Device Authorization Grant and Social Engineering Exploits

Pieter Kasselman

IETF 113:
Date: 24 March 2022

> The only way of discovering the limits of the possible is to venture a little way past them into the impossible.

Arthur C. Clarke

Content

- Device Authorization Grant (RFC 8682)
- Social engineering exploit pattern
- RFC 8682 Security Considerations
- Things we could do
- What should we do?

Device Authorization Grant

Why Device Authorization Grant (RFC 8682)?

OAuth 2.0 Device Authorization Grant

Abstract

The OAuth 2.0 device authorization grant is designed for Internet-connected devices that either lack a browser to perform a user-agent-based authorization or are input constrained to the extent that requiring the user to input text in order to authenticate during the authorization flow is impractical. It enables OAuth clients on such devices (like smart TVs, media consoles, digital picture frames, and printers) to obtain user authorization to access protected resources by using a user agent on a separate device.

Why Device Authorization Grant (RFC 8682)?

OAuth 2.0 Device Authorization Grant

Abstract

The OAuth 2.0 device authorization grant is designed for Internet-connected devices that either lack a browser to perform a user-agent-based authorization or are input constrained to the extent that requiring the user to input text in order to authenticate during the authorization flow is impractical. It enables OAuth clients on such devices (like smart TVs, media consoles, digital picture frames, and printers) to obtain user authorization to access protected resources by using a user agent on a separate device.

Device Authorization Grant

Authorization Server

Endpoint

Consumption Device
(Initiate Session)

Authorization Device
(Authenticate/Authorize)

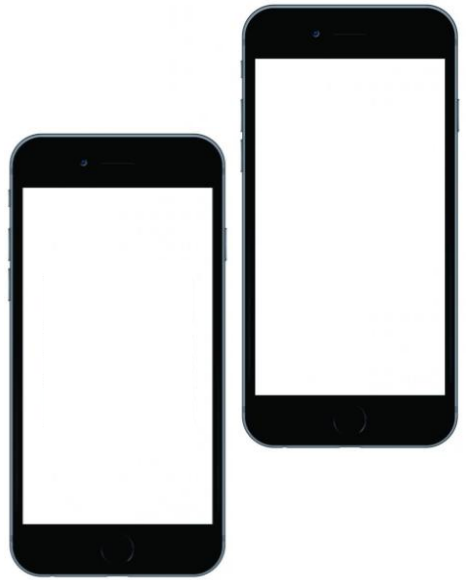
Device Authorization Grant (1 of 4)

Authorization Server

Endpoint

1. Get and Display User Code, Device Code, Verification URI

Consumption Device (Initiate Session)



Authorization Device (Authenticate/Authorize)

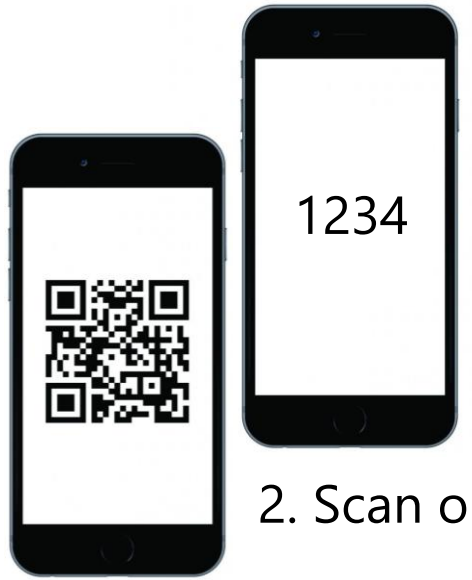
Device Authorization Grant (2 of 4)

Authorization Server

Endpoint

1. Get and Display User Code, Device Code, Verification URI

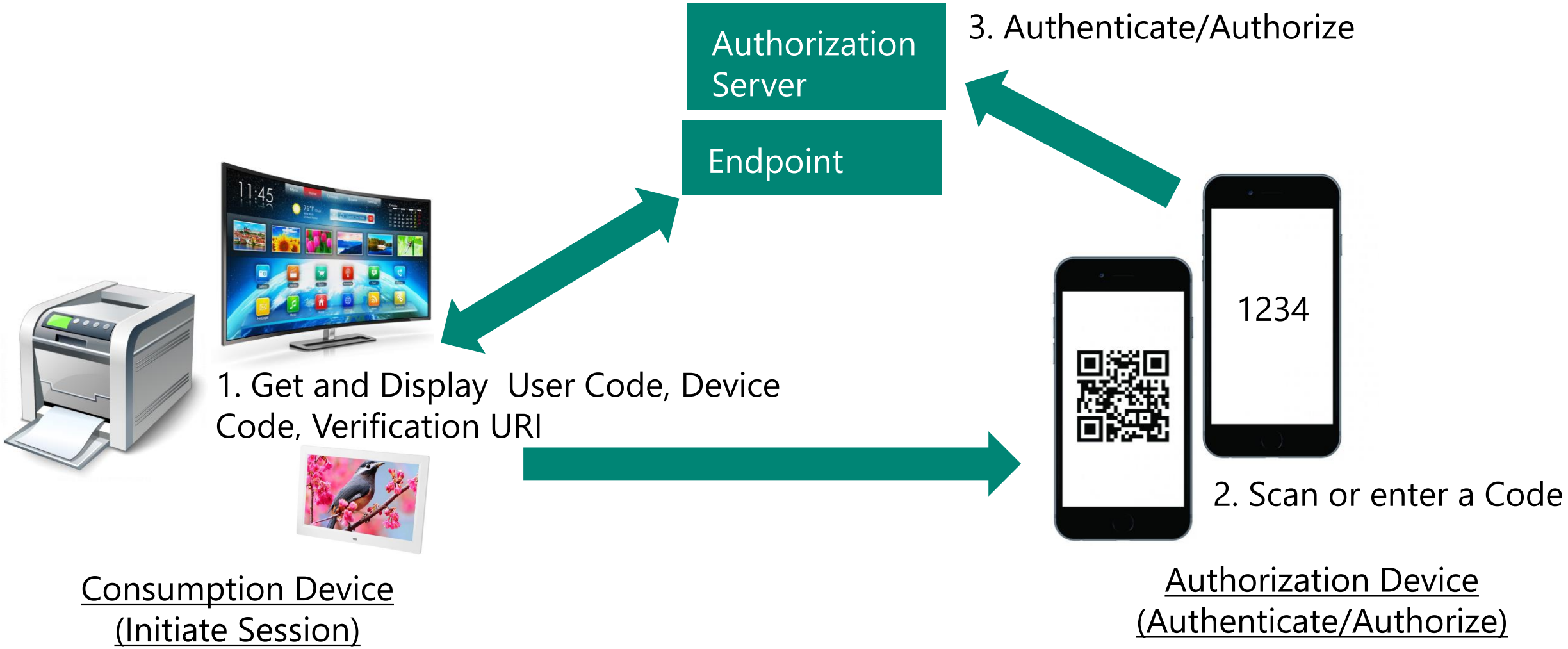
Consumption Device (Initiate Session)



