The Password is Dead…
Long Live the Passkey!

About Me

Focus

Modernizing Digital Workplace Solutions,
Microsoft MVP, Apple Mobile Device Management

From

The Netherlands

My Blog

https://www.intuneirl.com

Certifications

Microsoft 365 Certified Enterprise Administrator Expert

Prince2 & Six Sigma

Hobbies

Travelling & cooking

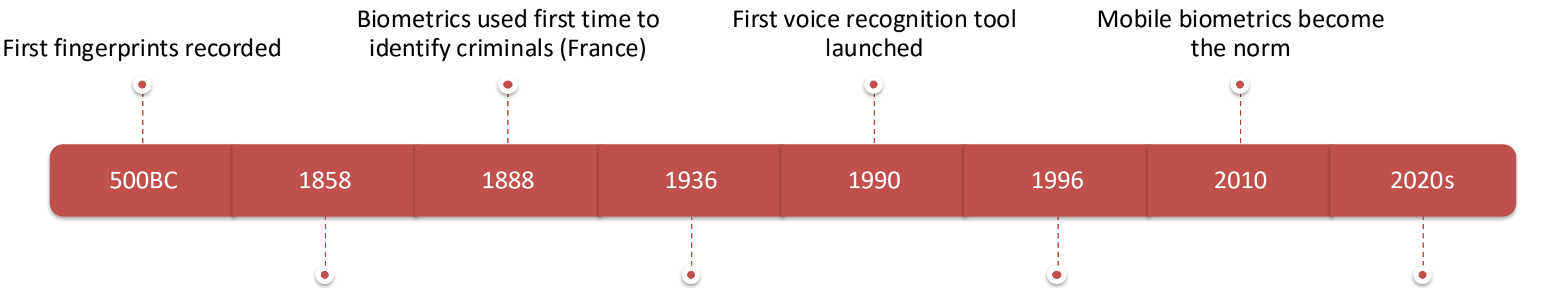
Contact

someshpathak@hotmail.com

https://x.com/pathak_somesh

https://www.linkedin.com/in/someshpathak/

History of Authentication



Biometrics Evolution

1960s — First computer passwords implemented at MIT

1970s — Unix introduces cryptographic password hashing

1985 — First commercial biometric system for time and attendance

1999 — FIDO (Fast Identity Online) Alliance formed to address password limitations

2004 — Bill Gates predicts passwords becoming obsolete at RSA Conference

2012 — Two-factor authentication becomes mainstream with SMS codes

2019 — WebAuthn becomes official W3C standard

2022 — Apple, Google, and Microsoft commit to passwordless FIDO standard

Apple Security Evolution

2007	2010	2013	2014	2016	2017	2020	2022	2022	2024
App sandboxing introduced	Find My Phone introduced	iPhone launches with Secure Enclave architecture	Data encryption – Enabled by default on all iOS devices	System Integrity Protection (SIP) introduced on macOS	iPhone X launches with Face ID and the Neural Engine	M1 chips with advanced security, including boot protections and isolation	Passkeys with iOS 16 & macOS Ventura	Lockdown Mode (iOS)	Managed Device Attestation

“Passkeys” or “passkeys” ???

Why passkeys?

Phishing Attacks (2.9%)

Credential Thefts (40%)

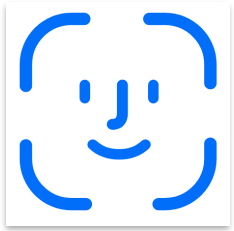
Two Factor Authentication Bypass

What Are “passkeys”?

Uses iCloud Keychain public key credentials, eliminating the need for passwords

Rely on biometric identification – Prove Who You are?

