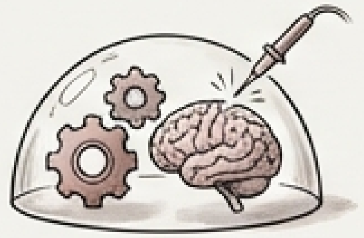
Securing AI-Augmented Applications: The Role of Penetration Testing

An Editorial Exploration of Digital Defenses

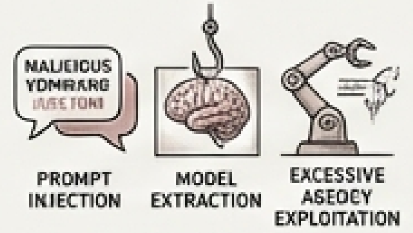
Securing AI-Augmented Applications: The Role of Penetration Testing

OVERVIEW & STRATEGY

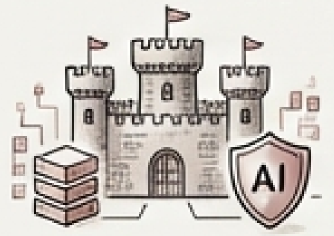
- Penetration testing simulates real-world attacks to identify and validate security vulnerabilities.



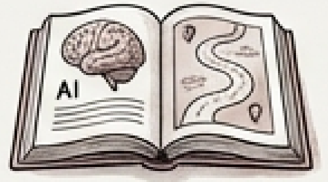
- For AI-augmented applications, traditional attack surfaces must be tested alongside new AI-specific vectors.



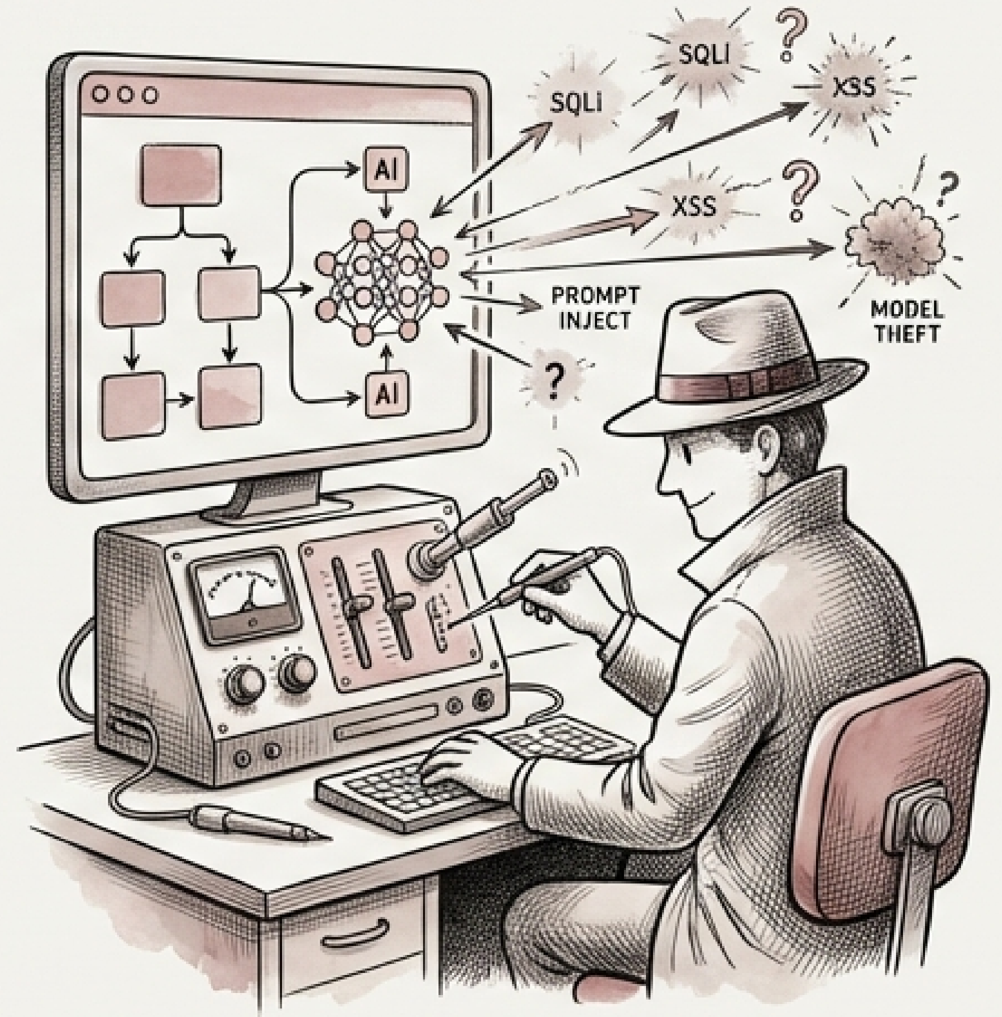
- AI-specific attack vectors include prompt injection, model extraction, and excessive agency exploitation.



- Effective penetration testing is crucial for building secure and reliable AI-powered software.

