414 Hacking Exposed Web Applications

his checklist summarizes the many recommendations and countermeasures made   
throughout this book. Although we have not reiterated every detail relevant to

each checklist item here, we hope they serve as discrete reminders of the many

T

security best practices that should be considered when designing and operating any web application.

|  |  |
| --- | --- |
| Item | Check |
| Network |  |
| Perimeter firewall, screening router, or other filtering device established  between web application and untrusted networks. Try to avoid using  filtering devices that do not support stateful packet inspection (SPI). |  |
| Firewall/router configured to allow only necessary traffic inbound to web  application (typically only HTTP and/or SSL). |  |
| Firewall/router configured to permit only necessary traffic outbound from  the web application (typically TCP SYN packets are dropped to prevent  servers from initiating outbound connections). |  |
| Appropriate denial-of-service countermeasures enabled on firewall/  gateway (for example, Cisco rate limit command). |  |
| Load balancers configured not to disclose information about internal  networks. |  |
| A Network Intrusion Detection System (NIDS) may be optionally  implemented to detect common TCP/IP attacks; appropriate log review  policies and resources should be made available if NIDS is implemented. |  |
| Disable Telnet on routers and other network devices that have it enabled  for remote administration. Use SSH instead. |  |
| Perform regular password audits of any services that may be used for  remote administration (e.g., SSH) and also limit the remote IP addresses  that can be used to access these services. |  |
| Network vulnerability scans conducted regularly to ensure no network or  system-level vulnerabilities exist. |  |
| Manual penetration tests conducted by a third party at least twice a year  or every time significant changes are made to the network infrastructure to  identify more complex vulnerabilities. |  |
| Web Server |  |
| Latest vendor software patches applied. |  |
| Servers configured not to disclose information about the server software  and plug-ins/modules installed (for example, banner information changed). |  |
| Servers configured not to allow directory listing and parent paths. |  |
| Servers configured to disallow reverse proxy. |  |

Appendix A: Web Application Security Checklist

415

|  |  |
| --- | --- |
| Item | Check |
| Unnecessary network services disabled on all servers. |  |
| OS and server vendor-specific security configurations implemented where  appropriate. |  |
| Unnecessary users or groups (e.g., Guest) disabled or removed. |  |
| Operating system auditing enabled, as well as web server logging in  W3C format. |  |
| Unnecessary HTTP modules or extensions disabled on all servers (e.g.,  unused IIS ISAPI DLLs unmapped and Apache mods uninstalled). |  |
| Sample web content/applications removed from all servers. |  |
| Appropriate authentication mechanisms configured for relevant  directories. |  |
| Secure Sockets Layer (SSL) is deployed to protect traffic that may be  vulnerable to eavesdropping (e.g., HTTP Basic Authentication). Require  128-bit encryption and do not allow downgrades to weaker export-grade  encryption for sensitive transactions. Also disable support for SSLv2; use  only SSLv3. |  |
| Virtual roots containing web content deployed on a separate, dedicated  disk drive/volume (without administrative utilities). |  |
| Disable directory listing and parent paths. |  |
| Customize error pages to avoid information leaks. |  |
| Account running HTTP service should be low-privileged. |  |
| Appropriate Access Control Lists (ACLs) set for web directories and files. |  |
| WebDAV functionality disabled or removed if not used; otherwise,  WebDAV should be heavily restricted. |  |
| Web Publisher functionality (for Netscape/iPlanet products) disabled. |  |
| Web server security modules deployed where appropriate (e.g., IIS UrlScan  or Apache ModSecurity). |  |
| Servers scanned by vulnerability scanner for remotely exploitable  vulnerabilities; issues addressed. |  |
| A Host Intrusion Detection System (HIDS) may be optionally implemented  to detect common applications; appropriate log review policies and  resources should be made available if HIDS is implemented. |  |
| Database Server |  |
| Database software installed to run with least privilege (e.g., in the context  of a low-privileged local or domain account on Microsoft SQL Servers). |  |
| Database software updated to the latest version with appropriate vendor  patches. |  |

