***ICSC -- Web Application Security Guidance***

*Building a website or web application at Hopkins involves decisions on a number of levels. Security is among the most complex of these decisions. It requires attention to infrastructure configuration, application development, encryption, interfaces and on-going maintenance. Even a simple website with a serious vulnerability can be used by attackers as a launching point against the server, the JH Network, other sites or the visitors to the site. Maintaining security for any web asset requires careful planning and on-going commitment by technically competent staff. Many individuals or organizations looking to establish a web presence, especially one that matches their unique contribution to the institution, would be better off using existing web hosting, development and maintenance services. The direct cost of a website may be low, but the cost of ensuring that the site remains well-managed and secure requires an on-going resource commitment.*

*Yet no matter what the size or complexity of a web presence this checklist should help individuals or departments consider how to build and manage a website or web application. It is generally best to work through the checklist with technical staff to ensure that available technical resources are identified.*

*Addendum (February 2021) – Johns Hopkins now has a comprehensive site on ADA Web Accessibility --* <https://accessibility.jhu.edu/electronic-information-technology/>

|  | **Control** | **Tools** | **Procedures** | **Notes** |
| --- | --- | --- | --- | --- |
|  | 1. Prior to Development |  |  |  |
| 1.1 | Write a detailed web requirements and specifications plan |  | The ICSC publishes a PAS template for system building, but the requirements document need not go into that detail. | Check for :   * Mobile interfaces * Authentication * Third party plugins * Restricted information * Other unique application requirements. |
| 1.2 | Identify experienced Hopkins resources that may be able to meet requirements. | JHM Technology Innovation Center, Institutional Communications Offices. Divisional IT | Many such projects can be added to an existing framework or site. The best way to reduce risk from a web application is to leave it to professional management. It is usually best to contact departmental IT resources. Conduct interviews regarding experience, security tools and testing techniques, references. Work with departmental IT to identify issue. | Hopkins has relationships with several companies for specific services (e.g. Qualtrics for surveys, Blackboard for student systems, Salesforce for business apps). |
| 1.3 | Identify a secure, monitored hosting site current with patching and Web technologies -- Restricted applications outside a Hopkins data center must be approved by CISO. |  | Web applications are common targets for hackers, it is therefore critical that the application host be configured, hardened and monitored. Consider responsibility for server and framework patching. | Deploying an externally hosted application should be coordinated with schools/divisional, departmental and/or JHM entity. |
| 1.4 | Choose regularly updated web and mobile frameworks. | Wordpress, Microsoft Web tools, Salesforce | It is difficult to build a robust, secure web application without using an industry-grade framework. Consider the competencies of the development, team, hosting entity and tools. Data persistence layers should be chosen with similar considerations in mind. | NoSQL data tools may be subject to SQL Injection-like attacks and the major ones are no more secure than SQL databases. Some older web frameworks (e.g. older ColdFusion) may be deprecated in the near future and should be avoided. |
| 1.5 | Develop a plan for continuous web application improvement and maintenance. |  | Web applications suffer ‘bitrot’ (degradation over time) at a rapid pace. It is therefore critical that plans be made for continuous development and refresh at least once every three years. |  |
| 1.6 | Identify resources for continuous monitoring and support, including patching and assessment of potential future vulnerabilities. | Tenable NESSUS, Accunetix | Web application vulnerabilities against configurations, shells, code and encryption are published frequently. Every Restricted web application or even one that requires https needs an individual to work with security and hosting groups to assess patching. | This individual may also ‘own’ the ongoing Tenable/NESSUS (see below) scanning required below. |
| 1.7 | For sites that allow users or third parties to contribute persistent data, curation and monitoring responsibilities should be assigned. |  | Sites that maintain third party content, including, for example, user comments or other feedback pages, should have a procedure for reviewing such input periodically for security and appropriateness. | This is not principally a security issue (although bad links and content injection can be problems). It is just a common sense check on trolls. |
