SESSION ID: LAB3-E02

Authentication on the Move: Challenges for Mobile Web Applications

Johannes Ullrich
Dean of Research
SANS Institute
@johullrich

Jason Lam
Certified Instructor
SANS Institute
@jasonlam_sec
Background

- Strong authentication can be a challenge in the mobile world
  - Small screen real estate
  - Hard to use Keyboards
  - Shoulder Surfing risks

- Mobile native applications may have more capabilities but what about web applications?

- How to effectively authenticate mobile users to web applications
Agenda

- Mobile Web Application Mistakes
- Assisting Users Entering Traditional Passwords
- Improved Authentication Standards for Mobile
- Additional Techniques to Improve Mobile Web Application Authentication Security and Usability
The Basic Authentication Schemes
# Authentication today?

<table>
<thead>
<tr>
<th>Users Authenticate via Username / Password.</th>
<th>Users Recognize Websites using the URL and TLS Certificates.</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Password Policies?</td>
<td>• Phishing?</td>
</tr>
<tr>
<td>- Account Lockout?</td>
<td>• Small URL Bars</td>
</tr>
<tr>
<td>- Credential Stuffing?</td>
<td>• Hard to identify security indicators</td>
</tr>
</tbody>
</table>
How Big Is Your Thumb?
What Phish?
Exercise 1

See https://rsac.authonthemove.com/exercise1 for instructions

Goal: Identify shortcomings of traditional username and password authentication for mobile devices and learn how to better integrate with mobile web browsers to improve authentication usability.
RSAC Conference 2020

Improved Authentication for Mobile
Password Stores/Safes

- OS platforms or 3rd party software offer capabilities to store password for you

- Benefits
  - Recognize the remote site, reduce phishing risk
  - More inclined to use complex (generated) passwords
  - High user acceptance level

- Master passphrase and OS password/biometrics protects the vault
Authenticator

- App based TOTP token system
  - RFC6238 based token system (or HOTP RFC4226)
- Website generate 80 bit of secret key which can be in form of QR code
  - Alternatively, can be manually entered into the phone
- Generate time based token based on the secret
- To cloud or not to cloud?
  - Some services like Authy send your keys to the cloud
SMS/Voice

- Popular form of authentication – ease of use
- Phone call or SMS a "token" to the user
  - The token needs to be generated securely
  - User needs to type the code back on the web page
- Pitfalls
  - SIM-jacking/SIM swapping possible
  - Social Engineering bundled with phishing

607788 is your Grab Activation Code (GAC). It expires in 2 minutes. Do not share it with anyone.

607789 is your Grab Activation Code (GAC). It expires in 2 minutes. Do not share it with anyone.
SMS New Style/standard

- Emerging standard from WebKit developers
- Common standard to allow the phone automatically submit the code/token back to the site
- In recent version of iOS, there is ability to copy the code automatically

12345 is the code for authonthemove
12345 is the code for rsac.authonthemove.com
12345 is the code for authonthemove
@rsac.authonthemove.com #12345
https://rsac.authonthemove.com
Submit 12345 to HTML form field
Mobile App Push Authentication

- Using an already authenticated native mobile app to push notification to user
- User then explicitly consent to the authentication
Good/Bad of App Push Authentication

- **Good**
  - Excellent user acceptance – what's not to like?
  - Low cost for the Web site
  - Lots of vendors to choose from

- **Bad**
  - Users often accidentally approve fraudulent request
  - Initial Setup factor - App download and initial key inject
  - Many security dependencies – App store, Device, Vendor...
Exercise 2

- OS/Browser integrated password vault
- 3rd party password vault – LastPass
  1. Save password
  2. AutoFill
  3. Tie in with biometrics
Exercise 2

See [https://rsac.authonthemove.com/exercise2](https://rsac.authonthemove.com/exercise2) for instructions

Goal: Learn how to implement and use a one-time password authentications (TOTP).
Advanced Mobile Authentication
Fast Identity Online (FIDO) is behind the FIDO2 standard
- Consists of WebAuthn and CTAP (Client to Authenticator Protocol)

WebAuthn is a W3C standard that defines browser to server communication for non-password-based authentication
- Uses asymmetric cryptographic authentication

CTAP standardizes the communication between the authenticators and the browsers
- Can be physical or software token or gesture/biometric recognition

Authenticator is often used with a PIN to add extra security
WebAuthn Registration

User registration please

JavaScript: Create a public key for me, here are a bunch of options

Create public key pair

Here is my public key
WebAuthn Authentication

Here is my username, I want to sign in

Please sign this challenge so I know it’s you

Signing the challenge with the Private Key

This is the signature, please validate
Do I Still Need a Password?

- You can still add a PIN/Passphrase to the authentication
  - Extra layer of security, may not buy you much given the sad state of password security
- Can even include the use of push app authentication if desired
- Can blend in biometrics to further improve security
- Did you know? If you have a recent Android phone, you already have a FIDO2 key
NFC Factor with Mobile

- All major mobile OSes allow CTAP interface with NFC/Bluetooth enabled keys
- When prompted, put token close to phone and activate the token
  - Seamless experience that's phish resilient
"Password" Reset & Backup

- What if you lost your token?
- There is no official way to "recover" as it is not needed
- Have a backup token that is stored at a safe location
  - Yes, that's double the investment $
- Use the backup token to login and then disable the lost token
- Website operators – Remind your users of the 2 tokens best practice
JavaScript APIs (FIDO2/WebAuthn)

navigator.credentials.create
  (PublicKeyCredentialCreationOptions)

Token will create new credentials and pass it to the browser for registration with the web site. Client will receive among other parameters:

- Public Key and parameters (type, algorithm)
- User ID
- Challenge
JavaScript APIs (FIDO2/WebAuthn)

`navigator.credentials.get(CredentialsRequestOptions)`

Used to “Login”. Requests existing credentials from Token. “Mediation” option determines if there is a user prompt.

Important:
- Registration and Login need to use the same origin.
- HTTPS Required
Exercise 3

- Adding WebAuthn to a web application
- See https://rsac.authonthemove.com/exercise3 for instructions
Other Options / Defense in Depth
Modern JavaScript APIs

- Modern JavaScript APIs (“HTML5”) provide access to various hardware sensors
- Not all of them are suitable for authentication, but they can be helpful to make authentication more user-friendly, or provide additional validation
- Examples: Camera, Microphone, GPS, Canvas
Modern JavaScript APIs

- **Camera/Microphone:**
  + Simple bio metrics.
  + JavaScript APIs exist for facial recognition
    - Quality varies widely. Not as good as built in facial recognition systems

- **GPS**
  + Easy to use and can be very accurate
    - Easy to spoof

- **Canvas**
  - Can be used for more graphical login schemes
Exercise 4

See https://rsac.authonthemove.com/exercise4 for instructions

Goal:

Implement an improved authentication experience taking advantage of modern JavaScript APIs.
Apply What You Have Learned Today

- Next week you should:
  - Review the authentication of mobile applications in your organization

- In the first three months following this presentation you should:
  - Plan out the roadmap to migrate away from password based authentication
  - Determine the risk based approach to authentication (more != better)

- Within six months you should:
  - Start the adoption of passwordless or push based authentication into your applications
  - Educate users on the benefit and best practices for these mechanisms