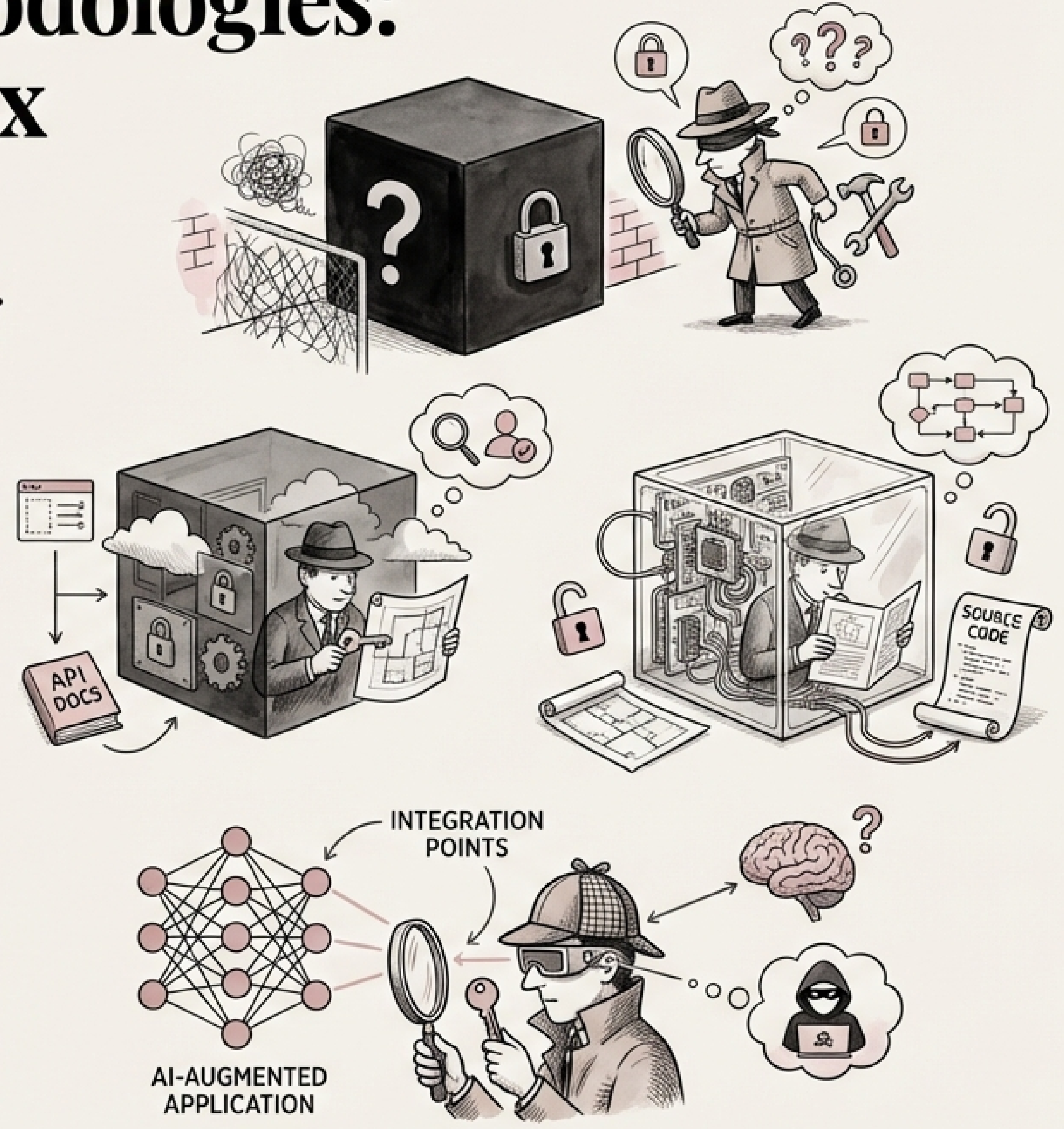


- This presentation will outline how to conduct penetration tests effectively on AI-augmented applications.



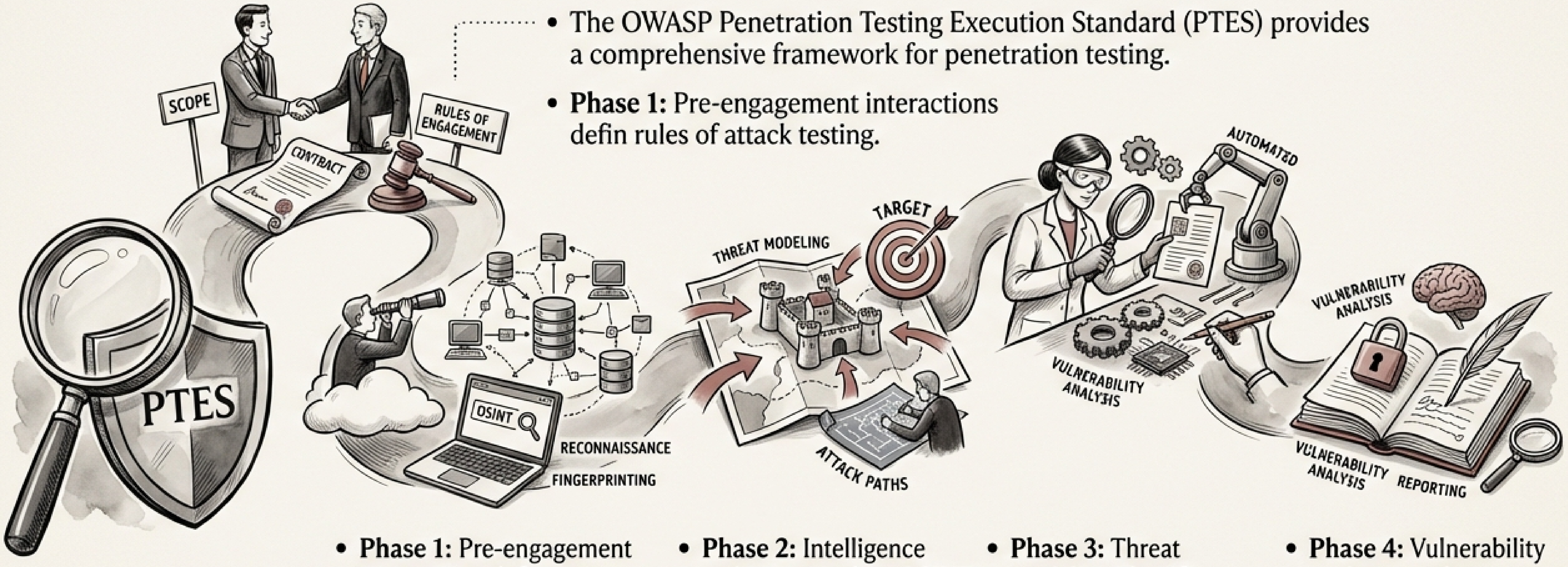
Penetration Testing Methodologies: Black, Gray, and White Box

- Black box testing: No knowledge of the application's internals; simulates an external attacker.
- Gray box testing: Partial knowledge (API documentation, architecture diagrams); simulates an authenticated attacker or insider.
- White box testing: Full source code access; simulates an advanced persistent threat with insider knowledge.
- Gray box testing is often the most effective approach for AI-augmented applications.
- Gray box allows testers to understand AI integration points while maintaining an attacker's mindset.