Your Apple device generates a unique public-private key pair for every account



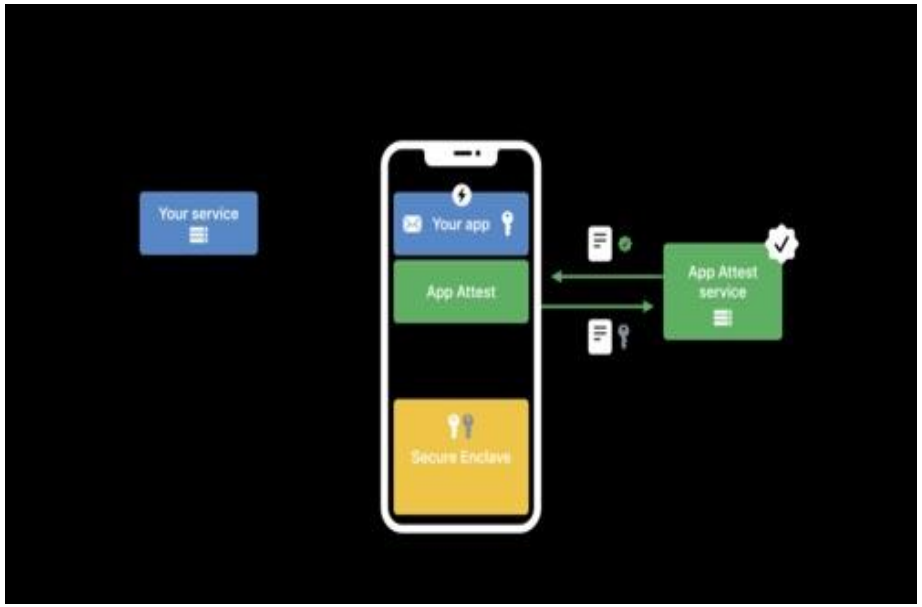
The authenticator retains the private key and shares its public key with the server

Tricking users to bypass 2FA

Type of 2FA	Attack
SMS	Phishing
TOTP	Phishing
Push notifications	Push fatigue

How Does Apps On Apple Platform Use passkeys?

The Power of Apple's App Attest Service



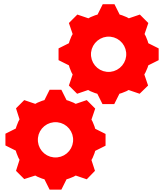
- Built on the principles of zero trust, it emphasizes that no app can self-validate its own security effectively
- This service ensures that rooted devices cannot falsify security checks, protecting the integrity of applications
- It encourages developers to rethink app security by leveraging secure enclave

The Mechanics of App Attest

At the heart lies the '***DCAppAttestService***' class, for creating cryptographic keys

Hardware-based key generation, ensuring that keys belong to valid app instances

With these keys, server requests are signed, establishing trust between the app and backend servers



Two-step validation process

The Role of MSFT Authenticator

MSFT Authenticator acts as a bridge between users and secure interactions

By verifying the attestation results, it ensures that the device is safe and authorized

A unique, one-time challenge is embedded in the attestation process to prevent replay attacks

This challenge is generated by Apple servers and helps create a secure handshake

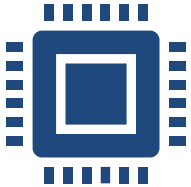
Uses SHA256 for further strengthening security

