PACKETLABS

Ready for more than a VA scan? ®

Methodology

Penetration Testing, Application Security Testing

February 9, 2021

Methodology

Penetration Testing

Our Penetration Testing methodology is derived from the SANS Pentest Methodology, the MITRE ATT&CK framework for enterprises, and NIST SP800-115 to ensure compliance with most regulatory requirements. Our methodology is comprehensive and has been broken up based on which areas can be tested with automation and those which require extensive manual testing.

Phase	Tasks Completed	Manual	Automated
Recon & Mapping	Conduct search engine discovery and reconnaissance for information leakage	✓	-
	Harvest emails, usernames, credentials, documents and various information from OSINT sources	✓	-
	Review email headers	✓	-
	Enumerate exposed applications and servers of the target organization	✓	-
	Review current and historical WHOIS metadata for information leakage	✓	-
	Identify application and external perimeter entry points	✓	-
	Identify host-based and network-based defense technologies	✓	-
	Reference IMINT sources for an organization's weak entry points	✓	-
	Document various vectors and avenues for initial access	✓	-
Discovery	**Initial Discovery**		
	Comprehensive port scanning, fingerprinting and mapping of services and applications	✓	✓
	Identify password length and lockout policy	✓	✓
	Identify centralized management authentication servers	✓	✓
	Conduct passive network traffic analysis	✓	✓
	Conduct DNS based enumeration	✓	✓
	Enumerate infrastructure and admin interfaces	✓	✓
Vulnerability Assessment	**Configuration and Deploy Management Testing**		
	Utilization of automated scanning tools & technologies to identify publicly known operating system, application and service vulnerabilities	✓	✓
	Manual validation of findings, removal of false-positives, and low-confidence findings where applicable	✓	-
	Test for unencrypted management interfaces and services	✓	✓
	Test for SSL/TLS configuration and certificates	✓	✓
	Test for egress filtering controls	✓	-
	Test for server and endpoint hardening	✓	-
	Test for stale network address configurations	✓	-
	Test for anonymous access	✓	✓
	Test for insecure configuration of various TCP and UDP services	✓	✓

Phase	Tasks Completed	Manual	Automated
	Test for servers and endpoints without endpoint protection	✓	-
	Patch Level Testing		
	Identify false positives through internal and commercial tooling	✓	✓
	Identify and prioritize vulnerable targets for initial access	✓	✓
	Test for missing security patches and EOL software	✓	✓
	Test for common and known vulnerabilities in weak applications	✓	✓
Exploitation	**Vulnerabilities and Misconfigurations**		
	Exploitation of identified vulnerabilities and misconfigurations using public exploit code, or custom exploits as applicable	✓	-
	Demonstrate impacts of identified vulnerabilities on the environment, data, services or applications	✓	-
	Credential Access		
	Conduct password spraying and brute force attacks on harvested usernames/emails;	✓	-
	Test for default credentials on operating systems, services and applications	✓	✓
	Harvest credentials and hashes on the network	✓	✓
	Relay network level hashes to unhardened targets	✓	-
	Test for credential reuses	✓	-
	Test for Kerberos authentication flaws	✓	-
	Test for Kerberos misconfigurations	✓	-
	Test for insecure credential storage (browser storage, plain-text documents, email, file-share, etc.)	✓	-
Post-Exploitation	**Permissions Testing and Privilege Escalation**		
	Identify and exploit endpoint-based privilege escalation vulnerabilities or misconfigurations on compromised systems	✓	✓
	Identify and exploit environment privilege escalation vulnerabilities or misconfigurations	✓	✓
	Conduct BloodHound enumeration and map privilege escalation paths for AD environments	✓	✓
	Test for Active Directory misconfigurations and vulnerabilities to elevate to a domain/enterprise administrator	✓	✓
	Test for user group segregation	✓	✓
	Test for file system permissions weakness	✓	✓
	Test for overly permissive share access	✓	✓
	Attempt to exfiltrate data out of restricted environment(s) if applicable	✓	✓
	Attempt to gain unauthorized access to mission critical application/systems with established access	✓	✓
	Service misconfigurations	✓	✓
	Lateral Movement		
	Pivot to from compromised systems to other internal systems	✓	-
	Test for reoccurring usage of credentials	✓	-