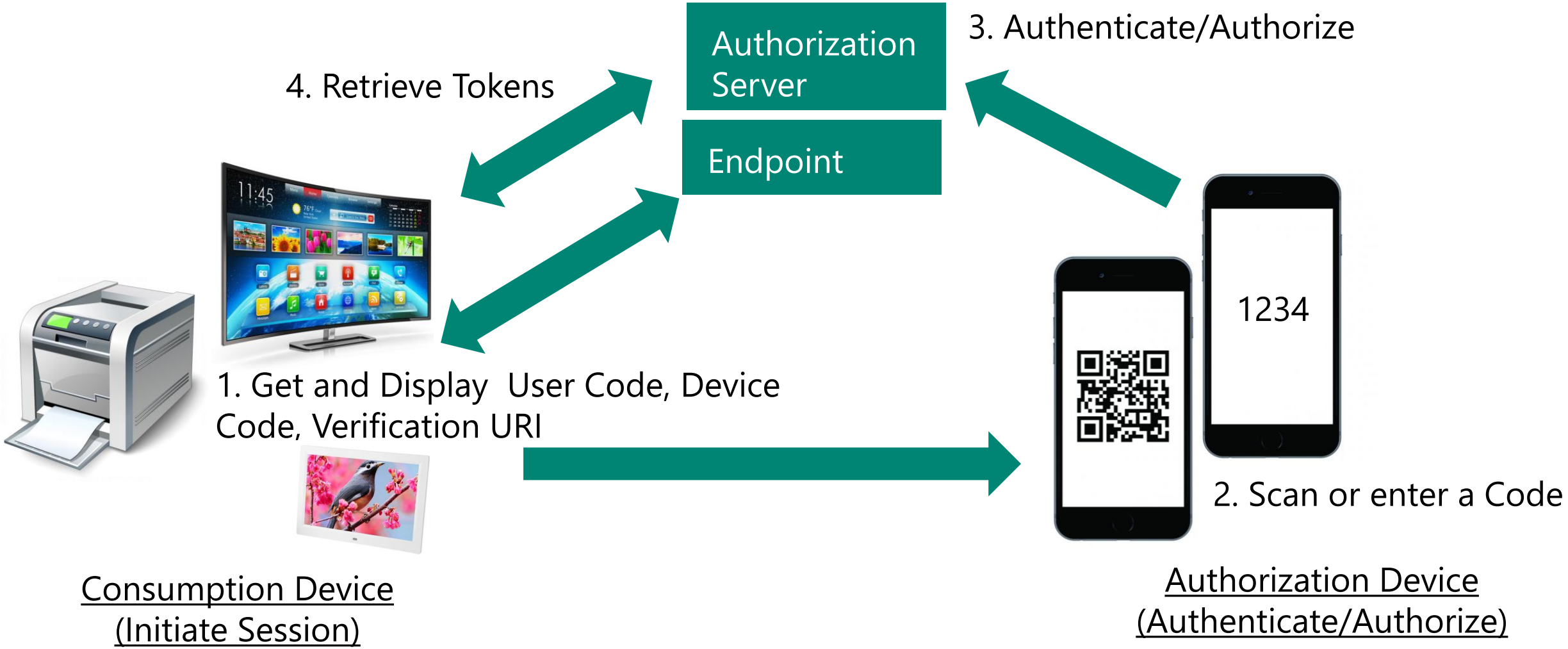
2. Scan or enter a Code

Authorization Device (Authenticate/Authorize)