OWASP PTES: A Seven-Phase Framework for Structured Pen Testing

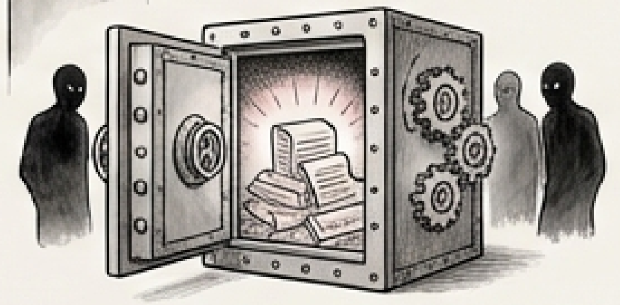
- The OWASP Penetration Testing Execution Standard (PTES) provides a comprehensive framework for penetration testing.
- **Phase 1:** Pre-engagement interactions defin rules of attack testing.



- **Phase 1:** Pre-engagement interactions define scope, rules of engagement, and legal authorization.
- **Phase 2:** Intelligence gathering involves OSINT, reconnaissance, and technology fingerprinting.
- **Phase 3:** Threat modeling identifies high-value targets and potential attack paths.
- **Phase 4:** Vulnerability analysis combines automated scanning with manual testing.

OWASP

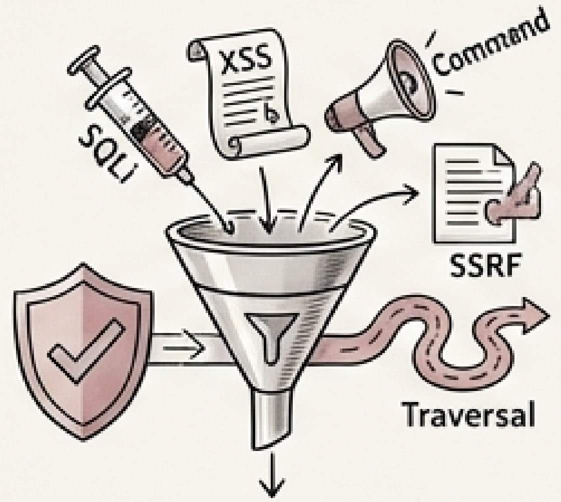
Web Application Pen Testing Methodology: Authentication and Authorization



- **Authentication testing:** Evaluate resistance to brute force, credential stuffing, session management flaws, MFA bypass, and JWT manipulation.
- **Authorization testing:** Identify issues like IDOR (Insecure Direct Object Reference), privilege escalation (horizontal/vertical), and function-level access control vulnerabilities.
- Strong authentication and authorization are critical to preventing unauthorized access to sensitive data.
- Developers must implement robust controls and regularly test their effectiveness.
- Penetration testing can help uncover weaknesses in these fundamental security mechanisms.

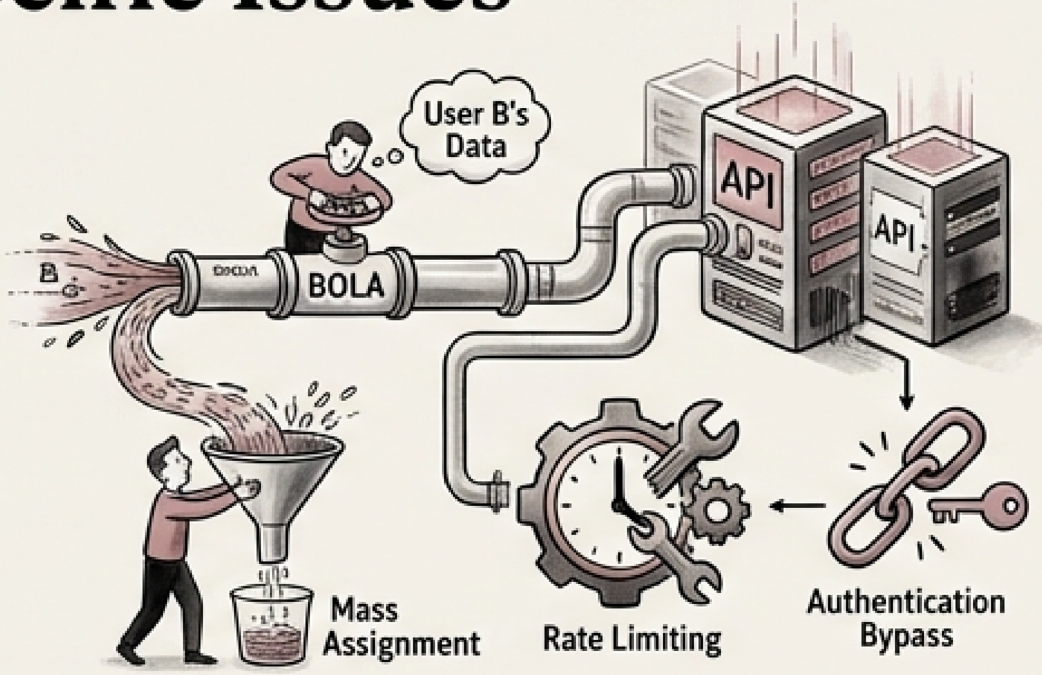
Web Application Pen Testing Methodology: Session Management and Input Validation