Phase	Tasks Completed	Manual	Automated
	Test for lateral movement on exposed network protocols	✓	✓
	Test for lateral movement with common system binaries	✓	✓
	Conduct exploitation of vulnerable remote services	✓	-
	Encryption keys, i.e., SSH keys	✓	-
	Test for network segregation	✓	-
	Test for common Windows and Active Directory Lateral Movement techniques; • Cached credentials • Pass the hash attacks • Pass the ticket attacks • Windows services (RDP, PSExec, task scheduler, etc.)	✓	-
	Defense Evasion		
	Attempt to bypass application whitelisting	✓	✓
	Attempt to bypass endpoint security solution	✓	✓
	Attempt to circumvent network access controls (Firewall, IDS, IPS, WAF, etc.)	✓	✓
	Password Audit		
	Crack hashes using specialized computing hardware to reveal plain-text passwords	✓	✓
	Usage of optimized wordlists and masks for target organization and individual targets	✓	-
	Conduct a breach analysis of cracked passwords	✓	✓
	Conduct a statistical analysis and review of all plain-text credentials obtained	✓	-
	Customized recommendations for password policy improvements	✓	-
Reporting	A draft detailed report outlining findings coupled with control recommendations including an executive summary outlining the overall state of the application.	✓	✓
	Document steps to reproduce findings to ensure application developers can validate remediation efforts prior to retesting.	✓	-
	Conduct root cause analysis of findings outlining common themes observed with recommendations to improve security within the environment.	✓	-

Application Methodology

Our Application Security Testing methodology is derived from the OWASP Top 10:2017 and has been enhanced with current threats and our overall experience in the industry. Our methodology is comprehensive and has been broken up based on which areas can be tested with automation and those which require extensive manual testing.

Phase	Tasks Completed	Manual	Automated
Recon & Mapping	Conduct search engine discovery and reconnaissance for information leakage	✓	✓
	Fingerprint web server	✓	✓
	Review web server metafiles for information leakage	✓	✓
	Enumerate applications on web servers	✓	✓
	Review webpage comments and metadata for information leakage	✓	✓
	Identify application entry points	✓	✓
	Identify technologies (e.g., web applications, frameworks or CMS platforms) used	✓	✓
	Map visible content and perform automated spidering of referenced content	✓	✓
	Test for debug parameters	✓	✓
	Discover hidden & default content	✓	✓
Discovery	**Configuration and Deploy Management Testing**		
	Test network/infrastructure configuration	✓	✓
	Test application platform configuration	✓	✓
	Test file extensions handling for sensitive information	✓	✓
	Analyze backup and unreferenced files for sensitive information	✓	✓
	Enumerate Infrastructure and application admin interfaces	✓	✓
	Test HTTP methods	✓	✓
	Test HTTP strict transport security	✓	✓
	Test RIA cross-domain policy	✓	✓
	Test for web server vulnerabilities	✓	✓
	Testing for vulnerabilities in third-party applications (e.g. WordPress, Joomla, Drupal, SharePoint)	✓	✓
	Identity Management Testing		
	Test role definitions	✓	-
	Test user registration process	✓	-
	Test account provisioning process	✓	-
	Testing for account enumeration and guessable user account	✓	-
	Testing for weak or unenforced username policy	✓	-
	Test permissions of guest/training accounts	✓	-
	Test account suspension/resumption Process	✓	-
	Authentication Testing		
	Testing for credentials transported over an encrypted channel	✓	✓
	Testing for default credentials	✓	✓
	Testing for a weak lockout mechanism	✓	✓