| 1.8 | Establish procedures for management/curation of internally created site content | Change management procedures | A procedure should be documented briefly for adding content to content-rich sites, principally to protect against publication of sensitive information and for removing out-of-date information or links. | Publication of sensitive information on the web is the most common cause of notification events at JH and is caused almost entirely by not carefully watching what is made accessible through our web sites. |
|  |  |  |  |  |
| 1.9 | Do not accept credit card purchases on a Hopkins-domain website. |  | All credit card completion pages should be linked to a third party secure website as the credit card completion page. These completion pages should not be maintained on the JH Network. Discuss payment options with a Hopkins Treasure Office. | Any page that links to a third party completion page should be monitored closely to ensure against injection attacks that send users to malicious spoofed sites. |
|  | 2. Access Control |  |  |  |
| 2.1 | Enable enterprise Web Single Sign On (SSO) for authenticated access for Hopkins-specific sites. |  | If the application is to be used principally by more than a handful of Hopkins personnel, Web SSO from the enterprise IT (i.e. JHED or Cloud teams) should be used. This greatly simplifies log-in for users and administrators. Web SSO is an authentication tool for JHED authentication and can be used against multiple sites. *Ensure that JHED credentials are not harvested by the application through any authentication mechanism.* | Any JHED authentication of a desktop web application must be done through the web SSO. No other forms may be used and, of course, credentials may not be cached or stored. |
| 2.2 | For authenticating individuals outside Hopkins, specifically those associated with .edu’s, InCommon/Shibboleth is a preferred approach. |  | For web applications for which there may be users at other academic organizations, Shibboleth authentication through SSO may be used. Shibboleth is a SAML Standards based authentication tool | Shibboleth is the technology of the InCommon foundation. |
| 2.3 | For authenticating individuals outside Hopkins with no university or research, affiliations, industry authentication through web services (OAuth) is generally acceptable. For Restricted applications, third party authentication must be approved by CISO. |  | It may be necessary to use credentials from third party providers (e.g. Facebook Connect, Google or Microsoft) Hopkins policy allows use of SAML, OAuth and OAuth2 for these purposes. For Restricted applications, developers should confer with the enterprise AD group for third party access options (e.g. Microsoft Azure Auth) | No JHED password credentials may be provided to a third party identity provider. All communications must be encrypted using TLS, without reliance on deprecated ciphers or SSL 3.0. |
| 2.4 | Do not insert hard coded connection strings into applications. |  | This is frequently an issue for any code (e.g. JavaScript) used by the client, but it also good guidance for back-end code. | This also applied to passwords and password hashes. |
| 2.5 | If the site uses non-JHED authentication and must store passwords, appropriate encryption and salting must be used. | bcrypt, PBKDF2 | JH prohibits storage of JHED credentials, but where other forms of authentication are used, storage of credentials may be allowed. If possible, credentials should not be stored on the web server. In addition credentials must be stored in a salted and encrypted folder. | All logins and storage must be encrypted and salted. |
| 2.6 | For certain Restricted applications where password authentication alone may not be sufficient, enterprise Multi-Factor Authentication (MFA) may be required. | Google Authenticator, Duo | Hopkins has a number of methods to protect web services, including cell-phone-based one-time passwords to protect highly sensitive assets. For many users, the preferred authentication method is the Google Authenticator mobile application. | Generally requires JHED sign-in and that users have a smart phone, table or dedicated desktop application at the time of login. |
|  | 3. Server Configuration |  |  |  |
| 3.1 | Repurposed web servers should be re-imaged, wiped and a fresh copy of the operating system installed prior to deployment. Virtual servers should not be repurposed and should use a clean copy of data and the operating system. |  | Servers for the web should include fresh OS and framework installations so as to avoid hidden folders. | Ensuring that servers are appropriately patched upon loading with endpoint protection and monitoring is critical. One should be especially careful regarding test/dev environments. |
| 3.2 | Web servers hosting sites accessible outside the Hopkins network should be treated as potentially vulnerable and placed in an appropriate Web DMZ network segment. |  | Hopkins has several network segments structured for external services, web servers should be placed in these DMZ-like segments. Academic segments may be used to host non-Restricted web servers. |  |