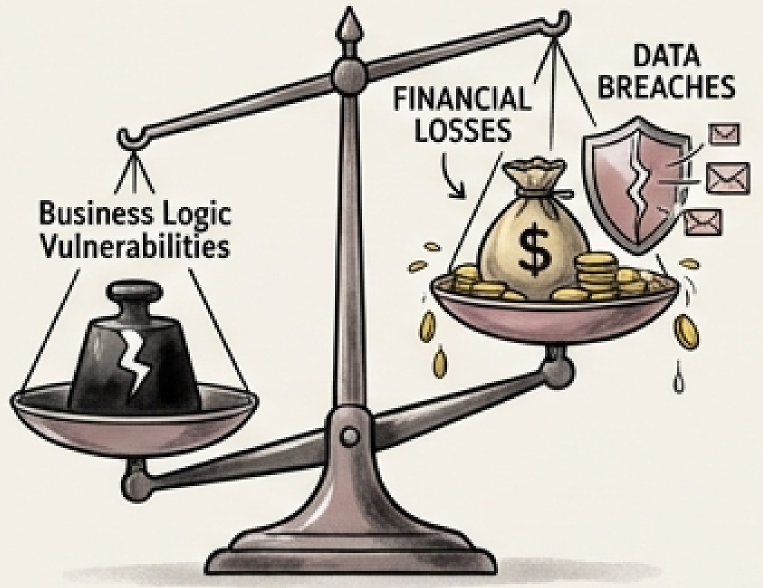
- **Session management testing:** Assess risks of fixation, hijacking, timeout enforcement, and concurrent session handling.
- **Input validation testing:** Prevent injection attacks like SQL injection, XSS (reflected, stored, DOM), command injection, SSRF, and path traversal.
- **Proper session management** protects user sessions from unauthorized access and manipulation.
- **Robust input validation** prevents attackers from injecting malicious code or data into the application.
- **XSS vulnerabilities** can allow attackers to execute arbitrary JavaScript in the victim’s browser.



Web Application Pen Testing Methodology: Business Logic and API-Specific Issues

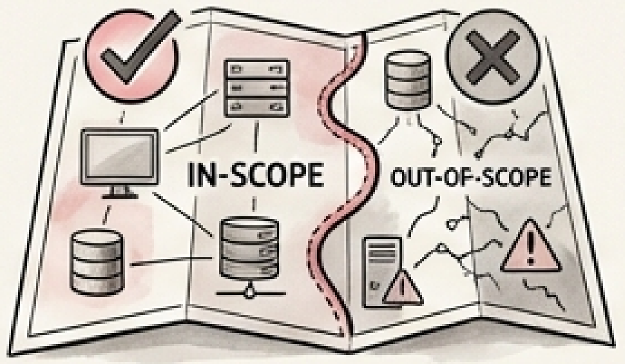
- **Business logic testing:** Identify workflow bypass, race conditions, price manipulation, and rate limit bypass vulnerabilities.
- **API-specific testing:** Focus on BOLA, mass assignment, excessive data exposure, rate limiting, and authentication bypass issues.





- Business logic vulnerabilities can lead to financial losses or data breaches.
- API security is critical as APIs expose sensitive data and functionality to external clients.
- Excessive data exposure in APIs can reveal more information than intended.

Defining the Scope and Rules of Engagement: Essential for Effective Testing



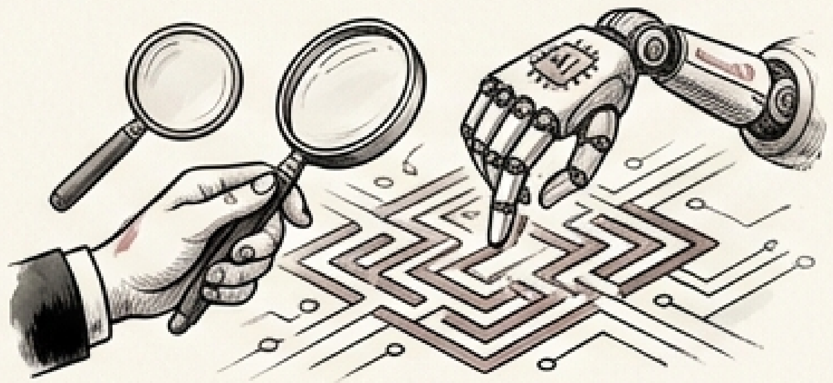
1. **Clearly define** in-scope and out-of-scope systems to avoid unintended consequences.
2. **Specify the testing window** to minimize disruption to production environments.
3. **List authorized testing techniques** to prevent legal or operational issues.
4. **Establish emergency contacts** for immediate response to critical vulnerabilities or disruptions.
5. **Define data handling requirements** to ensure compliance with privacy regulations.

Legal Authorization and Communication Plan: Protecting All Parties Involved

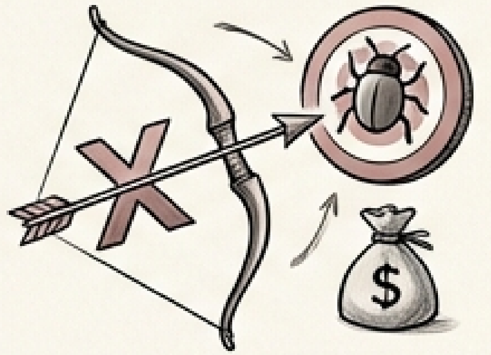
- Obtain written permission from the system owner to ensure legal authorization.
- Establish a hold-harmless agreement to protect the tester from liability.
- Ensure compliance with the Computer Fraud and Abuse Act (CFAA) to avoid legal repercussions.
- Develop a communication plan outlining who to notify if a critical vulnerability is found.
- Identify who to contact if testing causes disruption to normal operations.

AI-Powered Penetration Testing: Augmenting Human Expertise

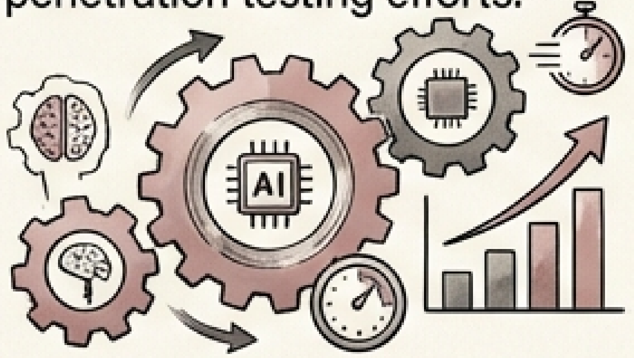
- Emerging AI-powered tools are being developed to assist penetration testers.