Phase	Tasks Completed	Manual	Automated
	Testing for bypassing authentication schema	✓	✓
	Test remember password functionality	✓	✓
	Testing for browser cache weakness	✓	✓
	Testing for weak password policy	✓	✓
	Testing for weak security question/answer	✓	✓
	Testing for weak password change or reset functionalities	✓	✓
	Testing for weaker authentication in alternative channel	✓	✓
	Authorization Testing		
	Testing directory traversal/file include	✓	-
	Testing for bypassing authorization schema	✓	-
	Testing for privilege escalation	✓	-
	Testing for insecure direct object references	✓	-
	Session Management Testing		
	Testing for bypassing session management schema	✓	-
	Analyze cookies attributes (e.g., HttpOnly, Secure flags and scope)	✓	-
	Testing for session fixation	✓	-
	Testing for cross-site request forgery	✓	-
	Testing for logout functionality	✓	-
	Test session timeout	✓	-
	Testing for session puzzling	✓	-
	Persistent cookies	✓	-
	Test tokens for predictability	✓	-
	Check for insecure transmission of session tokens	✓	-
	Input Validation Testing		
	Fuzz all input parameters	✓	✓
	Testing for reflected cross-site scripting	✓	✓
	Testing for stored cross-site scripting	✓	✓
	Testing for HTTP verb tampering	✓	✓
	Testing for HTTP parameter pollution	✓	✓
	Testing for HTTP splitting/smuggling	✓	✓
	Testing for SQL injection (Oracle, MySQL, MsSQL, PostgreSQL, Microsoft Access, NoSQL)	✓	✓
	Testing for LDAP injection	✓	✓
	Testing for ORM injection	✓	✓
	Testing for XML injection	✓	✓
	Testing for SSI injection	✓	✓
	Testing for XPath injection	✓	✓
	Testing for IMAP/SMTP injection	✓	✓
	Testing for code injection	✓	✓
	Testing for local file inclusion	✓	✓
	Testing for remote file inclusion	✓	✓
	Testing for command injection	✓	✓

Phase	Tasks Completed	Manual	Automated
	Testing for native software flaws (buffer overflow, integer bugs, format strings)	✓	✓
	Testing for incubated vulnerabilities	✓	✓
	Testing for open redirection	✓	✓
	Testing for SOAP injection	✓	✓
	Error Handling		
	Analysis of error codes	✓	✓
	Analysis of stack traces	✓	✓
	Cryptography		
	Testing for weak SSL/TLS ciphers, insufficient transport layer protection	✓	✓
	Testing for padding oracle	✓	✓
	Testing for sensitive information sent via unencrypted channels	✓	✓
	Testing for CBC bit flipping	✓	✓
	Testing for hash length extension	✓	✓
	Business Logic Testing		
	Identify the logic attack surface	✓	-
	Test business logic data validation	✓	-
	Test the ability to forge requests	✓	-
	Test integrity checks	✓	-
	Test for process timing (race conditions, TOCTOU)	✓	-
	Testing for the circumvention of workflows	✓	-
	Test defenses against application misuse	✓	-
	Test upload of unexpected file types	✓	-
	Test upload of malicious files	✓	-
	Analyze SSL responses for caching of sensitive content	✓	-
	Analyze content for sensitive data in URL parameters	✓	-
	Testing for reliance on client-side input validation	✓	-
	Testing of trust boundaries	✓	-
	Client-Side Testing		
	Testing for DOM-based cross-site scripting	✓	✓
	Testing for JavaScript execution	✓	✓
	Testing for HTML injection	✓	✓
	Testing for client-side open redirection	✓	✓
	Testing for CSS injection	✓	✓
	Testing for client-side resource manipulation	✓	✓
	Test cross-origin resource sharing	✓	✓
	Testing for cross-site flashing	✓	✓
	Testing for clickjacking	✓	✓
	Testing WebSockets	✓	✓
	Test web messaging	✓	✓
	Test local storage	✓	✓

Phase	Tasks Completed	Manual	Automated
	Testing of thick-client components (Java, ActiveX, Flash)	✓	✓
	Audit: WordPress		
	Test for outdated plugins	✓	-
	Test for XMLRPC exposure	✓	-
	Test for exposed admin portal	✓	-
	Audit: JavaScript		
	Test for overly permissive Content Security Policy (CSP)	✓	-
	Test for subresource integrity checks	✓	-
	Testing for linking to third-party Code	✓	-
	Testing for advertisement and analytics on critical flows	✓	-
	Testing for critical flows isolation	✓	-
	Leverage findings from previous phases in order to expand foothold in the environment.	✓	-
Exploitation	Execute a number of exploits focusing on: • bypass attacks • injection attacks • session attacks	✓	-
	Attempt to escalate privileges and/or gain unauthorized access	✓	-
	Attempt to pivot from compromised systems to other internal systems.	✓	-
Reporting	A draft detailed report outlining findings coupled with control recommendations including an executive summary outlining the overall state of the application.	✓	✓
	Document steps to reproduce findings to ensure application developers can validate remediation efforts prior to retesting.	✓	-
	Conduct root cause analysis of findings outlining common themes observed with recommendations to improve security within the environment.	✓	-