Unique Hardware Keys for Every User

Unique Crypto Key for Each User

Each key identifier is provided only once

Private Key Storage in Secure Enclave



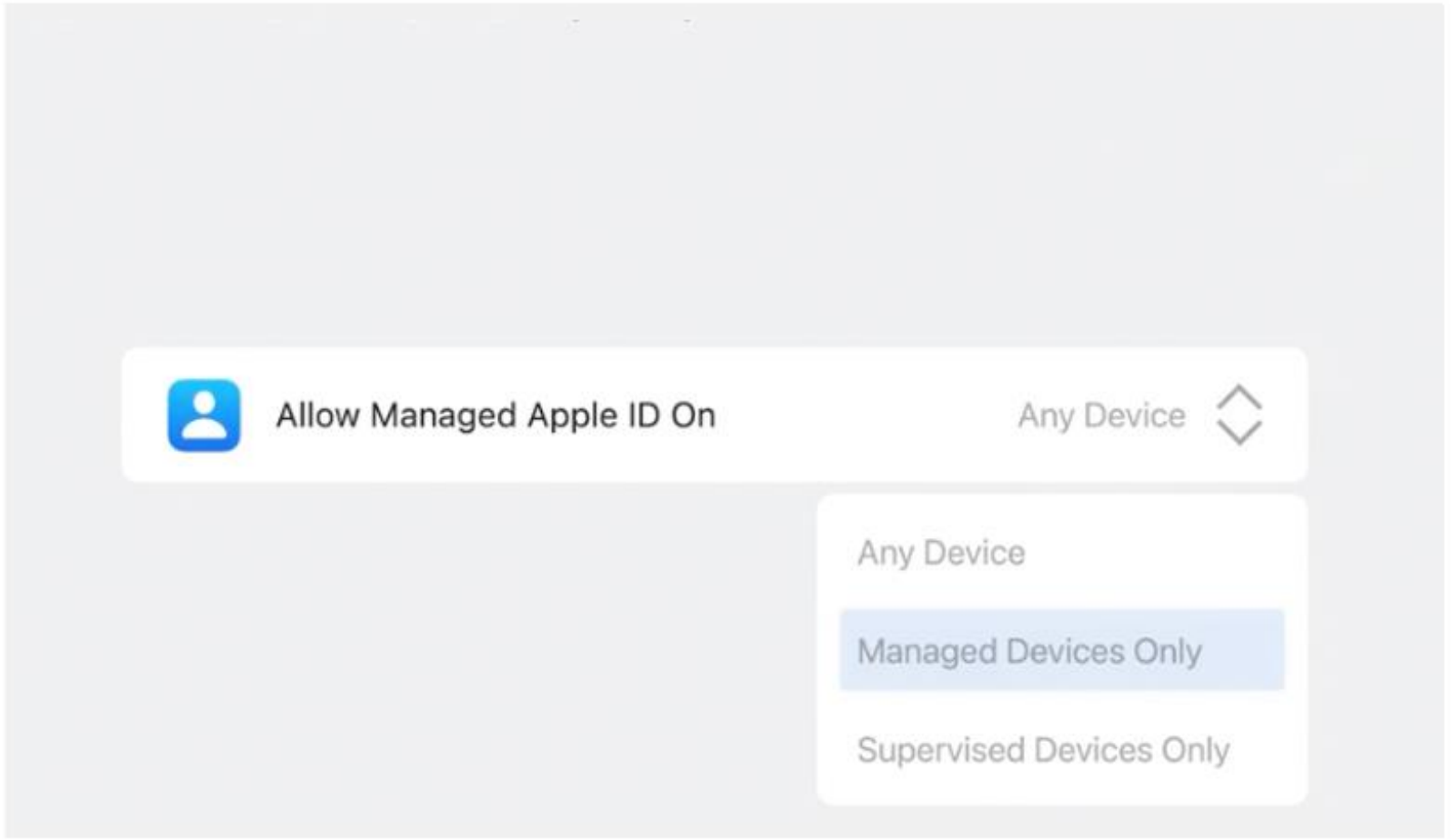
No processes can read or modify this key directly

passkeys @ work

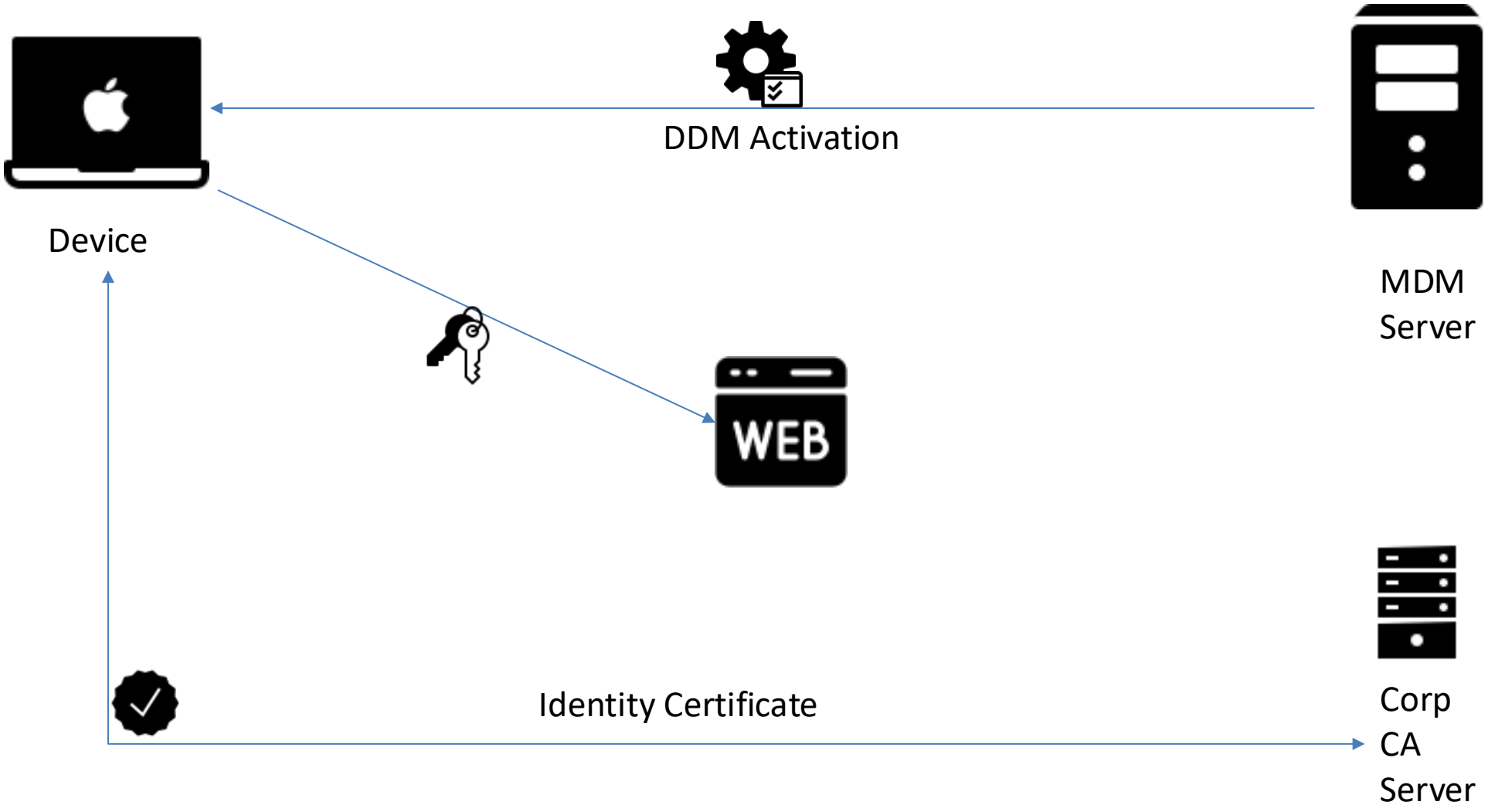
What Do You Need?