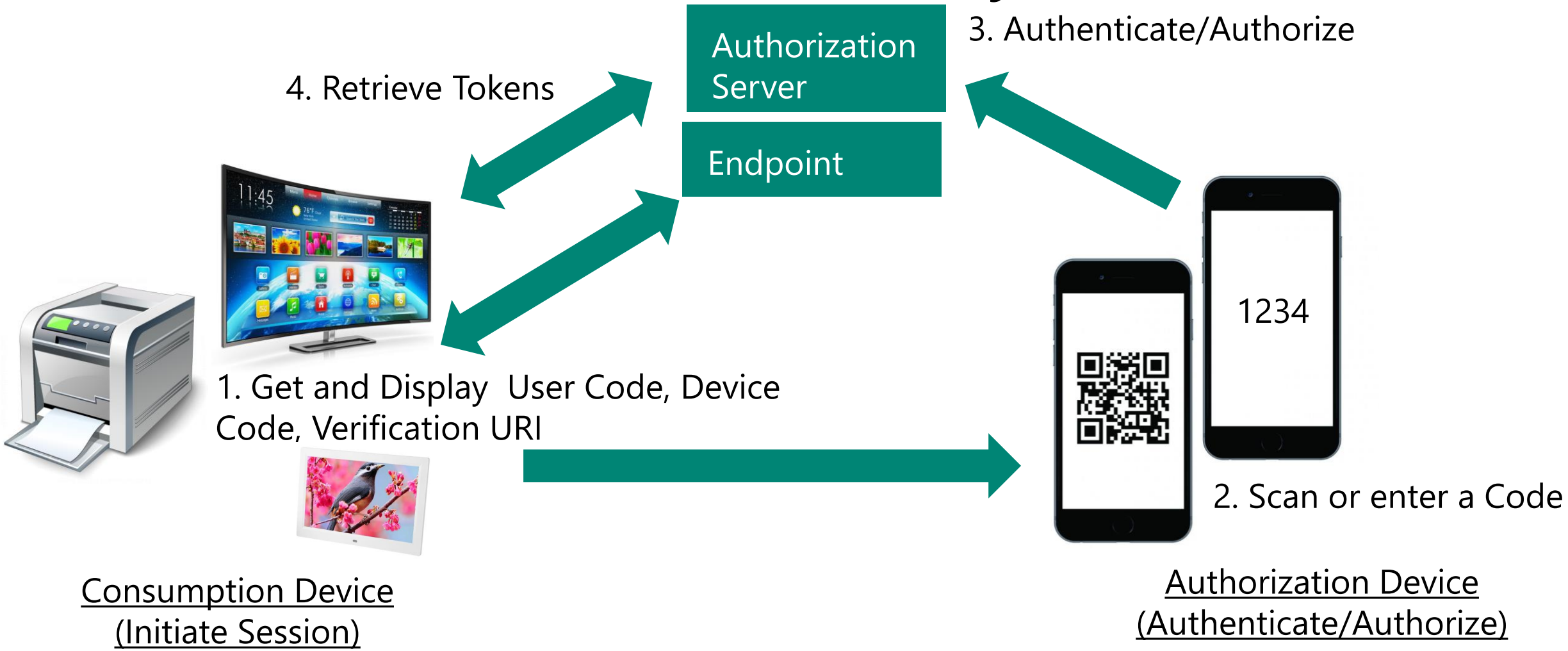
Device Authorization Grant (3 of 4)



Device Authorization Grant (4 of 4)



