# 🏛️ NIST Risk Management Framework (RMF)

**Overview**

The NIST Risk Management Framework provides a structured approach to managing security and privacy risks.

**The Seven Steps**

### 1️⃣ PREPARE

Essential activities to prepare the organization to manage security and privacy risks.

**Activities**:

- Establish risk management strategy
- Define roles and responsibilities
- Identify stakeholders
- Prepare organizational risk assessment

**Tools**:

- Asset inventory tools
- Documentation templates
- Risk registers

### 2️⃣ CATEGORIZE

Categorize the system and information processed, stored, and transmitted based on impact analysis.

**Process**:

Impact Analysis → Categorization → Documentation

**Categories**:

- **Confidentiality (C)**: 1-3
- **Integrity (I)**: 1-3
- **Availability (A)**: 1-3

**Example**:

System: Web Application

C: 2 (Moderate)

I: 3 (High)

A: 2 (Moderate)

Overall: HIGH (highest value)

**Impact Levels**:

- **Low (1)**: Limited adverse effect
- **Moderate (2)**: Serious adverse effect
- **High (3)**: Severe or catastrophic adverse effect

### 3️⃣ SELECT

Select NIST SP 800-53 controls to protect the system based on risk assessment.

**Process**:

1. Identify baseline controls
2. Tailor controls to organizational needs
3. Document control selection
4. Develop implementation plan

**Control Families**:

- Access Control (AC)
- Awareness and Training (AT)
- Audit and Accountability (AU)
- Configuration Management (CM)
- Identification and Authentication (IA)
- Incident Response (IR)
- And 14 more families...

## 4️⃣ IMPLEMENT

Implement the controls and document how controls are deployed.

**Activities**:

- Deploy selected controls
- Configure security settings
- Document implementation
- Create system security plans

**Implementation Example**:

bash

```
# Implement firewall rules (AC-4: Information Flow Enforcement)
sudo ufw enable
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow 22/tcp
sudo ufw allow 443/tcp
```

## 5️⃣ ASSESS

Assess to determine if controls are in place, operating as intended, and producing desired results.

**Assessment Methods**:

- **Testing**: Hands-on validation
- **Examination**: Documentation review
- **Interviewing**: Personnel discussions

**Tools for Assessment**:

- Nessus
- OpenVAS
- Qualys
- Manual testing

- Compliance scanners

### 6️⃣ AUTHORIZE

Senior official makes risk-based decision to authorize the system to operate.

**Authorization Decision**:

Accept Risk → Authorize to Operate (ATO)

or

Reject Risk → Deny Authorization

or

Conditional → Authorize with Conditions

**Documents Required**:

- System Security Plan
- Security Assessment Report
- Plan of Action and Milestones (POA&M)
- Risk Assessment Report

### 7️⃣ MONITOR

Continuously monitor control implementation and risks to the system.

**Monitoring Activities**:

- Continuous vulnerability scanning
- Security event monitoring
- Configuration management
- Incident tracking
- Regular reassessment

**Tools**:

bash

```
# Continuous monitoring with Nessus
# Schedule: Daily for critical, Weekly for others

# SIEM integration
# Splunk, ELK Stack, QRadar

# Automated compliance checking
# OpenSCAP, Chef InSpe
```

## 🔐 CIA Triad Mapping

### The CIA Triad

The foundation of information security is based on three principles:

#### Confidentiality

Ensuring information is accessible only to authorized individuals.

**Threats**:

- Unauthorized access
- Data breaches
- Eavesdropping
- Social engineering

**Controls**:

- Encryption (AES-256)
- Access control lists
- Authentication (MFA)
- Data classification

**Example Implementation**:

bash

```
# Encrypt sensitive file
openssl enc -aes-256-cbc -salt -in file.txt -out file.txt.enc

# Set file permissions
chmod 600 sensitive_file.txt

# Implement file encryption at rest
```

### ✅ Integrity

Ensuring information remains accurate and unaltered.

**Threats**:

- Unauthorized modification
- Malware
- Human error
- System malfunctions

**Controls**:

- Hashing (SHA-256)
- Digital signatures
- Version control
- Input validation
- Checksums

**Example Implementation**:

bash

```
# Create file hash
sha256sum file.txt > file.txt.sha256
```

```
# Verify integrity
sha256sum -c file.txt.sha256

# Digital signature
gpg --sign document.pdf
```

### ⚡ Availability

Ensuring information and systems are accessible when needed.