- Apple Business Manager
- Managed Apple IDs that uses iCloud Keychain & passkeys
- Sync passkeys ONLY to managed devices
- Store passkeys for work in iCloud Keychain of managed accounts
- Restrict passkey creation ONLY on managed devices
- Turn off passkeys sharing

Managed Apple ID



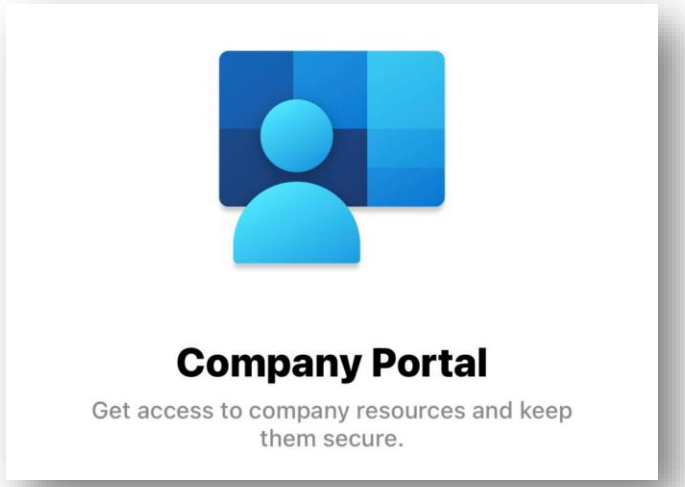
Behind The Scenes

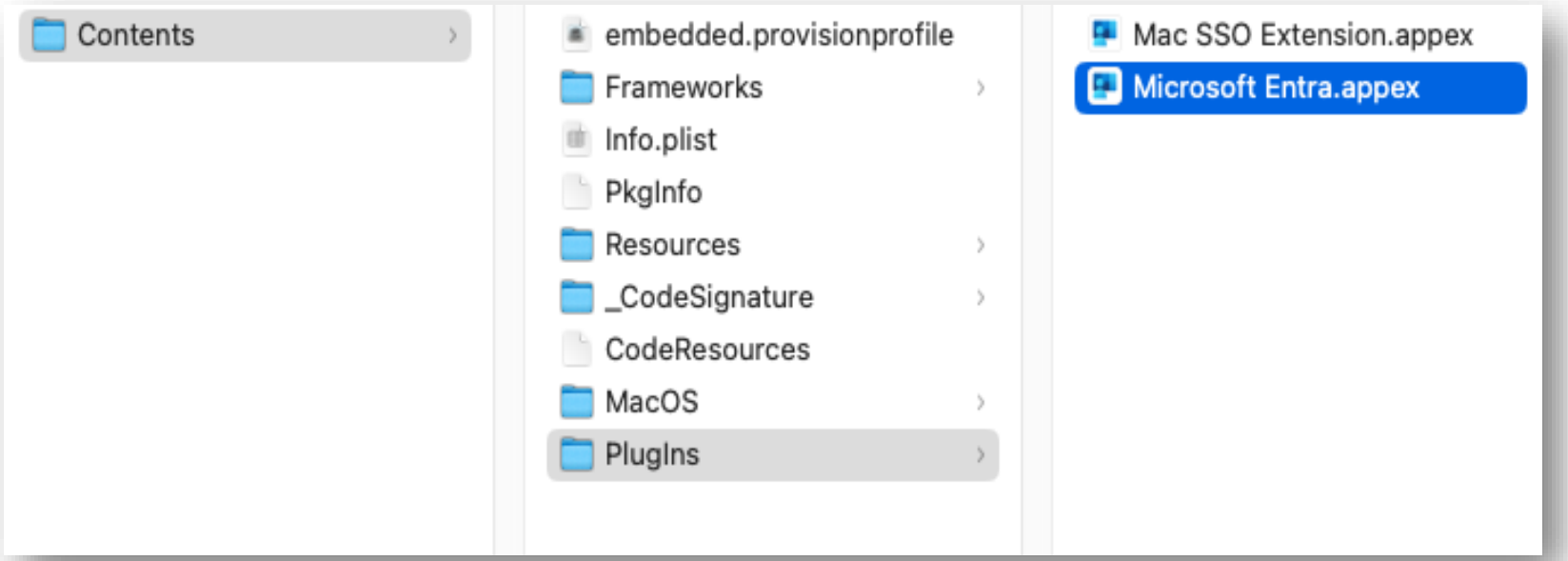

DDM Activation
Device
MDM
Server
WEB
Identity Certificate
Corp
CA
Server

Declarative Device Management


```
{
        "Configuration": "com.apple.configuration.security.passkey.attestation",
         "Type": "com.apple.configuration.security.passkey.attestation",
        "Identifier": "2148C88B-53F8-433C-ADD2-D19CA3B3127A",
         "ServerToken": "E1626D31-1623-41C9-A100-FF8CBEB28CD6",
         "Payload": {
                "AttestationIdentityAssetReference": "E1626D31-1623-41C9-A100-FF8CBEB28CD6",
                 "RelyingParties": [
                        "www.office.com"
                ]
        }
}
```

Microsoft Company Portal





Passkey Implementation

Info.plist

Info.plist

Info.plist ⟩ No Selection

Key	Type	Value
Information Property List	Dictionary	(22 items)
BuildMachineOSBuild	String	22G830
Default localization	String	en
Bundle display name	String	Microsoft Entra
Executable file	String	Microsoft Entra
Bundle identifier	String	com.microsoft.CompanyPortalMac.Mac-Autofill-Extension
InfoDictionary version	String	6.0
Bundle name	String	Microsoft Entra
Bundle OS Type code	String	XPC!
Bundle version string (short)	String	1.0
CFBundleSupportedPlatforms	Array	(1 item)
Bundle version	String	1
DTCompiler	String	com.apple.compilers.llvm.clang.1_0
DTPlatformBuild	String	
DTPlatformName	String	macosx
DTPlatformVersion	String	14.2
DTSDKBuild	String	23C53
DTSDKName	String	macosx14.2
DTXcode	String	1510
DTXcodeBuild	String	15C65
Minimum system version	String	14.0
NSExtension	Dictionary	(3 items)
NSExtensionAttributes	Dictionary	(2 items)
ASCredentialProviderExtensionCapabilities	Dictionary	(2 items)
ProvidesPasskeys	Boolean	YES
ProvidesPasswords	Boolean	NO
ASCredentialProviderExtensionShowsConfigurationUI	Boolean	YES
NSExtensionPointIdentifier	String	com.apple.authentication-services-credential-provider-ui
NSExtensionPrincipalClass	String	Microsoft_Entra.CredentialProviderViewController
Copyright (human-readable)	String	Copyright © 2024 Microsoft. All rights reserved.

Demo

- Secure Enclave
- http://webauthn.me
- iOS MS Authenticator

Thank You!