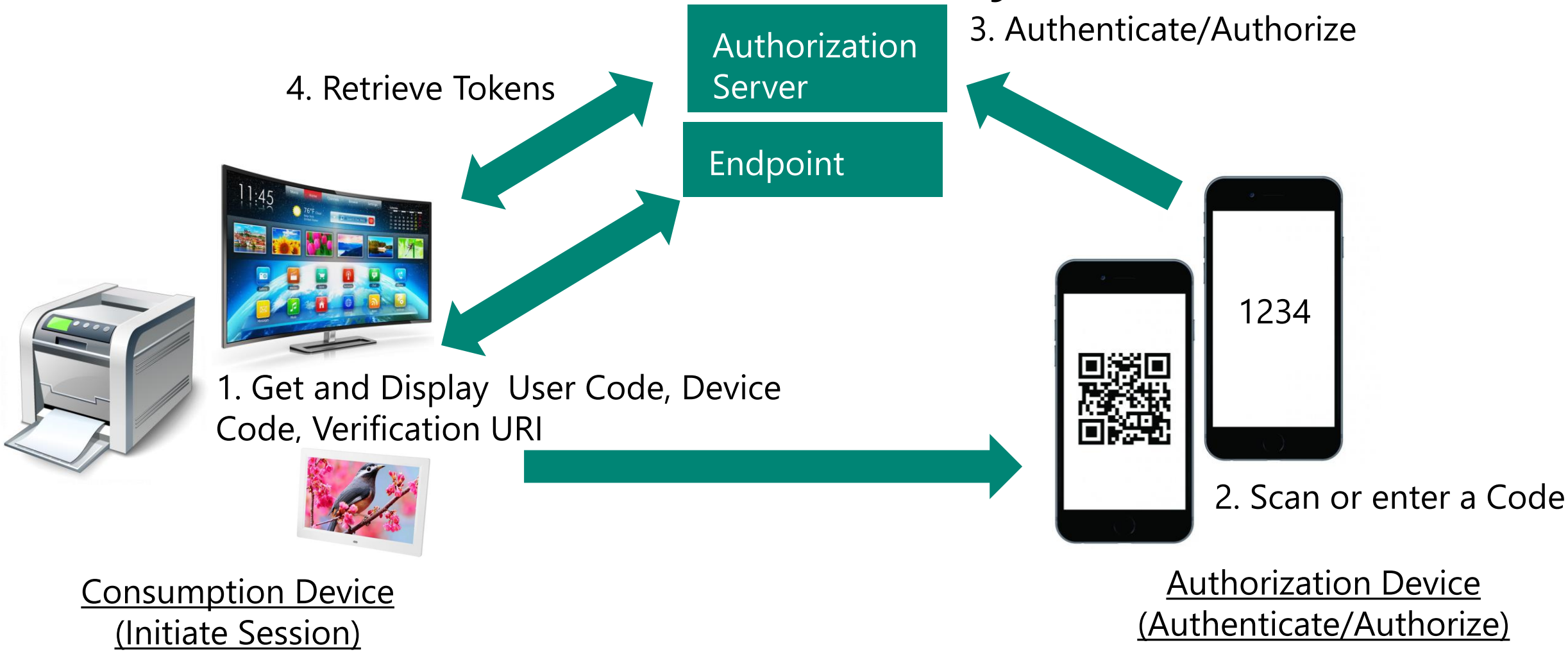
Device Authorization Grant: Summary



In a nutshell

- Initiate session on one device
- Authorize on second device

Device Authorization Grant: Summary



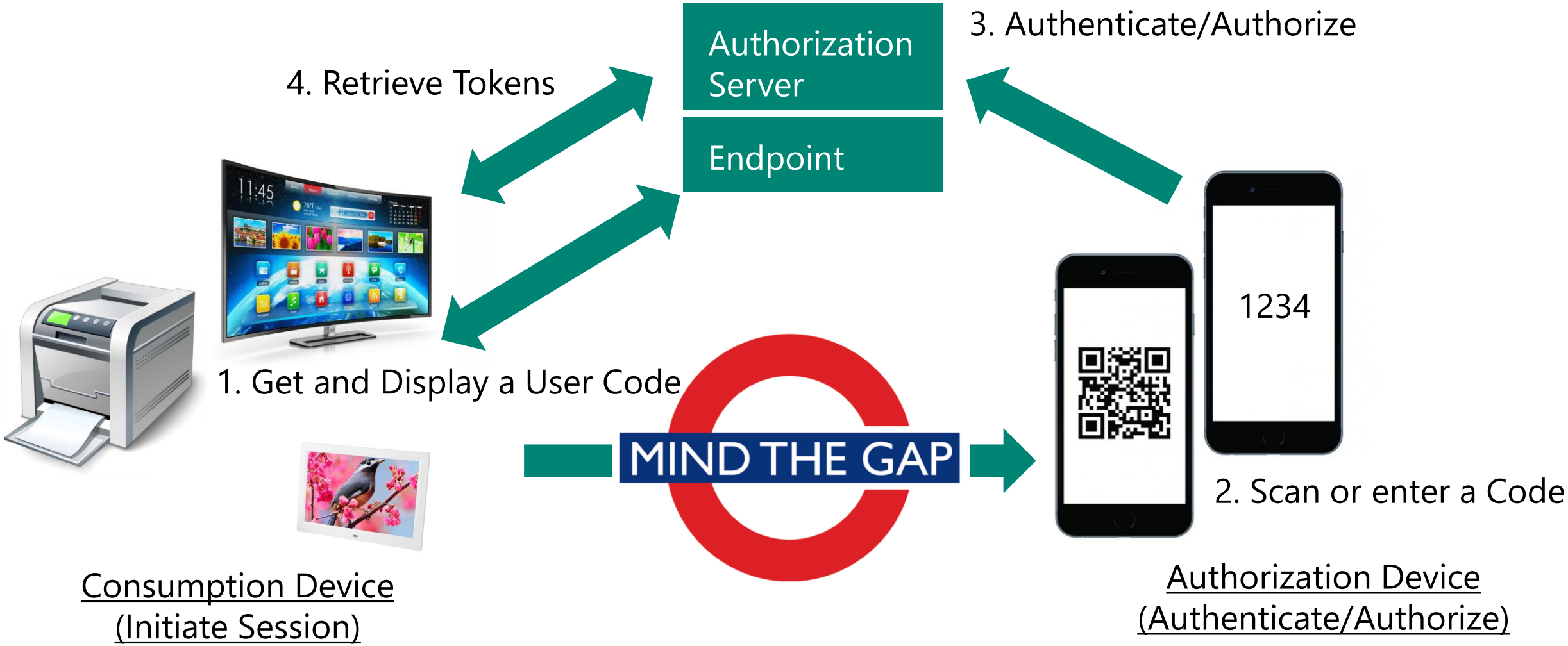
In a nutshell

- Initiate session on one device
- Authorize on second device

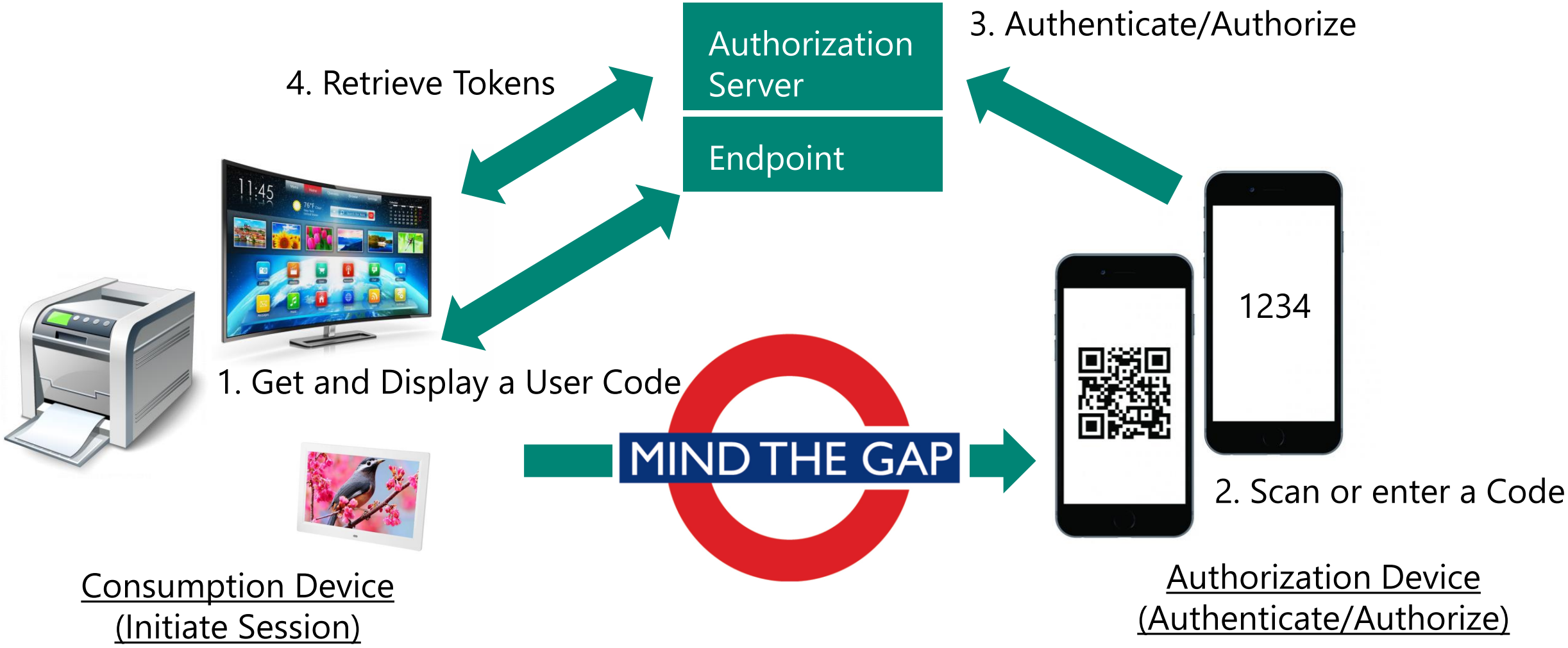
Benefits

- Authorization for devices with limited input capabilities
- Strong authentication with a personally trusted device

But… Mind the Gap



But… Mind the Gap



Session Transfer

1. No protocol to establish trust relationship between Consumption Device and Authorization Device
2. Push responsibility for trust decision to the user.

Device Authorization Grant and Social Engineering

Social Engineering Exploit

Authorization Server

Endpoint

1. Get a Code

Attacker Controlled Device
(Initiate Session)

Authorization Device
(Authenticate/Authorize)

Social Engineering Exploit (1 of 5)

Attacker Controlled Device
(Initiate Session)

Authorization Server

Endpoint

1. Get a Code

Authorization Device
