Australian Government
Australian Signals Directorate

ACSC Australian Cyber Security Centre

Information Security Manual

Published: 02 March 2023

Guidelines for Software Development

Application development

Types of application development

These guidelines are applicable to both traditional application development and mobile application development.

Development, testing and production environments

Segregating development, testing and production environments, and associated data, can limit the spread of malicious code and minimises the likelihood of faulty code being introduced into a production environment. Furthermore, protecting the authoritative source for software is critical to preventing malicious code being surreptitiously introduced into software.

Control: ISM-0400; Revision: 5; Updated: Aug-20; Applicability: All; Essential Eight: N/A
Development, testing and production environments are segregated.

Control: ISM-1419; Revision: 1; Updated: Sep-18; Applicability: All; Essential Eight: N/A
Development and modification of software only takes place in development environments.

Control: ISM-1420; Revision: 4; Updated: Mar-22; Applicability: All; Essential Eight: N/A
Data from production environments is not used in a development or testing environment unless the environment is secured to the same level as the production environment.

Control: ISM-1422; Revision: 3; Updated: Sep-18; Applicability: All; Essential Eight: N/A
Unauthorised access to the authoritative source for software is prevented.

Control: ISM-1816; Revision: 0; Updated: Dec-22; Applicability: All; Essential Eight: N/A
Unauthorised modification of the authoritative source for software is prevented.

Secure software design and development

The use of secure-by-design and secure-by-default principles, memory-safe programming languages (such as C#, Go, Java, Ruby, Rust and Swift), and secure programming practices, that are supported by agile software development practices and threat modelling, are an important part of application development as they can assist with the identification and mitigation of at risk software components and risky programming practices. In addition, providing mechanisms to assist in determining the authenticity and integrity of applications, while configuring them in a secure manner, can assist with software supply chain security activities.

Control: ISM-0401; Revision: 6; Updated: Mar-23; Applicability: All; Essential Eight: N/A

Secure-by-design and secure-by-default principles, use of memory-safe programming languages where possible, and secure programming practices are used as part of application development.

Control: ISM-1780; Revision: 0; Updated: Jun-22; Applicability: All; Essential Eight: N/A
SecDevOps practices are used for application development.

Control: ISM-1238; Revision: 4; Updated: Mar-22; Applicability: All; Essential Eight: N/A
Threat modelling is used in support of application development.

Control: ISM-1796; Revision: 0; Updated: Sep-22; Applicability: All; Essential Eight: N/A
Files containing executable content are digitally signed as part of application development.

Control: ISM-1797; Revision: 0; Updated: Sep-22; Applicability: All; Essential Eight: N/A
Installers, patches and updates are digitally signed or provided with cryptographic checksums as part of application development.

Control: ISM-1798; Revision: 0; Updated: Sep-22; Applicability: All; Essential Eight: N/A
Secure configuration guidance is produced as part of application development.

Software bill of materials

A software bill of materials is a list of open source and commercial software components used in application development. This can assist in providing greater cyber supply chain transparency for consumers by allowing for easier identification and management of security risks associated with individual software components used by applications.

Control: ISM-1730; Revision: 0; Updated: Dec-21; Applicability: All; Essential Eight: N/A
A software bill of materials is produced and made available to consumers of software.

Application security testing

Application security testing can assist software developers in identifying security vulnerabilities in their applications. In doing so, both static application security testing, as well as dynamic application security testing, should be performed in order to achieve comprehensive test coverage. Furthermore, software developers may choose to use an additional independent party to assist with removing any potential for bias that might occur when they test their own applications.

Control: ISM-0402; Revision: 5; Updated: Mar-23; Applicability: All; Essential Eight: N/A
Applications are comprehensively tested for security vulnerabilities, using both static application security testing and dynamic application security testing, prior to their initial release and any subsequent releases.

Control: ISM-1754; Revision: 1; Updated: Mar-23; Applicability: All; Essential Eight: N/A
Security vulnerabilities identified in applications are resolved by software developers in a timely manner.

Vulnerability disclosure program

Implementing a vulnerability disclosure program, based on responsible disclosure, can assist an organisation to improve the security of their products and services as it provides a way for security researchers and other members of the public to responsibly notify them of security vulnerabilities in a coordinated manner. Furthermore, following the verification and resolution of reported security vulnerabilities, it can assist an organisation in notifying their customers of security vulnerabilities that have been discovered in their products and services, and any patches, updates or vendor mitigations that should be applied.

