Title: NIST Compliant Incident Response Framework Playbook for SQL Injection and Command Injection Attacks

* Question: You are a security operations system analyst and helpful cybersecurity assistant that generates detailed incident response playbooks based on NIST guidelines. please generate a playbook for a SQL injection and command injection attacks effectively?
* Answer:

1. Preparation:

Objective: Establish contacts, define procedures, and gather information to save time during an incident.

- \*\*Identify Critical Applications\*\*:

- \*\*Documentation\*\*: Maintain an up-to-date inventory of OT applications, web interfaces, and APIs interacting with databases or executing commands. Include technology stacks and configurations.

- \*\*Prioritization\*\*: Assign criticality levels to applications based on their operational roles and injection attack impact.

- \*\*Security Assessments\*\*:

- \*\*Regular Vulnerability Scans\*\*: Schedule scans and penetration tests targeting SQL and command injection vulnerabilities.

- \*\*Code Reviews\*\*: Conduct regular reviews to ensure proper input validation and output encoding.

- \*\*Patch Management\*\*: Implement a patching process for timely updates.

- \*\*Incident Response Team (IRT)\*\*:

- \*\*Team Formation\*\*: Assemble members from IT, OT, and cybersecurity departments, including roles like incident commander, database administrators, and application security experts.

- \*\*Roles and Responsibilities\*\*: Define clear roles and ensure redundancy for critical positions.

- \*\*Training\*\*:

- \*\*Regular Training\*\*: Conduct simulation drills and exercises on injection attack scenarios.

- \*\*Secure Coding Practices\*\*: Train developers and engineers on secure coding to prevent vulnerabilities.

- \*\*Tools\*\*:

- \*\*Web Application Firewalls (WAF)\*\*: Protect web applications from SQL and command injection.

- \*\*Database Activity Monitoring (DAM)\*\*: Monitor and log database activities for anomalies.

- \*\*Intrusion Detection Systems (IDS)\*\*: Detect injection-related malicious activities.

2. Detection and Analysis:

Objective: Detect the incident, determine its scope, and involve appropriate parties.

- \*\*Anomaly Detection\*\*:

- \*\*Behavioral Analytics\*\*: Detect deviations such as unusual database queries or command executions.

- \*\*Threshold Alerts\*\*: Configure alerts for high database query volumes or abnormal command executions.

- \*\*Log Monitoring\*\*:

- \*\*Continuous Monitoring\*\*: Analyze logs from applications, databases, and servers for attack indicators.

- \*\*Log Correlation\*\*: Use tools to correlate related events and identify suspicious patterns.

- \*\*Real-Time Alerts\*\*:

- \*\*SIEM Integration\*\*: Centralize logs for real-time monitoring and alerting.

- \*\*Alert Prioritization\*\*: Prioritize based on severity and application criticality.

- \*\*Incident Categorization\*\*:

- \*\*Attack Identification\*\*: Determine if the attack is SQL injection or command injection by analyzing patterns.

- \*\*Scope Determination\*\*: Assess affected applications, databases, and systems.

- \*\*Impact Assessment\*\*:

- \*\*Operational Impact\*\*: Evaluate implications such as data breaches, unauthorized command executions, and downtime.

- \*\*Resource Utilization\*\*: Analyze CPU, memory, and bandwidth usage during the attack.

- \*\*Source Identification\*\*:

- \*\*IP Tracking\*\*: Identify IP addresses involved and analyze geolocation data.

- \*\*Payload Analysis\*\*: Study injected SQL queries or commands for attacker intent.

3. Containment:

Objective: Mitigate the attack’s effects on the targeted environment.

- \*\*Traffic Filtering\*\*:

- \*\*WAF Rules\*\*: Apply rules to filter malicious traffic based on detected patterns.

- \*\*IP Blacklisting\*\*: Block malicious IPs at the firewall level.

- \*\*Rate Limiting\*\*:

- \*\*Traffic Throttling\*\*: Prevent overwhelming system resources by limiting traffic.

- \*\*Adaptive Controls\*\*: Dynamically adjust thresholds based on real-time analysis.

- \*\*Access Control\*\*:

- \*\*Restrict Access\*\*: Temporarily block access to affected applications or endpoints.

- \*\*Enhanced Authentication\*\*: Implement MFA for accessing critical systems.

4. Eradication:

Objective: Remove the threat and prevent recurrence.

- \*\*Mitigation Tools\*\*:

- \*\*Database Security Tools\*\*: Scan and fix vulnerabilities in databases.

- \*\*Command Execution Controls\*\*: Restrict potentially dangerous server commands.

- \*\*Patch and Update\*\*:

- \*\*Vulnerability Patching\*\*: Apply patches to close exploited vulnerabilities.

- \*\*Security Updates\*\*: Update underlying systems, libraries, and dependencies.

5. Recovery:

Objective: Restore the system to normal operations.

- \*\*System Checks\*\*:

- \*\*Integrity Verification\*\*: Ensure applications and systems are functioning correctly.

- \*\*Health Monitoring\*\*: Track application and database performance metrics.

- \*\*Traffic Normalization\*\*:

- \*\*Gradual Restoration\*\*: Restore traffic flow while monitoring for residual activity.

- \*\*Baseline Comparison\*\*: Compare traffic patterns against established baselines.

- \*\*Data Integrity\*\*:

- \*\*Data Verification\*\*: Confirm data and configuration integrity.

- \*\*Backup Restoration\*\*: Restore corrupted or lost data from secure backups.

6. Lessons Learned:

Objective: Document the incident’s details, discuss lessons learned, and adjust plans and defenses.

- \*\*Documentation\*\*:

- \*\*Incident Report\*\*: Record attack type, detection methods, response actions, timeline, and impact.

- \*\*Lessons Learned\*\*: Identify gaps and improvement opportunities.

- \*\*Review and Improve\*\*:

- \*\*Post-Incident Review\*\*: Evaluate response effectiveness with stakeholders.

- \*\*Plan Updates\*\*: Revise IR plans based on findings.

- \*\*Training Update\*\*:

- \*\*Incorporate Learnings\*\*: Update training programs with new knowledge.

- \*\*Ongoing Training\*\*: Schedule regular awareness sessions to maintain readiness.