🧪 Experiment 3 – Static Application Security Testing (SAST) using Bandit

Aim: Detect code-level vulnerabilities using Bandit.
Concept: Analyze source code without executing it (white-box testing).

Question	Answer
1. What is SAST?	Static Application Security Testing analyzes source code for vulnerabilities before execution.
2. How is SAST different from DAST?	SAST = code-level scanning (before run); DAST = black-box testing (while running).
3. Why is Bandit used?	Bandit scans Python code for common security issues (e.g., eval(), hardcoded passwords, SQL injection).
4. What vulnerabilities can it detect?	SQL injection, command injection, insecure hash usage, hardcoded credentials.
5. Why is SAST done early in SDLC?	Fixing bugs early saves time and cost, ensures secure code from the start.
6. Example of a preventive measure for SQL Injection?	Use parameterized queries or ORM instead of string concatenation.

🧪 Experiment 4 – OWASP Methodologies

Aim: Study and implement at least 5 OWASP methodologies.

Question	Answer
1. What is OWASP?	Open Web Application Security Project — nonprofit that provides free tools and best practices for web security.
2. What are OWASP Top 10 vulnerabilities?	Injection, Broken Authentication, Sensitive Data Exposure, Security Misconfiguration, XSS, etc.

3. What is Injection and how to prevent it?	Sending untrusted data to interpreter (e.g., SQL injection). Prevent using input validation and prepared statements.
4. What is Broken Authentication?	Weak login/session handling. Prevent with MFA and strong session management.
5. What is Sensitive Data Exposure?	Leakage of confidential data. Prevent using HTTPS and encryption.
6. What is XSS?	Cross-Site Scripting—injecting malicious scripts. Prevent by escaping/validating user inputs and using CSP.
7. What is Security Misconfiguration?	Insecure default settings or unused services open. Prevent with regular audits and hardening configurations.
8. What is OWASP ZAP used for?	Tool for finding vulnerabilities via automated and manual web scans.
9. Difference between active and passive scan?	Passive = observes traffic; Active = sends test requests to find flaws.

🧪 Experiment 5 – OS Command Injection using PortSwigger

Aim: Demonstrate OS command injection and its prevention.

Question	Answer
1. What is OS Command Injection?	When user input is passed to a system command unsafely, allowing execution of OS commands.
2. Example of OS Command Injection?	`ping 127.0.0.1; whoami` – attacker adds `;whoami` to execute extra commands.
3. Why does this occur?	Because of poor input validation or unsafe functions like `system()` and `exec()`.
4. How can it be prevented?	Input sanitization, whitelisting, avoiding shell calls, using safe APIs.
5. What is PortSwigger Academy?	Free online platform for learning and practicing web vulnerabilities.
6. What are command separators in Linux?	;, &&, `

7. What is the impact of this vulnerability?	Unauthorized access, privilege escalation, data theft.

🧪 Experiment 6 – Data Validation (Registration Page Validation)

Aim: Apply frontend + backend validation.

Question	Answer
1. What is data validation?	Ensuring user input is correct, complete, and secure before saving or processing.
2. Why is data validation important?	Prevents wrong/malicious data, improves security and user experience.
3. Difference between client-side & server-side validation?	Client-side (browser, JS) is fast but bypassable; server-side (backend) is secure and reliable.
4. Example of validation checks?	Email format, password strength, phone number length, confirm password match.
5. Why implement both validations?	Client-side = user convenience; Server-side = actual data protection.
6. What is authentication?	Verifying user identity before granting access (login).
7. How can passwords be stored securely?	Hash with bcrypt/SHA256, not plain text.
8. What attacks does validation prevent?	SQL Injection, XSS, fake registrations.

🧪 Experiment 7 – Session Management using Flask

Aim: Study and implement secure session management.

Question	Answer
1. What is session management?	Maintaining user state across multiple HTTP requests.
2. Why is it needed?	HTTP is stateless—sessions track logged-in users and store their data securely.

3. What is a session ID?	Unique, random string identifying a user’s session.
4. Where is session ID stored?	In cookies on the client side.
5. How does the server use session ID?	To fetch user’s session data from memory or database.
6. What are alternate session tracking methods?	URL rewriting, hidden form fields, or tokens (JWT).
7. What is session hijacking?	Stealing a session ID to impersonate a user.
8. How to prevent session hijacking?	Use HTTPS, secure cookies, regenerate session IDs, set timeouts.

🧪 Experiment 8 – Burp Suite Proxy

Aim: Use Burp Suite to test web applications.

Question	Answer
1. What is Burp Suite?	A tool for web security testing that intercepts and modifies HTTP(S) traffic.
2. What is Burp Proxy used for?	Intercepting and analyzing requests/responses between browser and server.
3. What are main components of Burp Suite?	Proxy, Repeater, Intruder, Scanner, Sequencer, Target, Logger.
4. What is the Repeater tool?	Manually modify and resend requests to analyze server responses.
5. What is the Intruder tool?	Automates attacks like fuzzing and brute-forcing parameters.
6. What is the Sequencer tool?	Tests randomness of session tokens and CSRF tokens.
7. Why install Burp CA certificate?	To decrypt HTTPS traffic without browser warnings.
8. Why define scope?	To ensure only authorized domains are tested and avoid legal issues.
9. What is the difference between passive and active scanning?	Passive = safe observation, Active = sends test payloads to find vulnerabilities.