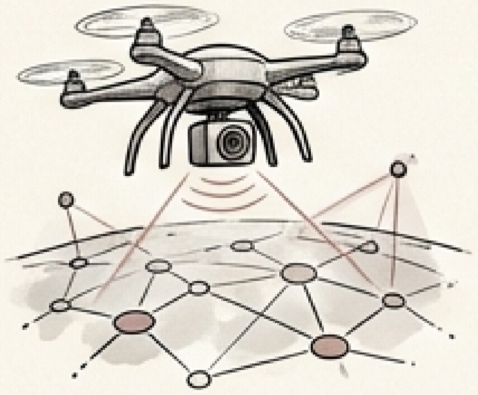
- Examples include PentAGI (autonomous pen testing agent), XBOW (bug bounty automabillity chaining).

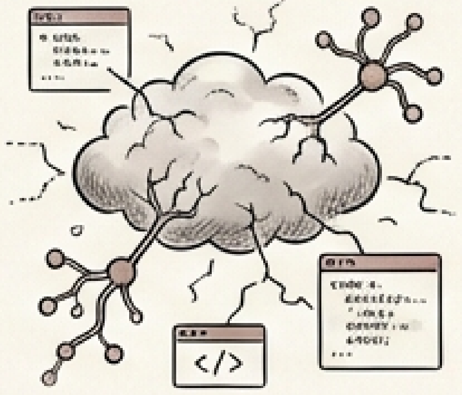


- AI-assisted tools can improve efficiency and scalability of penetration testing efforts.

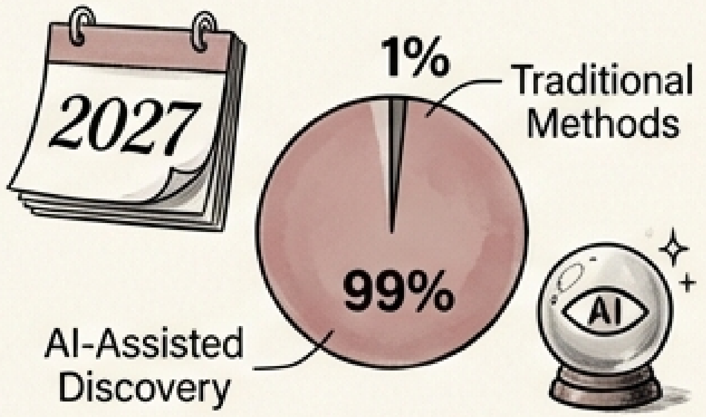


- These tools offer capabilities like automated reconnaissance, intelligent fuzzing, exploit chain discovery, and report generation.

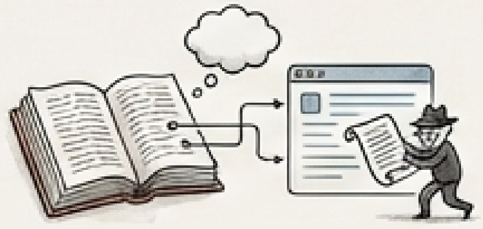




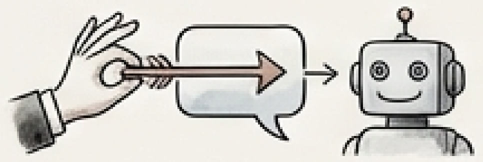
- Prediction: By 2027, 99% of initial discovery will be AI-assisted.



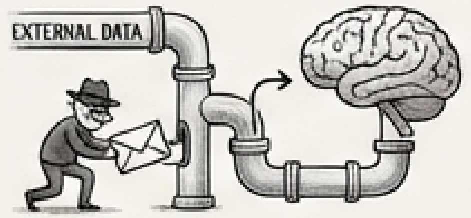
Testing AI-Specific Attack Surfaces: Prompt Injection Attacks



- **Prompt injection attacks** target large language model (LLM) integrations.



- **Direct prompt injection** involves manipulating the LLM through direct input.