416 Hacking Exposed Web Applications

|  |  |
| --- | --- |
| Item | Check |
| Sample accounts and databases removed from the server. |  |
| Appropriate IP packet filtering enabled to restrict traffic between web  servers and database servers (e.g., SPI Firewall, router, or IPSec filters on  Windows 2000 and above). If possible, locate database servers on their own  network segment with a dedicated SPI Firewall and do not allow outbound  traffic from that segment. |  |
| Appropriate authentication is employed between web servers and the  database (e.g., for Microsoft servers, use integrated authentication). |  |
| Default database user account passwords changed (no blank sa  passwords!). |  |
| Privileges for database users limited appropriately (queries should not  simply be executed as sa). |  |
| If not needed, extended stored procedures deleted from database software  and relevant libraries removed from the disk. |  |
| Database user passwords not embedded in application code. |  |
| Perform password audits regularly. |  |
| Applications |  |
| Threat models documented and approved by the appropriate team. |  |
| Appropriate security development lifecycle milestones achieved. |  |
| Development/QA/test/staging environments physically separated from  the production environment. Do not copy production data into QA/test/  staging. |  |
| Appropriately strong authentication implemented in the securest fashion  (e.g., via HTTPS, passwords stored as hashes, password self-support  functionality best practices, and so on). |  |
| Appropriate ACLs set for application directories and files. |  |
| Appropriate input validation and output encoding performed on the  server side. |  |
| Source code of application scripts, include files, and so on, sanitized of  secrets, private data, and confidential information. |  |
| Temporary and common files (e.g., .bak) removed from servers. |  |
| Authorization/session management implemented appropriately (strongly  recommend using platform-provided capabilities, such as ASPSESSIONID  or JSESSIONID, ASP.NET IsInRole, and so on). |  |
| Always perform explicit access control—don’t assume user won’t access  something just because he or she doesn‘t know the link or can’t tamper  with HTTP requests. |  |

Appendix A: Web Application Security Checklist

417

|  |  |
| --- | --- |
| Item | Check |
| Always grant a new session ID after a login; always have a logout feature;  use a timeout to expire sessions; and don’t allow multiple concurrent  sessions. |  |
| Application user roles established using least privilege. |  |
| If the application allows new users registration, use a CAPTCHA and  require e-mail validation. Do not allow weak passwords. |  |
| Encryption implemented using established algorithms that are appropriate  for the task. |  |
| Include files should be placed outside of virtual roots with proper ACLs. |  |
| On Microsoft IIS servers, include files should be renamed to .asp. |  |
| Dangerous API/function calls (e.g., RevertToSelf on IIS) identified and  avoided if possible. |  |
| Parameterized SQL queries required. |  |
| On .NET framework, review calls that can break out of the .NET  framework security (COM Interop, P/Invoke, Assert). |  |
| Proper error handling and security logging enabled. |  |
| Rigorous security source code audit performed. |  |
| Remote “black box” malicious input testing performed. |  |
| Perform password audits regularly. |  |
| Application vulnerability scans conducted regularly to mitigate against  application-level vulnerabilities. |  |
| Third-party manual pen-testing performed before release and after any  significant change is made to the application. |  |
| Client Side |  |
| *Note: In contrast to previous sections of this checklist, which are written from the*  *web application administrator or developer’s viewpoint, this section takes the end-*  *user’s perspective. Admins and developers should take note, however, and design*  *and implement their applications to meet these requirements.* |  |
| Personal firewall enabled with minimal allowed applications, both  inbound and outbound. |  |
| Run with least privilege. Never log on as Administrator (or equivalent  highly privileged account) on a system that you will use to browse the  Internet or read e-mail. |  |
| All client software is up-to-date on all relevant software security patches  (automatic updates optionally enabled). Be particularly diligent with IE—  we do not recommend using version prior to 8. |  |

418 Hacking Exposed Web Applications

|  |  |
| --- | --- |
| Item | Check |
| Antivirus software installed and configured to scan real-time (particularly  incoming mail attachments) and to automatically update. For example,  Microsoft Security Essentials is free and provides real-time protection  against viruses, spyware, and other malicious software (malware). |  |
| Anti-adware/spyware/malware and anti-phishing utilities installed in  addition to antivirus (assuming antivirus does not already have these  features). |  |
| Configure Internet client security conservatively; for example, Windows  Internet Options Control Panel (also accessible through IE and Outlook/  OE) should be configured as advocated in Chapter 9. |  |
| If configured separately, ensure other client software (especially e-mail!)  uses the most conservative security settings (e.g., Restricted Sites zone in  Microsoft e-mail clients). |  |
| Configure Office productivity programs as securely as possible; for  example, if you are using an old version of Microsoft Office, set the macro  security to Very High under Tools | Macro | Security (this is the default  setting in newer versions). |  |
| Cookie management enabled within the browser or via a third-party tool  such as CookiePal. |  |
| Disable caching of SSL data. |  |
| Don’t be gullible. Approach Internet-borne solicitations and transactions  with high skepticism. For sensitive URIs (e.g., online banking), manually  type addresses or use known-good Favorites/Bookmarks—never click  hyperlinks! |  |
| Keep your computing devices physically secure (especially mobile devices  such as laptops, Blackberrys, and cell phones). Do not store confidential  information on mobile devices unencrypted (including e-mail messages).  Also turn off Bluetooth and Wi-Fi when not in use. |  |
| Recommended Additional Client Configurations |  |
| Automatic software updates enabled (for example, Microsoft’s Automatic  Update Service). |  |
| E-mail software configured to read e-mail in plaintext. |  |
| Kill-bit set on unneeded ActiveX controls. |  |
| Change operating system default configurations (for example, instead of  the default C:\Windows, install with an unusual Windows folder name  like C:\Root). |  |
| Disable AutoComplete on your browser (automatic completion of HTML  forms with usernames, passwords, and other information). |  |
| Disable Browser History. |  |