(Authenticate/Authorize)

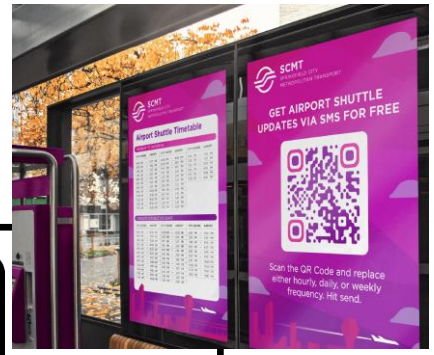
Social Engineering Exploit (2 of 5)

Attacker Controlled Device
(Initiate Session)

1. Get a Code

Authorization Server

Endpoint



Click here to sync your messages

2. Change Context

Authorization Device
(Authenticate/Authorize)

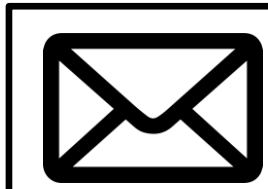
Social Engineering Exploit (3 of 5)

Attacker Controlled Device
(Initiate Session)

1. Get a Code

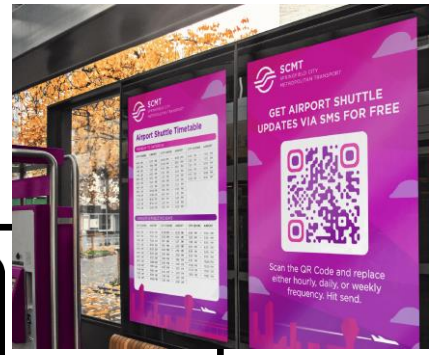
Authorization Server

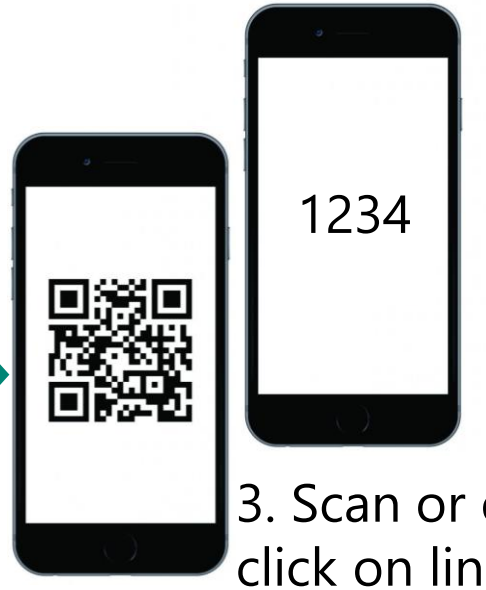
Endpoint



Click here to sync your messages

2. Change Context





3. Scan or enter a Code, click on link

Authorization Device
(Authenticate/Authorize)

Social Engineering Exploit (4 of 5)

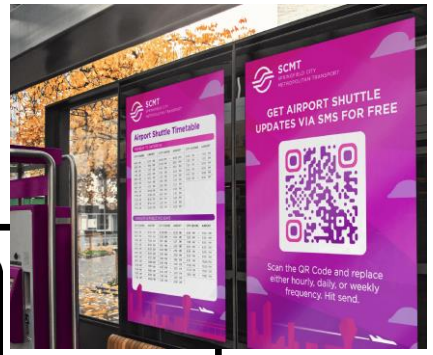
Attacker Controlled Device
(Initiate Session)

1. Get a Code

Authorization Server

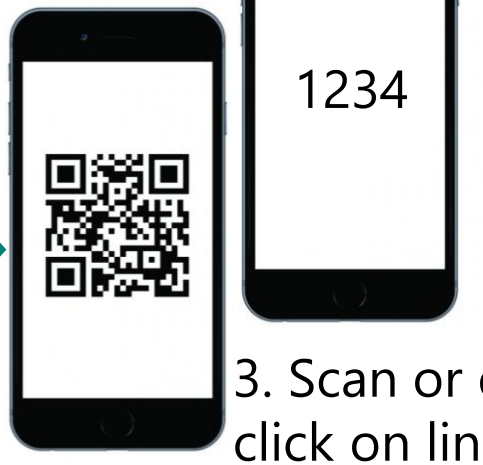
Endpoint

4. Authenticate/Authorize



Click here to sync your messages

2. Change Context



3. Scan or enter a Code, click on link

Authorization Device
(Authenticate/Authorize)