**Threats**:

- DDoS attacks
- Hardware failures
- Natural disasters
- Power outages

**Controls**:

- Redundancy
- Backups
- Load balancing
- Disaster recovery
- UPS systems

**Example Implementation**:

bash

```
# Automated backups
rsync -avz /data/ /backup/

# Database replication
# Master-slave configuration
# Load balancer setup
# HAProxy, Nginx
```

### 📈 Risk Components

**1. Risk Identification**

**Purpose**: Identify potential threats and vulnerabilities

**Methods**:

- Asset inventory
- Threat modeling
- Vulnerability scanning
- Historical data analysis
- Stakeholder interviews

**Documentation**:

markdown

```
| Asset | Threat | Vulnerability | Current Controls |
|-------|--------|---------------|------------------|
| Web Server | SQL Injection | Unvalidated input | None |
| Database | Unauthorized access | Weak passwords | Basic auth |
```

**2. Risk Evaluation**

**Purpose**: Analyze and evaluate identified risks

**Evaluation Criteria**:

- Likelihood of occurrence
- Potential impact
- Existing controls

- Cost of mitigation
- Regulatory requirements

**Qualitative vs Quantitative**:

**Qualitative Assessment**:

High, Medium, Low ratings

Based on expert judgment

Faster but less precise

**Quantitative Assessment**:

Numerical values

Statistical analysis

More precise but time-consuming

Example:

Annual Loss Expectancy (ALE) = SLE × ARO

Where:

- SLE = Single Loss Expectancy

- ARO = Annual Rate of Occurrence

**3. Risk Matrix**

Visual representation of risk levels based on impact and likelihood.

**Risk Matrix Template**

```
                IMPACT
            Low   Medium   High
          ┌───────┬──────────┬───────┐
   Low    │ Low │ Low    │ Med │
LIKELIHOOD ├───────┼──────────┼───────┤
   Medium │ Low │ Medium │ High │
```

High | Med | High | High |

**Risk Scoring**

**Formula**: Risk Score = Impact × Likelihood

**Example Calculations**:

**Scenario 1: SQL Injection**

Impact: 3 (High - Data breach)

Likelihood: 3 (High - No input validation)

Risk Score: 3 × 3 = 9 (Critical)

**Scenario 2: Weak Password Policy**

Impact: 2 (Medium - Unauthorized access)

Likelihood: 2 (Medium - Some controls exist)

Risk Score: 2 × 2 = 4 (Medium)

**Scenario 3: Outdated Software**

Impact: 3 (High - Known vulnerabilities)

Likelihood: 1 (Low - Patching schedule exists)

Risk Score: 3 × 1 = 3 (Low-Medium)

**Risk Levels**

| Score | Level | Priority | Action Required |
|---|---|---|---|
| 9 | Critical | Immediate | Fix within 24 hours |
| 6-8 | High | Urgent | Fix within 7 days |
| 3-5 | Medium | Important | Fix within 30 days |
| 1-2 | Low | Monitor | Fix within 90 days |

## 4. Risk Treatment

Four primary strategies for managing identified risks.

### Risk Mitigation (Reduce)

**Definition**: Implement controls to reduce the likelihood or impact of risk.

**Examples**:

- Install antivirus software
- Implement firewalls
- Apply security patches
- Enable logging and monitoring
- Conduct security training

**Implementation**:

bash

```
# Example: Mitigate SSH brute-force attacks
sudo apt-get install fail2ban

# Configure fail2ban
sudo nano /etc/fail2ban/jail.local
[sshd]
enabled = true
maxretry = 3
bantime = 3600

sudo systemctl restart fail2ban
```

**Cost**: Moderate to High **Effectiveness**: High

### Risk Avoidance (Eliminate)

**Definition**: Eliminate the risk entirely by not engaging in the risky activity.

**Examples**:

- Discontinue vulnerable service
- Remove legacy systems
- Cancel high-risk project
- Disable unnecessary features
- Block risky network protocols

**Implementation**:

bash

```
# Example: Avoid risk by disabling FTP
sudo systemctl stop vsftpd
sudo systemctl disable vsftpd

# Remove unnecessary services
sudo apt-get remove telnetd
```

**Cost**: Varies **Effectiveness**: Complete (100%)

### Risk Transference (Share/Transfer)

**Definition**: Transfer the risk to a third party.

**Examples**:

- Cyber insurance policies
- Outsource to managed security service provider (MSSP)
- Cloud service providers (shared responsibility model)
- Third-party security audits
- Bug bounty programs

**Implementation**:

markdown

```
# Example: Transfer risk through insurance
1. Purchase cyber liability insurance
2. Coverage includes:
   - Data breach response costs
   - Legal fees
   - Notification costs
   - Credit monitoring
   - Business interruption

Annual Premium: $10,000
Coverage: $1,000,000
```

