTACKER EXPLOIT FLAWS IN AN ACCESS CONTROL SYSTEM?

A. Attempting to guess the system password through brute-force attacks.

B. Obtaining undue elevated permissions or accessing resources for which you are not authorized.

C. Using an insecure communication protocol.

D. Modifying data encryption without authorization.

CORRECT ITEM: B


TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION

TIME OF
QUESTION


TURN MARKER
RISK
GUARD


TURN MARKER
RISK
GUARD


TURN MARKER
RISK
GUARD


TURN MARKER
RISK
GUARD


TURN MARKER
RISK
GUARD

BROKEN ACCESS CONTROL

SCENARIO

An employee of a company discovers a flaw in the financial system by modifying the URL, gaining unauthorized access to other users' reports. This exposes confidential data and could enable fraud due to missing permissions.

CORRECTION

Implement permission verification on the back-end, ensuring that the user only accesses their own data.

Pattern

SECURITY PATTERN

To fix the system flaw, it is essential to implement permissions on the back-end, ensuring that each user can access only their own data, preventing unauthorized use.

CRYPTOGRAPHIC FAILURES

SCENARIO

An attacker intercepts and decrypts sensitive data due to the use of weak encryption algorithms, exposing information such as passwords and financial data.

CORRECTION

Utilize strong and up-to-date encryption algorithms, apply secure hashing for passwords, and ensure the protection of sensitive data in transit and at rest.

Patterns

SECURITY PATTERN

Key Rotation and the Use of Strong Encryption Algorithms to protect Sensitive Data.

SECURITY PATTERN

Update on Cryptographic Algorithms and Protocols to Prevent Interception of Sensitive Data

SOFTWARE AND DATA INTEGRITY FAILURES

SCENARIO

An attacker exploits flaws in software integrity by modifying code or data, which compromises the reliability of the system.

CORRECTION

Use integrity checks and digital signatures to ensure that software and data are not altered in an unauthorized manner.

Patterns

SECURITY PATTERN

Implementing Integrity Verification to Protect Code and Data.

SECURITY PATTERN

Segregation of Duties and Access Control to Ensure Integrity and Reliability.

IDENTIFICATION AND AUTHENTICATION FAILURES

SCENARIO

An attacker exploits authentication flaws, such as weak passwords or lack of multi-factor authentication, to access user data or accounts without authorization.

CORRECTION

Use multi-factor authentication (MFA) and limit login attempts to protect against brute-force attacks.

Patterns

SECURITY PATTERN

Implementing Multi-Factor Authentication (MFA) to strengthen authentication against weak passwords.

SECURITY PATTERN

Using Strong Passwords and password policies to prevent authentication failures and unauthorized access.

SECURITY LOGGING AND MONITORING FAILURES

SCENARIO

An attacker exploits a lack of security monitoring, carrying out malicious actions without being detected due to missing logs and proper analysis.

CORRECTION

Implement detailed logs and continuous monitoring to detect suspicious activity in real time.

Patterns

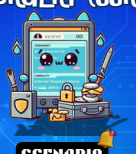
SECURITY PATTERN

Continuous Monitoring and Analysis of Security Logs.

SECURITY PATTERN

Centralization and Secure Storage of Security Monitoring Logs

SERVER-SIDE REQUEST FORGERY (SSRF)



SCENARIO

An attacker exploits an SSRF vulnerability to forge server requests, accessing internal resources and causing damage or exfiltrating sensitive data.

CORRECTION

Validate user input and restrict access to sensitive internal resources and services.

Patterns

SECURITY PATTERN

URL Validation and filtering to prevent SSRF vulnerabilities and unauthorized requests.

SECURITY PATTERN

Network and Firewall restrictions to mitigate SSRF vulnerabilities and protect internal resources.

INJECTION

SCENARIO

A hacker discovered an SQL injection vulnerability in your user login form. This could potentially allow them to access and manipulate your database.

CORRECTION

Add a data input validation mechanism.

Patterns

SECURITY PATTERN

Validation and Sanitization of input data to prevent SQL Injection attacks on the database.

SECURITY PATTERN

Implementation of Least Privilege principles to control database access and reduce unauthorized access.