Social Engineering Exploit (5 of 5)

5. Retrieve Tokens

Authorization Server

Endpoint

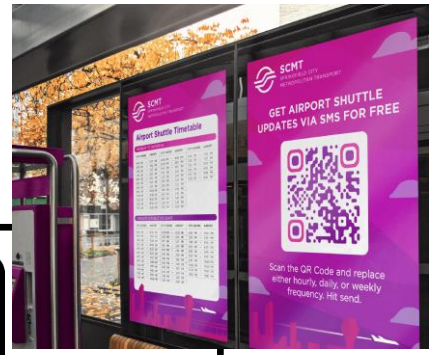
4. Authenticate/Authorize

1. Get a Code

Attacker Controlled Device (Initiate Session)

Click here to sync your messages

2. Change Context



3. Scan or enter a Code, click on link

Authorization Device (Authenticate/Authorize)

Social Engineering Exploit

5. Retrieve Tokens

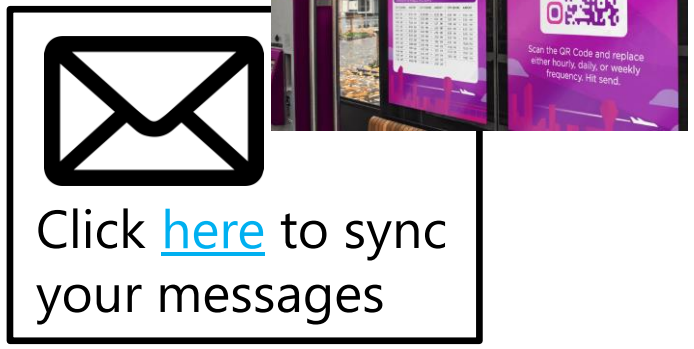
Authorization Server

Endpoint

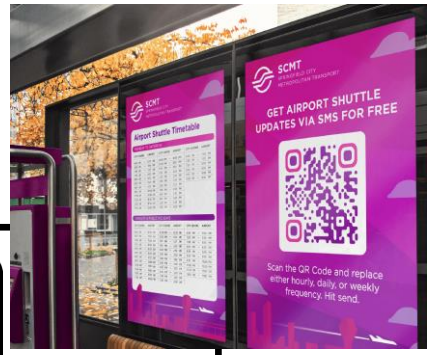
4. Authenticate/Authorize

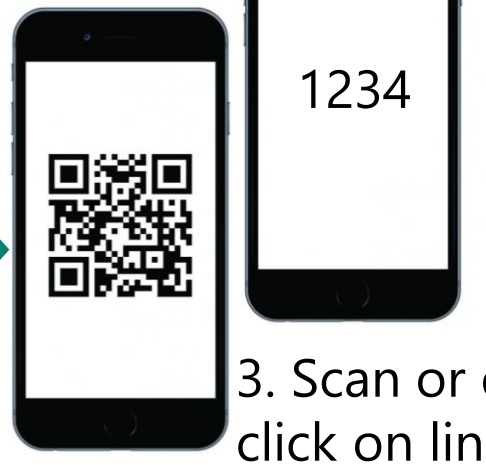
1. Get a Code

Attacker Controlled Device
(Initiate Session)



2. Change Context





3. Scan or enter a Code, click on link

Authorization Device
(Authenticate/Authorize)

MITM Summary

1. Initiate the session, retrieve user code
2. Use social engineering (Phishing etc) to change context and persuade user to authorize session.
3. Relevant to protocols with an air-gap.

Examples of Switching Context

Send | From: DevOps@msftsec.onmicrosoft.com | To | Cc | Subject: Microsoft Security - Required Action

Required Action

Your device is being logged out of Microsoft Office 365. Please use the following URL and device code to re-link your account.

Your code is: <REPLACE-WITH-DEVCODE-FROM-TOKENTACTICS>

https://microsoft.com/devicelogin

Sincerely,
Microsoft Device Security Team

Microsoft Corporation | One Microsoft Way Redmond, WA 98052-6399

This message was sent from an unmonitored email address. Please do not reply to this message.

Privacy | Legal

S Support
Tue 7/20/2021 9:55 AM
To:

Hi [redacted]

Your company, [redacted] updated the Mobile Enrollment policy for [redacted]

Please copy the following link to your web browser to review your policy:
https://microsoft.com/devicelogin