**Cost**: Insurance premiums, service fees **Effectiveness**: Reduces financial impact

### ✅ Risk Acceptance (Accept)

**Definition**: Accept the risk when the cost of mitigation exceeds the potential loss.

**When to Accept**:

- Low impact and low likelihood
- Cost of mitigation is too high
- No practical solution exists
- Residual risk after mitigation
- Business justification

**Requirements**:

- Formal acceptance by senior management
- Documented justification
- Regular review

- Monitoring plan

**Documentation Template**:

markdown

# Risk Acceptance Form

**Risk ID**: RISK-001

**Description**: Legacy system cannot be patched

**Risk Score**: 4 (Medium)

**Justification**:

- System will be decommissioned in 6 months

- Cost of mitigation: $50,000

- Potential loss: $10,000

- System isolated on separate network

**Accepted By**: [Senior Manager Name]

**Date**: [Date]

**Review Date**: [3 months]

**Cost**: Potential loss if risk materializes **Effectiveness**: N/A (Risk remains)

**CVE (Common Vulnerabilities and Exposures)**

**What is CVE?**

CVE is a list of publicly disclosed information security vulnerabilities and exposures maintained by MITRE Corporation.

**CVE Format**

CVE-YEAR-NUMBER

Example: CVE-2021-44228 (Log4Shell)

**CVE Components**

**CVE ID**: Unique identifier **Description**: Detailed explanation **References**: Links to advisories **Date**: Publication date

**Finding CVEs**

**1. CVE Database Search**

bash

```
# Search CVE database
# Visit: https://cve.mitre.org/


# Search by software
Example: "Apache 2.4.49"


# Search by CVE ID
Example: CVE-2021-41773
```

**2. Using Nmap NSE**

bash

```
# Check for CVE vulnerabilities
nmap --script vulners <target>
nmap --script vulscan <target>
```

**3. National Vulnerability Database (NVD)**

```
URL: https://nvd.nist.gov/
Features:
- CVSS scores
- CPE names
- CWE categories
- Patch information
```

**High-Profile CVEs**

**CVE-2021-44228 (Log4Shell)**

Severity: 10.0 (Critical)

Affected: Apache Log4j 2.0 - 2.14.1

Impact: Remote Code Execution

**CVE-2017-0144 (EternalBlue)**

Severity: 8.1 (High)

Affected: Windows SMB

Impact: Remote Code Execution

Used in: WannaCry ransomware

**CVE-2014-0160 (Heartbleed)**

Severity: 7.5 (High)

Affected: OpenSSL

Impact: Information Disclosure

## CVSS (Common Vulnerability Scoring System)

### CVSS Overview

CVSS provides a standardized method for rating IT vulnerabilities on a scale of 0-10.

### CVSS v3.1 Metrics

**Base Metrics (Intrinsic qualities)**

**Attack Vector (AV)**:

- Network (N): 0.85
- Adjacent (A): 0.62
- Local (L): 0.55
- Physical (P): 0.2

**Attack Complexity (AC)**:

- Low (L): 0.77
- High (H): 0.44

**Privileges Required (PR)**:

- None (N): 0.85
- Low (L): 0.62
- High (H): 0.27

**User Interaction (UI)**:

- None (N): 0.85
- Required (R): 0.62

**Scope (S)**:

- Unchanged (U)
- Changed (C)

**Impact Metrics**:

- Confidentiality (C): None, Low, High
- Integrity (I): None, Low, High
- Availability (A): None, Low, High

**CVSS Score Ranges**

| **Score** | **Severity** | **Action** |
|---|---|---|
| 0.0 | None | Informational |
| 0.1-3.9 | Low | Schedule fix |
| 4.0-6.9 | Medium | Plan remediation |
| 7.0-8.9 | High | Urgent fix |
| 9.0-10.0 | Critical | Immediate action |

**CVSS Calculator**

**Online Calculators**:

- https://www.first.org/cvss/calculator/3.1
- https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

**Example CVSS Calculation**

**Vulnerability**: SQL Injection

Attack Vector: Network (AV:N)

Attack Complexity: Low (AC:L)

Privileges Required: None (PR:N)

User Interaction: None (UI:N)

Scope: Unchanged (S:U)

Confidentiality: High (C:H)

Integrity: High (I:H)

Availability: High (A:H)

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Base Score: 9.8 (Critical)

**Using CVSS in Reporting**

markdown

## Vulnerability: SQL Injection

**CVSS v3.1 Score**: 9.8 (Critical)

**Vector**: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

**Metrics Breakdown**:

- Attack Vector: Network (Can be exploited remotely)
- Attack Complexity: Low (No special conditions required)
- Privileges Required: None (Unauthenticated)
- User Interaction: None (Fully automated)
- Confidentiality Impact: High (Total data disclosure)
- Integrity Impact: High (Data can be modified)
- Availability Impact: High (System can be disrupted)