10. What precautions should be taken?	Always get authorization, define scope, avoid scanning live systems.

🧪 Experiment 1 – Study of Cybersecurity Laws and Standards

📘 Aim:

To study different laws and standards of cybersecurity.

🔹 Concept-Based Viva Questions

Question	Answer
1. What is Cybersecurity?	Cybersecurity is the practice of protecting systems, networks, and data from digital attacks, unauthorized access, and damage.
2. Why is Cybersecurity important?	It protects personal and business data, prevents financial loss, maintains privacy, and ensures business continuity.
3. What are the main components of cybersecurity?	People, Processes, and Technology.
4. What are the major types of Cyber Attacks?	Web-based (SQL Injection, XSS, Phishing) and System-based (Virus, Worms, Trojans, Backdoors).
5. What is an Injection Attack?	When malicious data is inserted into a web app to manipulate it (e.g., SQL Injection).
6. What is Phishing?	Tricking users into sharing sensitive info by pretending to be a trusted entity.
7. What is a Denial-of-Service (DoS) attack?	Flooding a system or server with traffic to make it unavailable.
8. What is a Man-in-the-Middle (MitM) attack?	When an attacker intercepts communication between two parties.
9. What is a Trojan Horse?	A malicious program disguised as legitimate software that executes harmful code.

10. What is the IT Act 2000?	India’s main law for cybercrime and e-commerce. Defines offenses like hacking and identity theft.
11. What is GDPR?	EU regulation for protecting personal data and privacy.
12. What is HIPAA?	U.S. law to protect health information security and privacy.
13. What is ISO/IEC 27001?	An international standard for Information Security Management Systems (ISMS).
14. What is NIST Cybersecurity Framework?	U.S. framework defining core security functions – Identify, Protect, Detect, Respond, Recover.
15. What is PCI-DSS?	Standard ensuring secure processing of credit card payments.
16. What is CERT-IN?	India’s nodal agency for managing cybersecurity incidents.
17. What is Section 66 of the IT Act?	Deals with hacking and unauthorized access to computer systems.
18. What is Section 67 of the IT Act?	Punishes publishing obscene content in electronic form.
19. What are the core cybersecurity goals?	Confidentiality, Integrity, Availability (CIA Triad).
20. What are Web-based and System-based attacks?	Web-based: target web apps (SQLi, XSS). System-based: target OS or network (Virus, Worm, Trojan).

🧩 Conclusion:

Understanding cybersecurity laws and standards helps secure systems, ensure compliance, and reduce risk from cyber threats.

🧪 Experiment 2 – Case Study on SDLC with Secure Development Practices

📘 Aim:

Case study on Secure SDLC using “Secure Scholarship Application System.”

🔹 Concept-Based Viva Questions

Question	Answer
1. What is SDLC?	Software Development Life Cycle — process to design, develop, and maintain software systematically.
2. What is Secure SDLC (SSDLC)?	An enhanced SDLC that integrates security at every phase of software development.
3. Why is Secure SDLC important?	It reduces vulnerabilities early, ensures data protection, and lowers the cost of fixing issues later.
4. List the phases of Secure SDLC.	System Investigation, System Analysis, Logical Design, Physical Design, Implementation, Maintenance.
5. What happens in System Investigation?	Identify goals, risks, and define security policy — refer to ISO 27001, OWASP Top 10, NIST.
6. What is done in System Analysis?	Analyze data sensitivity, access control, and perform threat modeling using STRIDE/DREAD.
7. What happens in Logical Design?	Create secure architecture, define access control (RBAC), and set trust boundaries.
8. What is done in Physical Design?	Implement encryption (AES-256), HTTPS, IDS/IPS, 2FA, and secure database design.
9. What is Implementation Phase?	Apply secure coding practices, use OWASP guidelines, and static analysis tools like SonarQube.
10. What happens in Maintenance Phase?	Patch management, logging, monitoring, penetration testing, and security training.
11. What is Risk Management in SDLC?	Identifying, assessing, and mitigating security risks throughout the lifecycle.
12. What is Threat Modeling?	A structured way to identify potential security threats and plan mitigations (e.g., STRIDE model).
13. What is Role-Based Access Control (RBAC)?	Granting permissions based on user roles to limit unauthorized access.
14. What is Secure Coding?	Writing code that prevents vulnerabilities like SQL injection, XSS, and buffer overflow.
15. What are common coding errors?	Buffer overflow, injection flaws, XSS, broken authentication, race conditions.
16. What are good development practices for security?	Code reviews, secure frameworks, version control, input validation, and CI/CD with testing.

17. What is the difference between SDLC and Secure SDLC?	SDLC focuses on functionality; Secure SDLC integrates security at all stages.
18. What tools are used in Secure SDLC?	Static analyzers (Bandit, Fortify), DAST tools (ZAP, Burp), CI/CD scanners.
19. What is Security Testing?	Testing that finds vulnerabilities (via penetration testing or vulnerability scanning).
20. What is a Web Application Firewall (WAF)?	A security filter that monitors and blocks malicious HTTP traffic.
21. What is CI/CD Security Integration?	Automating security checks during build and deployment.
22. What is AES Encryption?	Advanced Encryption Standard — symmetric encryption algorithm used for data protection.
23. What is the significance of Secure SDLC?	Ensures software is reliable, compliant, and resistant to modern threats.

🧩 Conclusion:

This experiment shows how integrating security at every SDLC phase ensures confidentiality, integrity, and availability of user data, especially for web systems handling sensitive information.