Use the following secure code to access: E52XFKNA4

Thanks,
The Microsoft account team

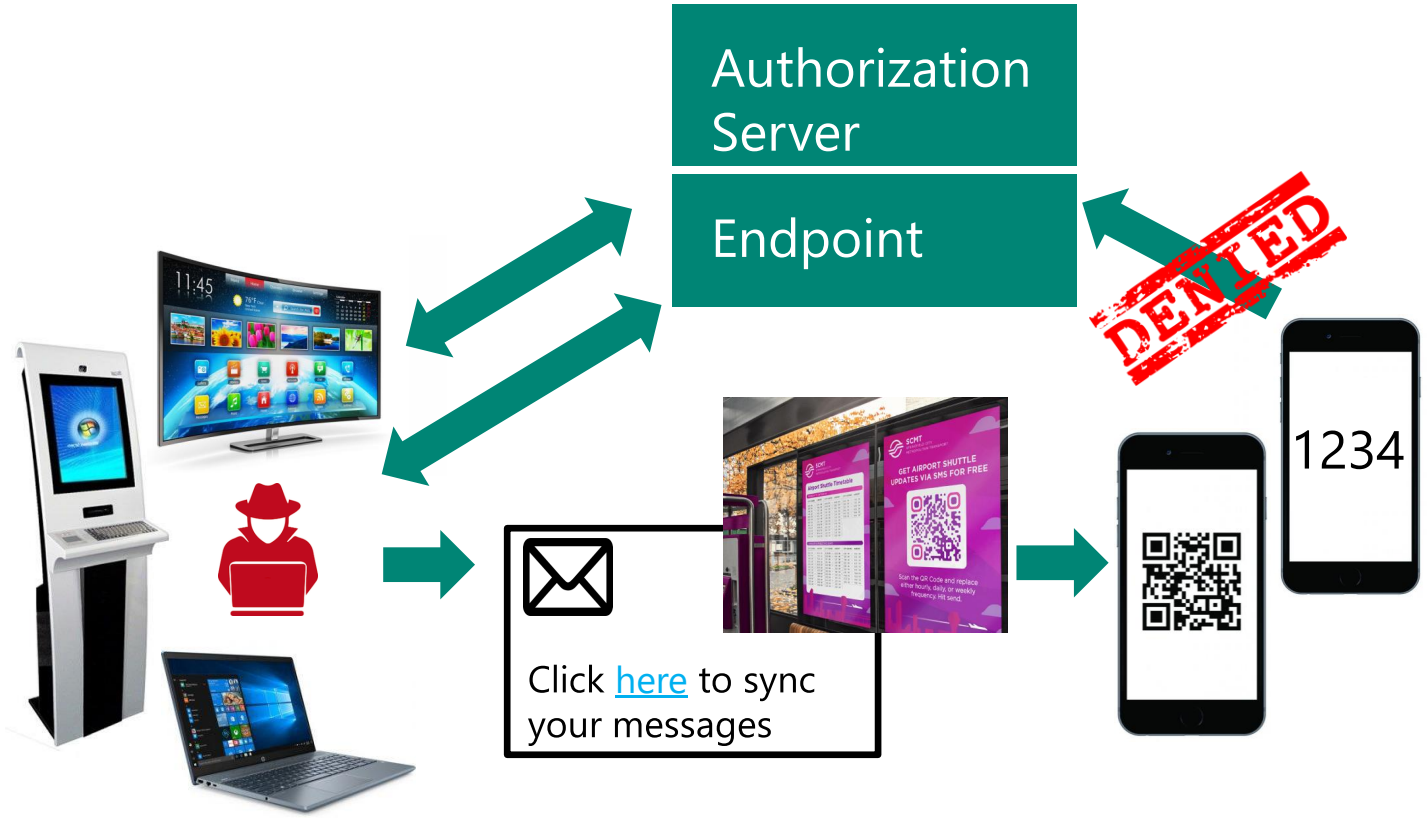
Reply | Forward

DD Don Director <dond@something.com>
Tue 10/13/2020 10:28 PM
To: William Victim

Hi!
Here is the link to the document. Use the following code to access:
CGWSDVSVL.