| 3.3 | Administrative access to Restricted or otherwise critical web servers must be protected with MFA (e.g. token or phone based) and it is strongly recommended that all web servers be so protected. | Duo, SafeNet | Many web attacks are designed to create new accounts with escalated privileges or to otherwise find credentials of current administrators. The best approach for preventing this is to strengthen administrative authentication through MFA. Organizations are free to experiment with these approaches, but the standard is that a password alone (even a long password) should not open administrative access to web servers or back-end databases. |  |
| 3.4 | Server configuration should follow JH standards and guidelines for server configuration. | Linux and Windows Server security standards | The JH Windows Server and Linux standards should be followed. The Hopkins PAS Template should be used for critical or Restricted web servers. It is especially important that internal messaging protocols such as snmp and smb are up-to-date with all vulnerable versions disable (i.e. SMB1.0 is strictly prohibited) | Web servers should be maintained in a protected data center with configurations for both checking infection by the server and stopping malicious code from spreading due to an injection attack. |
| 3.5 | Servers that host multiple websites may not host websites with substantially different risk/security postures (i.e. Restricted sites should not be mixed with non-Restricted sites). | Service bus, messaging systems | Web servers are often application servers and may be the nexus of multiple applications. Restricted sites/applications should not be mixed with those of a lesser security posture. | Even internal-only web servers may be vulnerable to attack, and therefore they should be designed to isolate web content from the service bus. |
| 3.6 | All sites that receive user input through forms or content pages must use appropriate https encryption. | Qualys -- https://www.ssllabs.  com/ssltest/ | Insecure protocols should be scanned and removed. For marginal ciphers, it may be necessary to scan application logs to determine whether a substantial number of visitors are using deprecated protocols and build a migration strategy accordingly. | TCP 1.0 and 1.1 are deprecated and prohibited. |
| 3.7 | If possible, place data persistence layer and web server on different servers or virtual instances. | Service bus | n-tier architectures generally require that persistent data storage be separate from the web application tier. For many frameworks, however, this separation is impractical. In those situations, it may be possible to create logical separation through access control and logging. | Web servers are highly vulnerable to zero-day attacks, and one should assume therefore that they will be compromised from time to time. The goal here is to protect underlying data assets even if the web tier is compromised. There are a number of techniques for this, and some are framework-dependent. |
| 3.8 | Place Wordpress sites behind a dedicated application firewall. | Wordfence | Wordpress sites are so routinely attacked that additional security measures should be deployed, including real-time patching of frameworks and plugins. Yet even with increased vigilance, WP-specific firewalls provide additional security |  |
| 3.9 | Ensure that all unnecessary ports and services are turned off. | Wireshark | Follow JH server guidance, vendor OS guidance and standards for the web development framework. Service accounts should also be minimized and authentication checked. | Identify all services and ports. If there are questionable services or ports, consider running Wireshark during testing to identify unanticipated communications.  Remove all sample scripts and example databases. |
| 3.10 | After all infrastructures are in place, run a Tenable/NESSUS scan against the configuration and remedy findings. | NESSUS Tenable | Routine vulnerability scanning is one of the cornerstones of our web application security strategy. It captures configuration issues and some potential errors in coding. The initial scan requires some attention to configuration and close analysis of results. | System owners can register in Tenable and run their own scans. Application owners and developers should review the results with systems administrators to ensure that a baseline is in place. |
|  |  |  |  |  |
|  | 4. Web Application Development |  |  |  |
| 4.1 | Take advantage of OWASP, framework vendor and other security guidance for coding practices. |  | Many Hopkins sites rely on frameworks (e.g. WordPress, .NET). Major frameworks maintain guidance on nearly all major security threats and risks. | The OWASP Top Ten is a good guide for overall security threat assessment |
| 4.2 | Ensure that the site has been added to the Hopkins URL database. |  | Internally and externally hosted sites must be registered in the Hopkins URL database managed by enterprise web services. Work with local/departmental IT to ensure that the URL is so noted in the systems. | The URL database is required for compliance and systems management purposes. |
