Application Development Guidance: Introduction

Guidance for risk assessors, and developers of applications which will be run on devices handling OFFICIAL data.

Guidance Principles

This guidance looks to mitigate potential loss of data marked OFFICIAL through secure development guidance. It provides guidance for both generic application development, and guidance specific to particular platforms. The following principles are followed throughout:

- Applications which store, process, handle, or have network access to protectively marked information, must be developed within certain security constraints, and should be audited and assessed before use.
- If protectively marked information is stored using the platform's native functionality, then third-party applications may be able to access that information through platform API methods. Exactly which data the applications have access to, and what constraints are involved, must be understood.
- At this time, CESG cannot provide any services for the assessment of third-party applications. Departments must undertake this work on an individual, per-application basis. CESG may look to provide such services or assessment schemes in the future.

Aims & Assumptions

The aims of the recommendations in this guidance are to:

- Ensure that sensitive data and credentials are protected appropriately when stored and transmitted.
- Ensure the device is protected by any existing protective monitoring solutions that the network might offer.

- Minimise the opportunity for accidental data leakage across application boundaries, whilst still permitting it where that was the intention.
- Ensure only authorised parties can access sensitive information.
- Maximise the usability of these applications whilst meeting a minimum required level of security in the development of the application.
- Restrict access to OFFICIAL classified files to applications designed to handle such material in a secure manner.

In attempting to achieve these goals, the following assumptions are made:

- The device is configured broadly in line with the CESG platform configuration guidance for iOS and Android.
- The device is able to be Jailbroken/rooted, which could compromise unencrypted data directly, and even encrypted data if the user subsequently unlocks the device and accesses the data without realising the device has been attacked.
- The device could have other third-party applications installed.
- The device will be in a locked state if lost or stolen.

Generic Mobile Application Development Guidance

Applications across multiple platforms can share common vulnerabilities. When developing an application, developers should seek to make use of any security mechanisms that are built into the platform they are developing for. Developers should be aware of many of the common security issues that they may fall victim to, in order to protect their application from compromise. The following is platform agnostic advice on how to mitigate common security issues.

Secure Data Storage

Data Storage APIs

Sensitive information should not be stored on a device when it is not required. When sensitive data is required to be stored on a device, developers should look to make use of any native protected data storage APIs that are available to the platform. Most contemporary mobile platforms provide documented APIs that allow data to be stored at different levels of security. Sensitive information should

always be encrypted while not in use, and should be guarded by an authentication mechanism, such as a passcode. When it is no longer required on the device, it should be securely removed.

Cryptography

Implementing an additional layer of data encryption before using data storage APIs provided on a device will help protect sensitive information in the event that an attacker has compromised the host device. Storing any cryptographic keys on the device will reduce the effectiveness of an additional cryptographic layer as keys stored locally could be recovered from the device (though these keys could be combined with a user credential to strengthen them). Storing the keys on a remote server would prevent an attacker with physical access to the device from retrieving them, though would require the application to authenticate to the server, and have an internet connection.

Data Access Authorisation

Sensitive information should always be stored securely until the user has been authorised to view the information, either through a passcode, or some other means of authentication.

Secure Data Transmission

Any sensitive information being transferred between device and server should be sent over an appropriate encryption mechanism. Supported ciphers should be restricted on both ends of the communication so that only strong ciphers may be used. Additional steps should be taken in order to maximise the security of the data connection. SSL Pinning for example, would allow the application to be restricted so that it may only form secure connections to a host with a known, trusted certificate. Sensitive data should never be sent over an unencrypted connection.

Session Handling

Session handling requires appropriate controls to be placed on the backend server to which the application connects. The backend server should treat the application as an untrusted entity; only allowing it access to content that it has been authorised to. When an application has authenticated, the backend server should enforce a session timeout, after which the application is forced to re-authenticate. The timeout of the session will depend on the application and

content that it has access to, however 15 minutes is a recommended amount of time for sensitive information. The backend server should be responsible for management of the session; enforcing access limitations on the mobile application when its session is invalidated.

Application Hardening

Stack Protection

Applications should be compiled to take advantage of any protection mechanisms that are available to the platform. Features such as Address Space Layout Randomisation (ASLR) and Stack Canaries should be enforced during compilation in order to make the application more difficult to exploit.

Reverse Engineering

Steps should be taken to make the application more difficult to reverse engineer. Using a non-reflection based language such as C to perform sensitive actions will require more effort from an attacker to modify its logic at runtime. Developers should be careful when implementing this however, as using an unmanaged language can open up the application to memory corruption vulnerabilities. Implemented obfuscation techniques can also make the application more difficult to reverse. Code obfuscation tools for example, will make it much more difficult for a human to read reversed engineered source code.

Jailbreak/Root Detection

Jailbroken or rooted devices are a threat to sensitive data on end user devices. Implementing checks to detect if the device has been compromised should be considered a method of hardening the application. Detection will always be subject to circumvention by a determined attacker, however tests for common jailbreak/rooting methods allows for the application to take appropriate steps.

Third Party Applications

Organisations may consider permitting third party applications on to the same device alongside applications that handle sensitive data. When deploying third-party applications, two of the primary concerns for an organisation are protecting the enterprise network infrastructure from attack via the third-party applications, and preventing data from leaking from a sensitive data store into a third-party application.

As the behaviour of third-party applications cannot normally be modified, this means that the protection has to be provided elsewhere, via network protections and appropriate use of data-stores. Where possible, the developers of the application(s) should be approached in order to gain a deeper understanding of the construction of their product. Applications that are likely to be sources of data leaks, track user movements or interfere with other applications should not be installed.

PUBLISHED

7 August 2016

REVIEWED

7 August 2016

VERSION

1.0