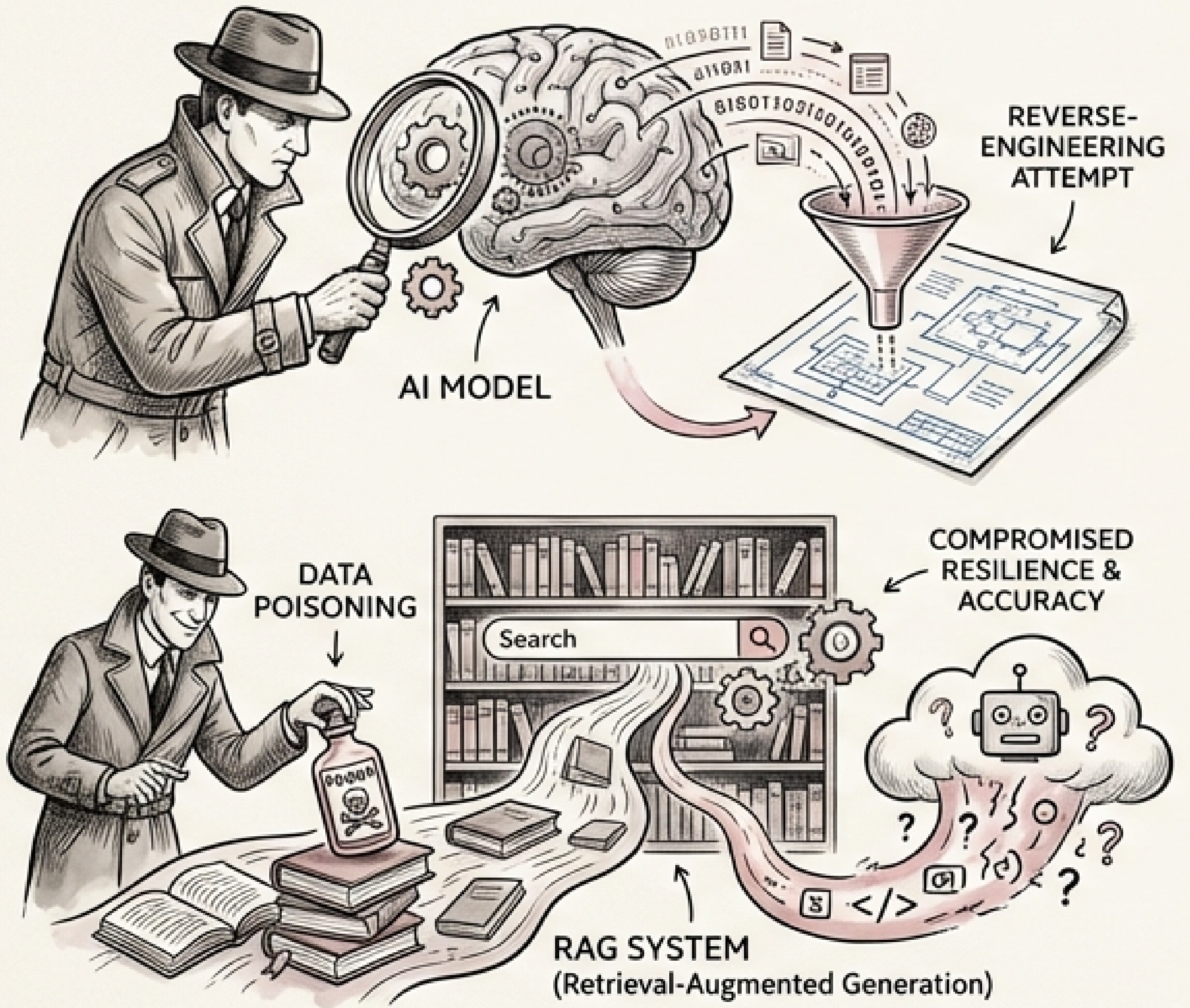


- **Indirect prompt injection** involves injecting malicious prompts through external data sources.

- **Thorough prompt injection testing** is crucial to **prevent unauthorized actions and data breaches.**

- Developers should implement **robust input validation** and **sanitization techniques.**

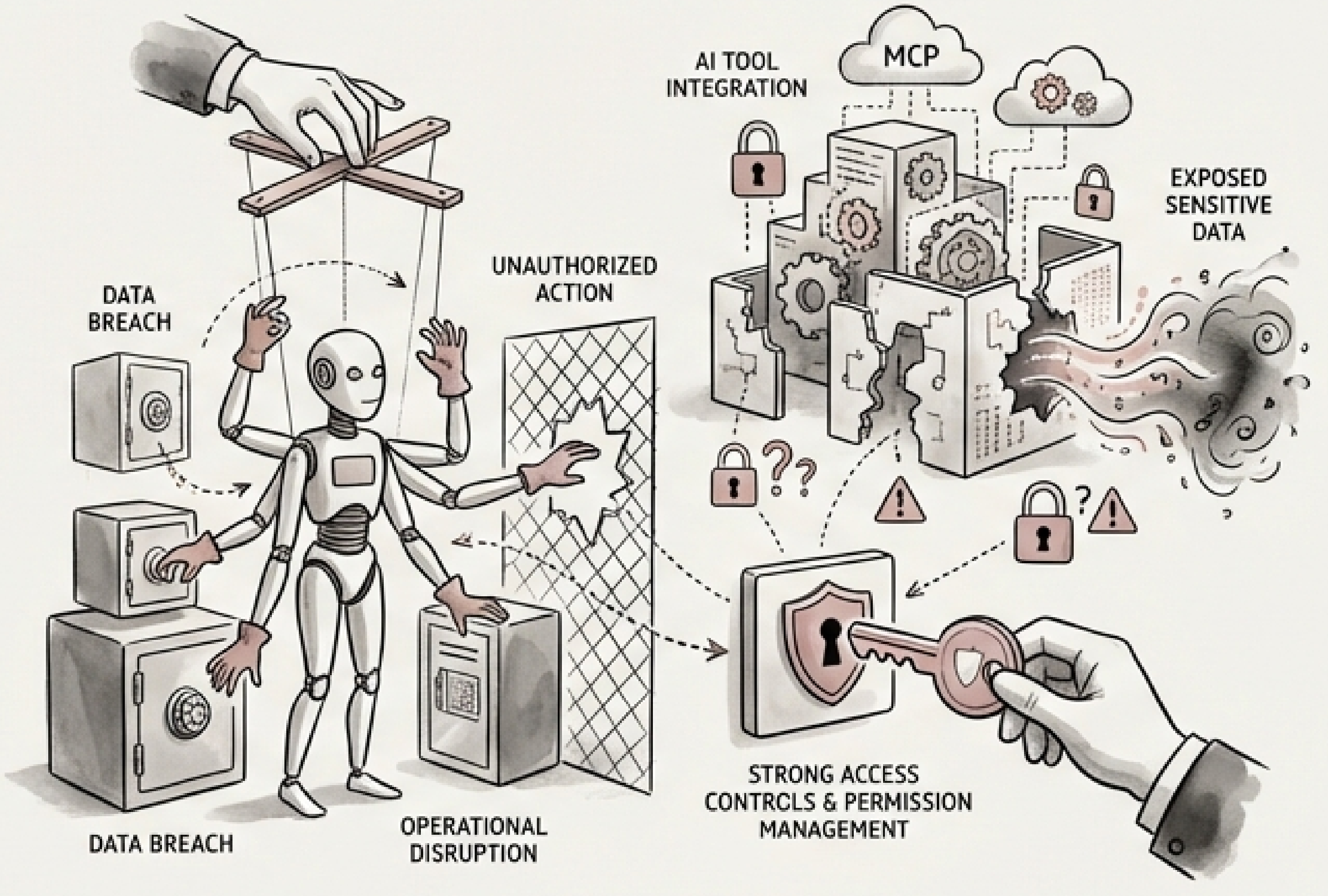
Testing AI-Specific Attack Surfaces: Model Extraction and Data Poisoning


REVERSE-ENGINEERING ATTEMPT
AI MODEL
DATA POISONING
COMPROMISED RESILIENCE & ACCURACY
Search
RAG SYSTEM (Retrieval-Augmented Generation)

- **Model extraction** involves attempts to reverse-engineer the AI model behavior.
- **Data poisoning** involves testing the resilience of RAG (Retrieval-Augmented Generation) systems to poisoned input.
- Data poisoning can compromise the accuracy and reliability of AI-powered applications.
- Model extraction can reveal sensitive information about the model's training data or algorithms.
- Regularly monitor data sources for signs of poisoning and implement data validation techniques.