A vulnerability disclosure program should include processes and procedures for receiving, verifying, resolving and reporting security vulnerabilities disclosed by both internal and external parties. In support of this, a vulnerability disclosure policy should be made publicly available that covers:

- the purpose of the vulnerability disclosure program

- types of security research that are and are not allowed
- how to report any security vulnerabilities
- actions, and associated timeframes, upon notification of security vulnerabilities
- expectations regarding the public disclosure of security vulnerabilities
- any recognition or reward for finders of security vulnerabilities.

Finally, the Australian Cyber Security Centre (ACSC) encourages security researchers and other members of the public to responsibly report security vulnerabilities directly to an organisation. However, the ACSC recognises that this is not always practical, initial attempts at communication may be unsuccessful or the person making the report may not wish to do so directly. In such cases, security vulnerabilities can be reported to the ACSC as an independent coordinator.

Control: ISM-1616; Revision: 0; Updated: Aug-20; Applicability: All; Essential Eight: N/A
A vulnerability disclosure program is implemented to assist with the secure development and maintenance of products and services.

Control: ISM-1755; Revision: 1; Updated: Dec-22; Applicability: All; Essential Eight: N/A
A vulnerability disclosure policy is developed, implemented and maintained.

Control: ISM-1756; Revision: 1; Updated: Dec-22; Applicability: All; Essential Eight: N/A
Vulnerability disclosure processes, and supporting vulnerability disclosure procedures, are developed, implemented and maintained.

Control: ISM-1717; Revision: 1; Updated: Mar-22; Applicability: All; Essential Eight: N/A
A 'security.txt' file is hosted for all internet-facing organisational domains to assist in the responsible disclosure of security vulnerabilities in an organisation's products and services.

Further information

Further information on a secure development life cycle model, known as *The Trustworthy Computing Security Development Lifecycle*, is available from Microsoft.

Further information on a secure software development framework can be found in National Institute of Standards and Technology Special Publication 800-218, *Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities*.

Further information on memory-safe programming languages can be found in the United States' National Security Agency's *Software Memory Safety* publication.

Further information on secure programming practices is available from the Carnegie Mellon University's Software Engineering Institute.

Further information on cyber supply chain transparency, and recommended content for a software bill of materials, can be found in the United States' National Telecommunications and Information Administration's *The Minimum Elements For a Software Bill of Materials (SBOM)* publication.

Further information on implementing a vulnerability disclosure program can be found in:

- Google's *Starting a Vulnerability Disclosure Program*
- European Union Agency for Cybersecurity's *Good Practice Guide on Vulnerability Disclosure*
- Netherland's National Cyber Security Centre's *Coordinated Vulnerability Disclosure: The Guideline*
- Carnegie Mellon University's Software Engineering Institute's *The CERT Guide to Coordinated Vulnerability Disclosure*

- International Organization for Standardization/International Electrotechnical Commission 29147:2018, *Information technology – Security techniques – Vulnerability disclosure*
- International Organization for Standardization/International Electrotechnical Commission 30111:2019, *Information technology – Security techniques – Vulnerability handling processes*.

Further information on developing a vulnerability disclosure policy is available from the disclose.io project to assist an organisation with their implementation.

Further information on recommended contents for a 'security.txt' file is available to assist an organisation with their implementation.

Further information on reporting security vulnerabilities to the ACSC as an independent coordinator is available from the ACSC.

Web application development

Open Web Application Security Projects

The Open Web Application Security Project (OWASP) provides comprehensive resources for software developers that should be followed when developing web applications.

Control: ISM-0971; Revision: 8; Updated: Mar-23; Applicability: All; Essential Eight: N/A
The OWASP Application Security Verification Standard is used in the development of web applications.

Control: ISM-1849; Revision: 0; Updated: Mar-23; Applicability: All; Essential Eight: N/A
The OWASP Top Ten Proactive Controls are used in the development of web applications.

Control: ISM-1850; Revision: 0; Updated: Mar-23; Applicability: All; Essential Eight: N/A
The OWASP Top 10 are mitigated in the development of web applications.

Web application frameworks

Web application frameworks can be leveraged by software developers to enhance the security of web applications while decreasing development time. These resources can assist in securely implementing complex software functions, such as session management, input handling and cryptographic operations.

Control: ISM-1239; Revision: 4; Updated: Mar-22; Applicability: All; Essential Eight: N/A
Robust web application frameworks are used in the development of web applications.

