Application Developer Guidance

This mitigation describes any guidance or training given to developers of applications to avoid introducing security weaknesses that an adversary may be able to take advantage of.

ID: M1013
Version: 1.0
Created: 25 October 2017
Last Modified: 17 October 2018

Version Permalink

Techniques Addressed by Mitigation

ATT&CK® Navigator Layers

Domain	ID		Name	Use
Enterprise	T1564	.009	Hide Artifacts: Resource Forking	Configure applications to use the application bundle structure which leverages the `/Resources` folder location.[1]
Enterprise	T1574		Hijack Execution Flow	When possible, include hash values in manifest files to help prevent side-loading of malicious libraries.[2]
		.002	DLL Side-Loading	When possible, include hash values in manifest files to help prevent side-loading of malicious libraries.[2]
Enterprise	T1559		Inter-Process Communication	Enable the Hardened Runtime capability when developing applications. Do not include the `com.apple.security.get-task-allow` entitlement with the value set to any variation of true.
		.003	XPC Services	Enable the Hardened Runtime capability when developing applications. Do not include the `com.apple.security.get-task-allow` entitlement with the value set to any variation of true.
Enterprise	T1647		Plist File Modification	Ensure applications are using Apple's developer guidance which enables hardened runtime.[3]
Enterprise	T1078		Valid Accounts	Ensure that applications do not store sensitive data or credentials insecurely. (e.g. plaintext credentials in code, published credentials in repositories, or credentials in public cloud storage).
Mobile	T1626		Abuse Elevation Control Mechanism	Applications very rarely require administrator permission. Developers should be cautioned against using this higher degree of access to avoid being flagged as a potentially malicious application.
Mobile	T1517		Access Notifications	Application developers could be encouraged to avoid placing sensitive data in notification text.
Mobile	T1513		Screen Capture	Application developers can apply the `FLAG_SECURE` property to sensitive screens within their apps to make it more difficult for the screen contents to be captured.[4]
Mobile	T1635		Steal Application Access Token	Developers should use Android App Links[5] and iOS Universal Links[6] to provide a secure binding between URIs and applications, preventing malicious applications from intercepting redirections. Additionally, for OAuth use cases, PKCE[7] should be used to prevent use of stolen authorization codes.
		.001	URI Hijacking	Developers should use Android App Links[5] and iOS Universal Links[6] to provide a secure binding between URIs and applications, preventing malicious applications from intercepting redirections. Additionally, for OAuth use cases, PKCE[7] should be used to prevent use of stolen authorization codes.
Mobile	T1474		Supply Chain Compromise	Application developers should be cautious when selecting third-party libraries to integrate into their application.
		.001	Compromise Software Dependencies and Development Tools	Application developers should be cautious when selecting third-party libraries to integrate into their application.

References