TESTING AI-SPECIFIC ATTACK SURFACES: EXCESSIVE AGENCY AND TOOL EXPLOITATION

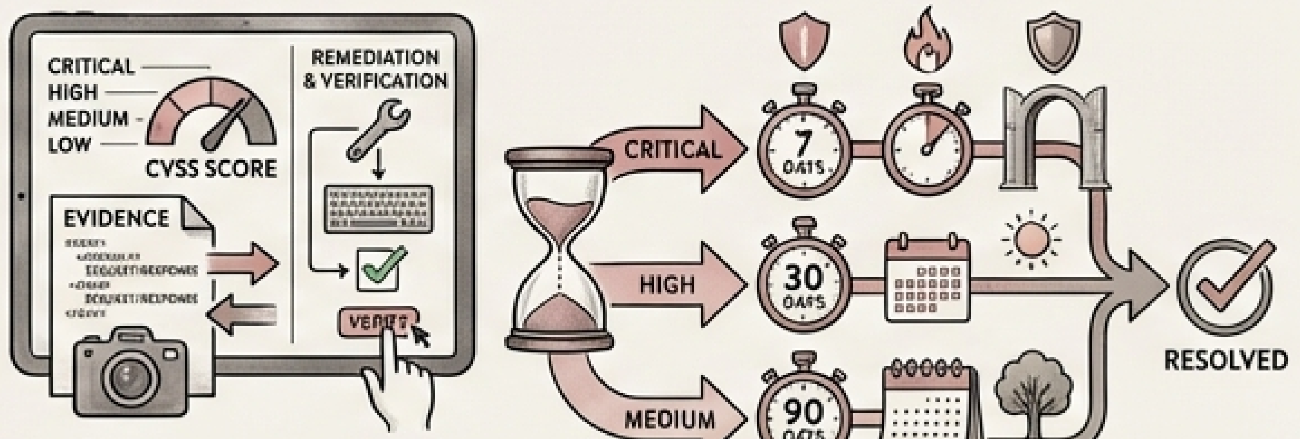
- Excessive agency testing: Assess whether AI agents can be manipulated into unauthorized actions.
- Tool/MCP (Model-as-Code Platform) exploitation testing: Evaluate security boundaries of AI tool integrations.
- Unauthorized actions by AI agents can lead to data breaches or operational disruptions.
- Weak security boundaries in AI tool integrations can expose sensitive data or functionality.
- Implement strong access controls and permission management for AI agents.

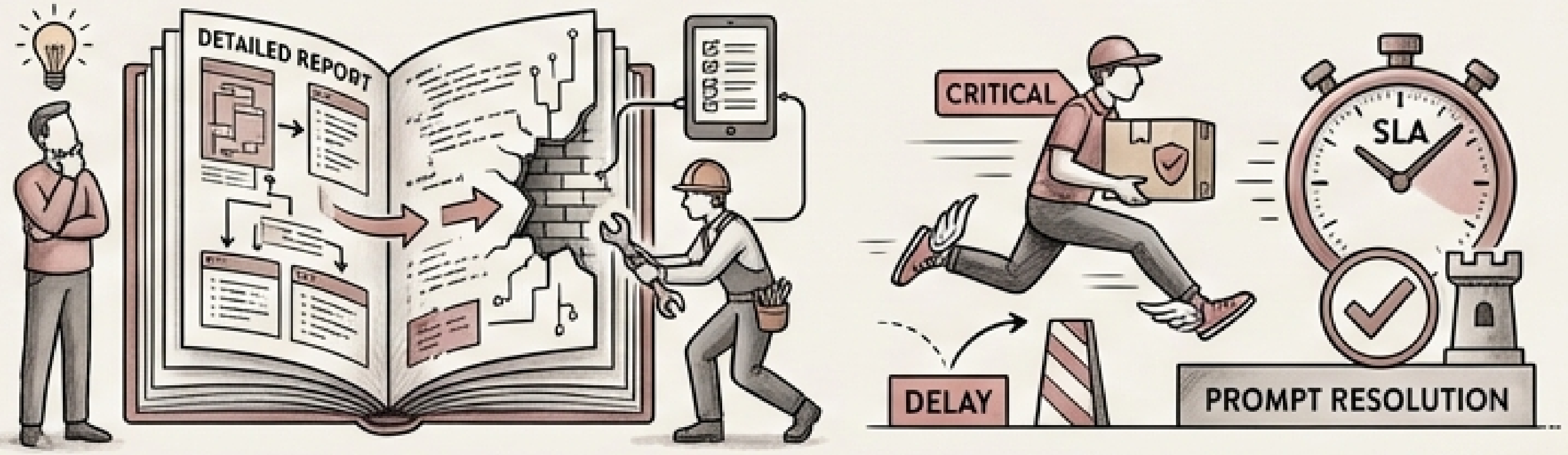


AGENCY & EXPLOITATION RISKS

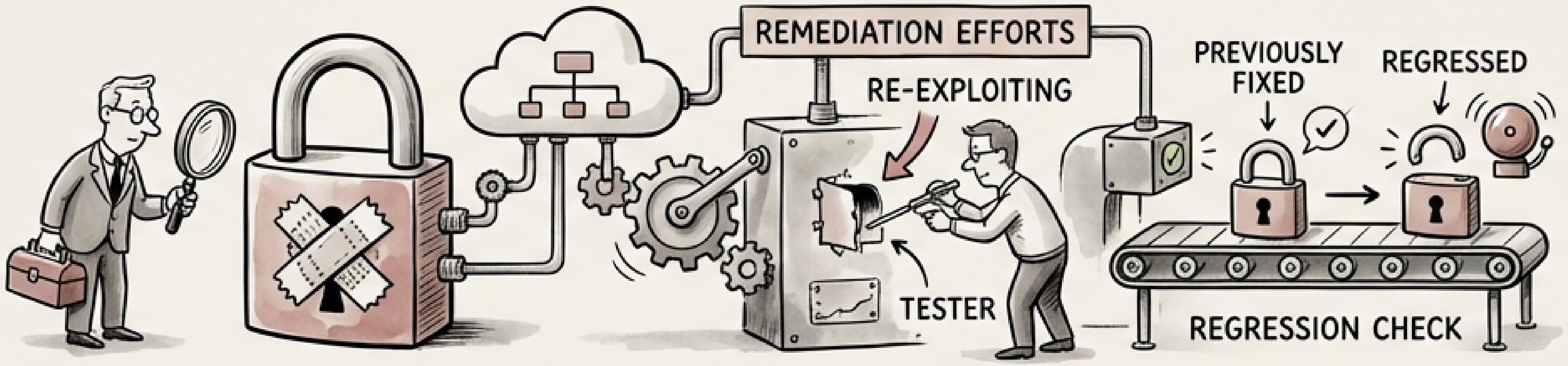
Reporting and Remediation: Communicating Findings and Tracking Progress