| 4.3 | Use web application scanning tools to check for code vulnerabilities. | Accunetix (JH-managed), Burp Suite, sqlmap | IT@JH maintains licenses to several commercial application scanning tools for web vulnerabilities. No Restricted web application web site should be deployed without an initial scan and others upon change of code or update. | These application scans are complementary to Tenable/NESSUS scans. They can provide deeper views into code quality. |
| 4.4 | Prevent SQL injection attacks through input validation, escaping characters and proper prepared statements/stored procedures. Stored procedures should be circumscribed. | ORM, stored procedures, parameterized queries | The most serious web vulnerabilities are often in SQL injection. Fortunately, these are some of the easiest to prevent with a number of approaches. Audit ‘execute’ permissions on stored procedures or prepared statements should have minimum necessary polices. | OWASP has guidance on this for all major frameworks and programming languages. |
| 4.5 | Prevent cross-site scripting attacks (XSS) through input validation escaping characters, etc. Clean and validate user input (including all machine input) to protect against injection attacks. | ORM, data input validation | XSS is another subtler form of injection attack involving non-validated input. | iFrame hijacking can also be a problem here, therefore developers should be careful about third party content. |
| 4.6 | Avoid mixing http and https on the same site or application. For sites with both http and https, test for session hijacking through cookies and other means. |  | Do not expose identifiers in URL’s, error messages or logs. Session identifiers should only be located in the HTTP cookie header. | Do development of https protected sites using https in the development environment. |
| 4.7 | Check for file inclusion safety. |  | Applications should prevent users from directly accessing internal objects, API's, files, and databases. The application should interact on behalf of the user. |  |
| 4.8 | Avoid requiring client-side Java, Flash, Silverlight or other third party browser plugins |  | HTML5 provides a rich framework for interactivity. Third party plugins are difficult to standardize across browsers |  |
| 4.9 | Any externally-facing Restricted application must use security headers. It is strongly recommended that applications use them also. | ssllabs | These are now recognized by nearly all web browsers. | X-Content-Type-Options  X-XSS-Protection  X-Frame-Options  Cache-Control  X-Content-Security-Policy  Strict-Transport-Security |
| 4.10 | To ensure continuous service and performance, sites should maintain CDN context locally even if live linked to a third party. | jQuery, bootstrap | CDN sites may be blocked from time to time, thus interrupting web application performance. A local copy therefore should be maintained. |  |
| 4.11 | Mobile development should follow the Mobile Development Security Guidance on the IT@JH information security page. | Wireshark | Mobile development involves a number of unique challenges and the Mobile Guidance walks through many of the major areas. | Mobile application’s routinely have issues with losing TLS connectivity and reverting to plain text. Therefore test these applications with a packet capture tool like Wireshark. |
| 4.12 | Privacy policy – for public sites, maintain a live pointer to an appropriate Hopkins web privacy policy |  | Identify appropriate institutional Privacy policy and link on site. |  |
| 4.13 | Evaluate the application’s ADA accessibility risk and conduct tests accordingly to maintain compliance. |  | Test for ADA compliance using appropriate tools. <https://accessibility.jhu.edu/electronic-information-technology/> |  |
| 4.14 | All testing should be on test data and on dev/test versions rather than production |  | Hopkins standards requires that testing not be done on Restricted information except under carefully controlled (and risk assessed) circumstances. De-identify data and create specific testing data sets. | Check with the JH Testing and Release Standards by the ICSC |
|  |  |  |  |  |
|  | 5. Monitoring and Maintenance |  |  |  |
| 5.1 | Follow ICSC guidance on log monitoring and management | Splunk | Web applications generate rich logs and a program of log retention and management can be built on these logs as a matter of course |  |
| 5.2 | Alert on new administrator accounts or changed access rights on existing accounts |  | This is a common practice for web application sites. |  |
| 5.3 | Log and monitor input and output validation errors |  | These are indicators of injection attacks whether as GET’s or POST’s |  |
| 5.4 | Log and monitor session management failures |  | Systems failures vary by OS and framework, but these can be good indicators of Denial of Service attacks. |  |
| 5.5 | Other logging events of value |  | There are other events worth logging including -- Application shutdowns, Use of high risk functionality, Configuration changes, Data changes, Anomalous actions |  |
| 5.6 | All applications must be scanned routinely for security vulnerabilities. |  | Registering for enterprise vulnerability scans from Tenable NESSUS and Accunetix is typically required as part of the site registration process. |  |
|  |  |  |  |  |