Reply | Forward

Homo Securitus to the Rescue

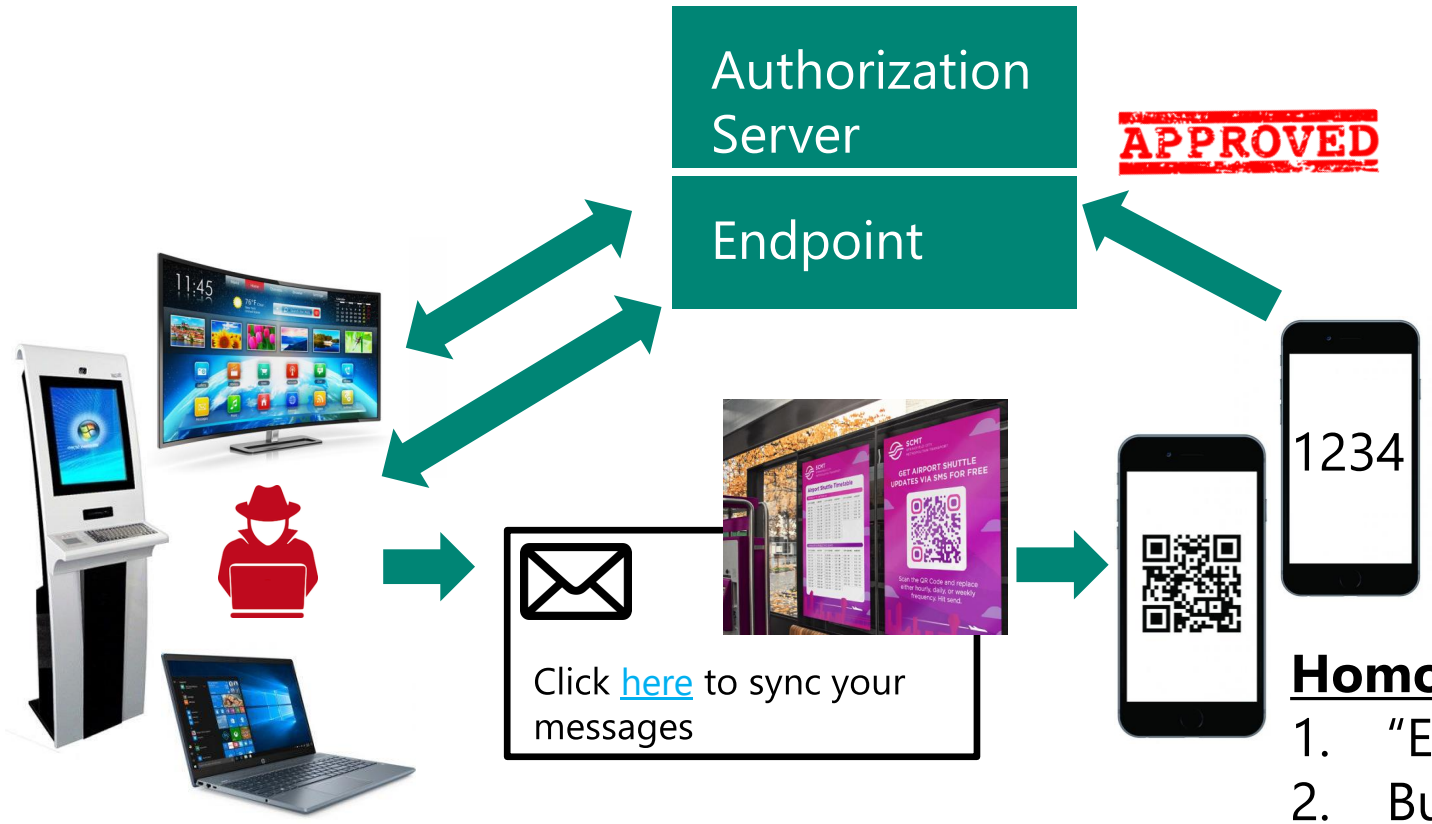


Homo Securitus

1. A security expert
2. Knows how the protocol should work
3. Detects a social engineering attempt
4. Is laser focused on current context
5. Foolproof mitigation for cross device flows

But is a rare species….

But what about Homo Sapiens?



Homo Sapiens

1. "Expertise elsewhere" - not a security expert
2. Busy and in a rush, needs to get things done
3. Worries about breaking things
4. Wants to help

Needs to make fewer decision,
Needs help to make better decisions
Needs protection even if a bad decision is made

What does RFC 8628 say about social engineering?

What does RFC 8628 say about Phishing?

5.4. Remote Phishing

It is possible for the device flow to be initiated on a device in an attacker's possession. For example, an attacker might send an email instructing the target user to visit the verification URL and enter the user code. To mitigate such an attack, it is RECOMMENDED to inform the user that they are authorizing a device during the user-interaction step (see Section 3.3) and to confirm that the device is in their possession. The authorization server SHOULD display information about the device so that the user could notice if a software client was attempting to impersonate a hardware device.

For authorization servers that support the "verification_uri_complete" optimization discussed in Section 3.3.1, it is particularly important to confirm that the device is in the user's possession, as the user no longer has to type in the code being displayed on the device manually. One suggestion is to display the code during the authorization flow and ask the user to verify that the same code is currently being displayed on the device they are setting up.

The user code needs to have a long enough lifetime to be useable (allowing the user to retrieve their secondary device, navigate to the verification URI, log in, etc.) but should be sufficiently short to limit the usability of a code obtained for phishing. This doesn't prevent a phisher from presenting a fresh token, particularly if they are interacting with the user in real time, but it does limit the viability of codes sent over email or text message.

Better Decision
Making More UI about the device

Better Decision
A bit more UI for the user to follow

Protect against bad decisions
Limited lifetime

RFC 8628 also says….

5.7. Non-Visual Code Transmission

There is no requirement that the user code be displayed by the device visually. Other methods of one-way communication can potentially be used, such as text-to-speech audio or Bluetooth Low Energy. To mitigate an attack in which a malicious user can bootstrap their credentials on a device not in their control, it is RECOMMENDED that any chosen communication channel only be accessible by people in close proximity, for example, users who can see or hear the device.

Better Decision

Proximity should apply to visuall one way communication as well as well

One more thing….

There are two devices

Malicious Authorization Server

Silent on mechanisms for establishing trust (or verify trust)

5.3. Device Trustworthiness

Unlike other native application OAuth 2.0 flows, the device requesting the authorization is not the same as the device from which the user grants access. Thus, signals from the approving user's session and device are not always relevant to the trustworthiness of the client device.

Note that if an authorization server used with this flow is malicious, then it could perform a man-in-the-middle attack on the backchannel flow to another authorization server. In this scenario, the man-in-the-middle is not completely hidden from sight, as the end user would end up on the authorization page of the wrong service, giving them an opportunity to notice that the URL in the browser's address bar is wrong. For this to be possible, the device manufacturer must either be the attacker and shipping a device intended to perform the man-in-the-middle attack, or be using an authorization server that is controlled by an attacker, possibly because the attacker compromised the authorization server used by the device. In part, the person purchasing the device is counting on the manufacturer and its business partners to be trustworthy.

What else could we do?

Examples of Defence in depth mitigations

Proximity: Use location in the absence of physical connectivity for proximity

- Was the user code used in the same building, town, region, country or continent where it was issued?
- Add geo-location information in the user interface

Content Filtering: Monitor communications channels for valid user codes being sent to users

- Spam filters, risk management systems etc.

Add Meta Data to User Code: Add additional information in a QR code with the user code

- Location information, signed QR codes, etc – verified by smart wallets.

Token Binding: Use DPoP to prevent tokens obtained through social engineering to be exfiltrated

- Limits impact of a compromised device

Trusted Devices: Only allow Device Authorization Grant from trusted devices

- Pre-registered devices, device attestation

Secure QR Codes : OASIS Electronic Secure Authentication (ESAT) Technical Committee

- "The work in this TC aims to remedy the risks associated with the use of QR code for strong authentication."

More…

Other protocols: Relationships and ranking

- How can we help implementors select the right protocols
- “Cross Device” protocol examples:
  - Client Initiated Backchannel Authentication (CIBA)
  - Self-Issued OpenID Provider (SIOP)
  - Anywhere a QR code is scanned and authorization is transferred
- Matching\Ranking protocols to use cases and risk profiles

What should we do?

How should we respond?

A. Do nothing
B. Provide additional Implementation Guidance
   - Threat models
   - Risk assessment frameworks
   - User Experience guidance
   - Defence in depth mitigations
   - Secure profiles
   - Protocol selection/recommendation guides
C. New protocols/New Paradigms
   A. CIBA, Session transfer protocols, etc
D. Any other options?