Web application interactions

Hypertext Transfer Protocol Secure (HTTPS) is the Hypertext Transfer Protocol secured by Transport Layer Security (TLS) encryption. The use of HTTPS for web applications can assist in ensuring that interactions with web applications are confidential and that the integrity of such interactions are also maintained.

Control: ISM-1552; Revision: 0; Updated: Oct-19; Applicability: All; Essential Eight: N/A
All web application content is offered exclusively using HTTPS.

Web application programming interfaces

Web application programming interfaces (APIs) can facilitate the exchange of data between computing devices. However, poorly-secured web APIs might facilitate unauthorised access to data not authorised for release into the public domain. In addition, poorly-secured web APIs might facilitate unauthorised modification of data. In such cases, ensuring authentication and authorisation of clients is performed when clients call web APIs can assist in mitigating unauthorised access to, or modification of, data.

Control: ISM-1817; Revision: 1; Updated: Mar-23; Applicability: All; Essential Eight: N/A
Authentication and authorisation of clients is performed when clients call web APIs that facilitate access to data not authorised for release into the public domain.

Control: ISM-1818; Revision: 1; Updated: Mar-23; Applicability: All; Essential Eight: N/A
Authentication and authorisation of clients is performed when clients call web APIs that facilitate modification of data.

Control: ISM-1851; Revision: 0; Updated: Mar-23; Applicability: All; Essential Eight: N/A
The OWASP API Security Top 10 are mitigated in the development of web APIs.

Web application input handling

Most web application security vulnerabilities are caused by a lack of secure input handling. As such, it is essential that web applications do not trust any input, such as website addresses and their parameters, Hypertext Markup Language (HTML) form data, cookie values, or request headers, without performing validation or sanitisation. Examples of validation and sanitisation include ensuring a telephone form field contains only numerals, ensuring data used in a Structured Query Language query is sanitised properly and ensuring Unicode input is handled appropriately.

Control: ISM-1240; Revision: 3; Updated: Mar-22; Applicability: All; Essential Eight: N/A
Validation or sanitisation is performed on all input handled by web applications.

Web application output encoding

The likelihood of cross-site scripting and other content injection attacks can be reduced through the use of output encoding. In particular, output encoding is useful when external data sources, which may not be subject to the same level of input filtering, are output to users. The most common example of output encoding is the conversion of potentially dangerous HTML characters into their encoded equivalents, such as '<', '>' and '&' into '&lt;', '&gt;' and '&amp;'.

Control: ISM-1241; Revision: 4; Updated: Mar-22; Applicability: All; Essential Eight: N/A
Output encoding is performed on all output produced by web applications.

Web browser-based controls

Web browser-based controls, such as Content-Security-Policy, Hypertext Transfer Protocol Strict Transport Security (HSTS) and X-Frame-Options, can be used by web applications to help protect themselves and their users. This is achieved via setting security policy in response headers from web applications which web browsers then apply. Note, since the controls are applied via response headers, they can be applied to legacy or proprietary web applications where changes to their source code may be impractical.

Control: ISM-1424; Revision: 4; Updated: Mar-22; Applicability: All; Essential Eight: N/A
Web applications implement Content-Security-Policy, HSTS and X-Frame-Options via security policy in response headers.

Web application event logging

Web application events can assist in monitoring the security posture of web applications, detecting malicious behaviour and contributing to investigations following cyber security incidents. To facilitate such activities, web application event logs should be captured and stored centrally.

Control: ISM-1536; Revision: 1; Updated: Mar-22; Applicability: All; Essential Eight: N/A
The following events are logged for web applications: attempted access that is denied, crashes and error messages, and search queries initiated by users.

Control: ISM-1757; Revision: 1; Updated: Dec-22; Applicability: All; Essential Eight: N/A
Web application event logs are stored centrally.

Further information

Further information on web application security can be found in the *OWASP Application Security Verification Standard* and *OWASP Top Ten Proactive Controls* publications.

Further information on web application security risks can be found in the *OWASP Top 10* publication.

Further information on implementing HTTPS can be found in the ACSC's *Implementing Certificates, TLS, HTTPS and Opportunistic TLS* publication.

Further information on using TLS in HTTPS can be found in the Transport Layer Security section of the *Guidelines for Cryptography*.

Further information on web application security can be found in the ACSC's *Protecting Web Applications and Users* and *Securing Content Management Systems* publications.

Further information on API security can be found in the *OWASP API Security Top 10* publication.

Further information on strong authentication can be found in the authentication hardening section of the *Guidelines for System Hardening*.

Further information on event logging can be found in the event logging and monitoring section of the *Guidelines for System Monitoring*.