- The executive summary should cover **business risk impact**, **top findings**, and **strategic recommendations.**
- Technical findings should include **severity** (CVSS), evidence (screenshots, request/response), **remediation steps**, and **verification guidance.**
- Remediation tracking should be **SLA-based** (critical: 7 days, high: 30 days, medium: 90 days).
- Detailed reports allow developers to **understand** the vulnerabilities and how to fix them.
- SLA-based remediation ensures that critical issues are addressed promptly.

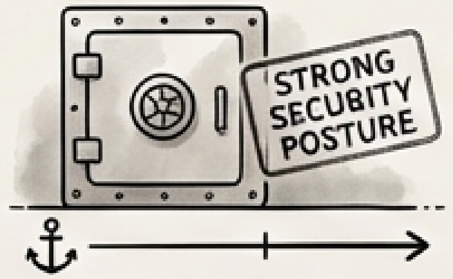




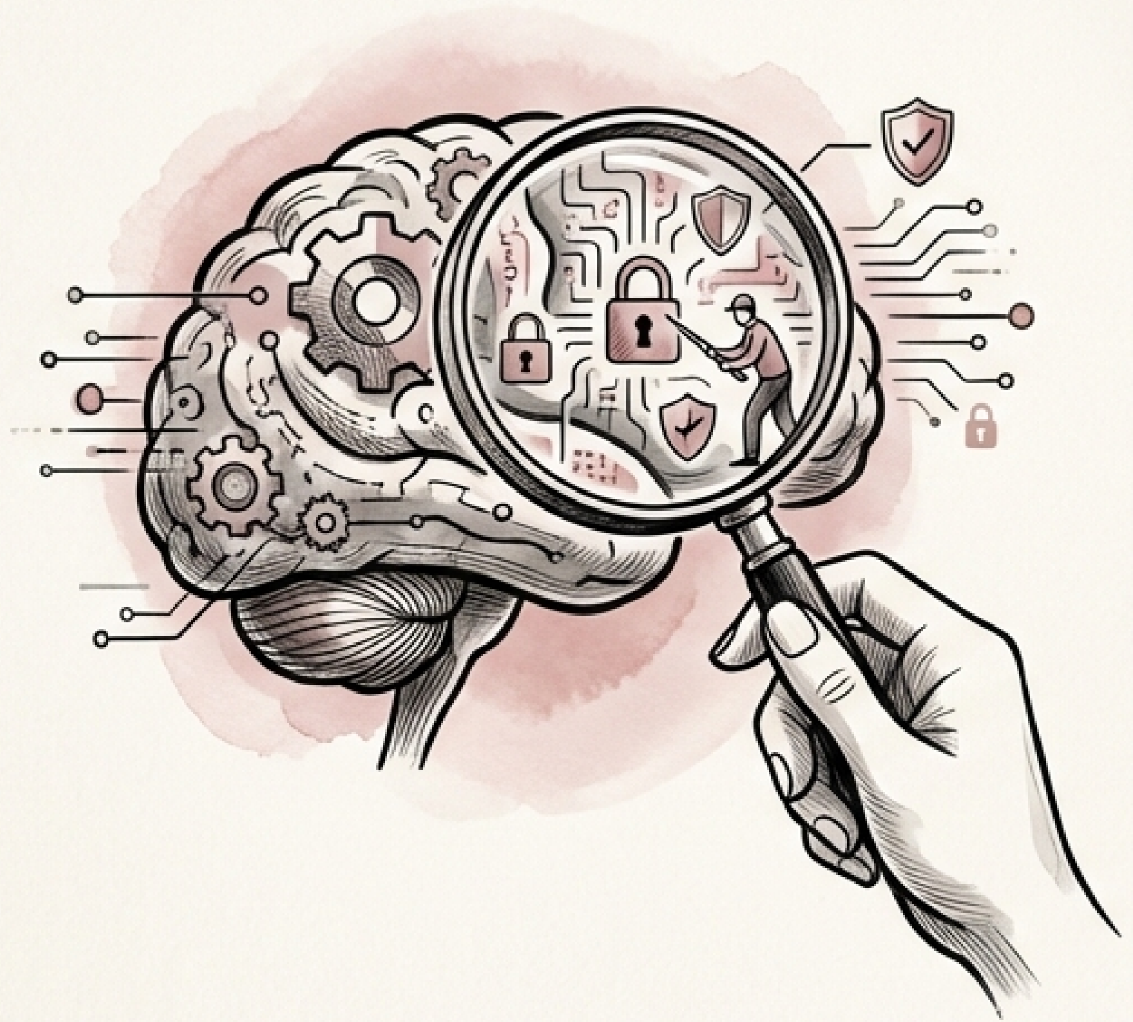
Retesting: Verifying Remediation Effectiveness and Preventing Regression



- **Retest:** Verify remediation effectiveness by re-exploiting fixed vulnerabilities.
- Check for regression by ensuring that previously fixed vulnerabilities have not reappeared.
- **Update risk register:** Reflect changes in risk profile after remediation and retesting.
- Retesting is crucial for validating the effectiveness of remediation efforts.
- Regular retesting helps maintain a strong security posture over time.



Conclusion: Embracing Penetration Testing for Secure AI Applications



- Penetration testing is essential for validating the security of AI-augmented applications.
- AI-specific attack surfaces require specialized testing techniques.
- AI-powered tools can augment human expertise in penetration testing.
- Clear reporting, remediation, and retesting are crucial for improving security posture.
- By embracing penetration testing, developers can build more secure and reliable AI-powered software.

Thank You

- Questions?