



# Experimental Security Analysis of the App Model in Business Collaboration Platforms

---

Yunang Chen\*, Yue Gao\*, Nick Ceccio,  
Rahul Chatterjee, Kassem Fawaz, Earlence Fernandes<sup>+</sup>

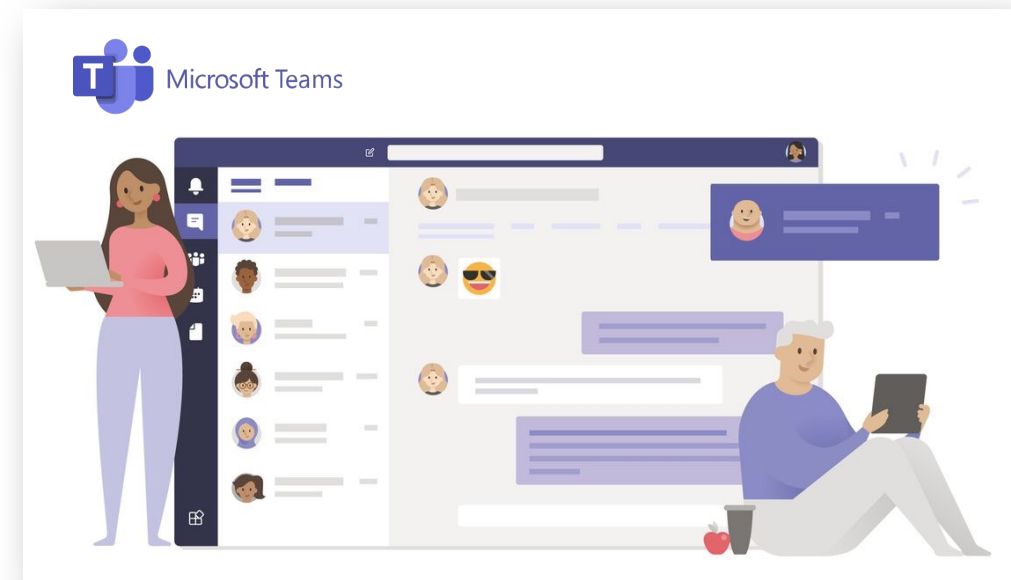
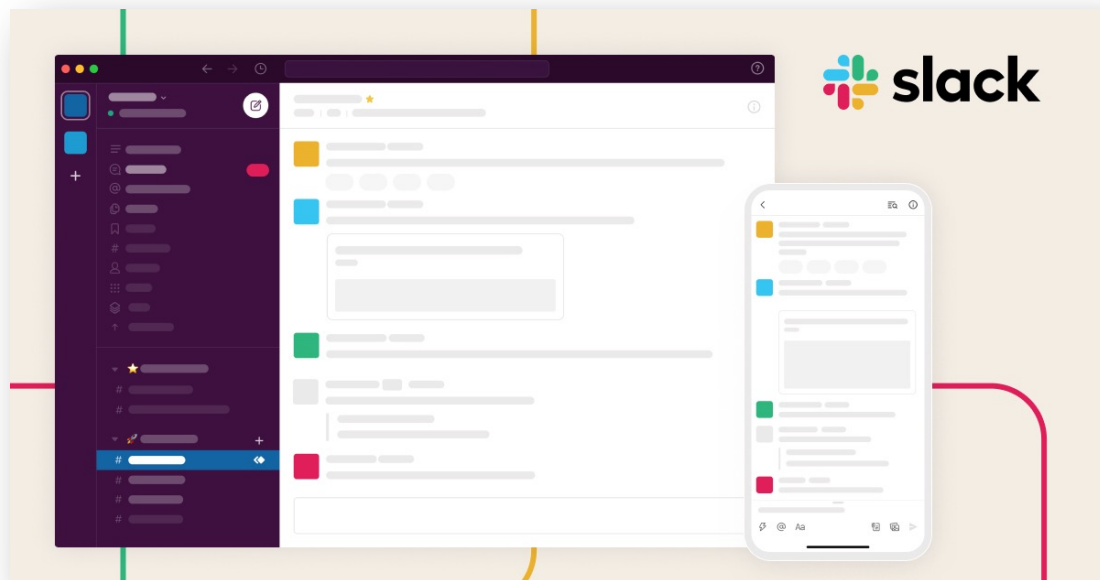
University of Wisconsin–Madison, University of California San Diego

\* *Equal contribution*

<sup>+</sup> *Work done while at UW-Madison*

# Business Collaboration Platforms (BCPs)

- Productivity & Team Collaboration
- Third-Party Integrations (Apps)



# BCPs Have Become A Hub for Sensitive Resources

- Zoom Calls
- DropBox File Sharing
- Email Forwarding
- Code Repository Management
- ...

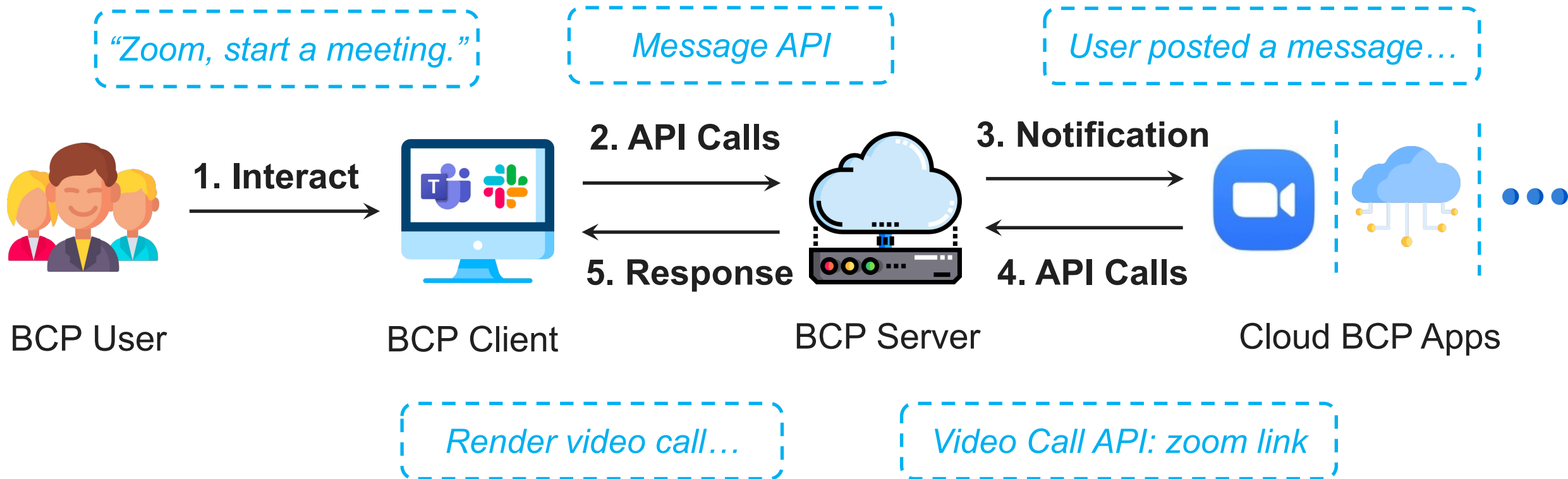
**What if the apps are malicious?**

**Can BCPs enforce security correctly?**

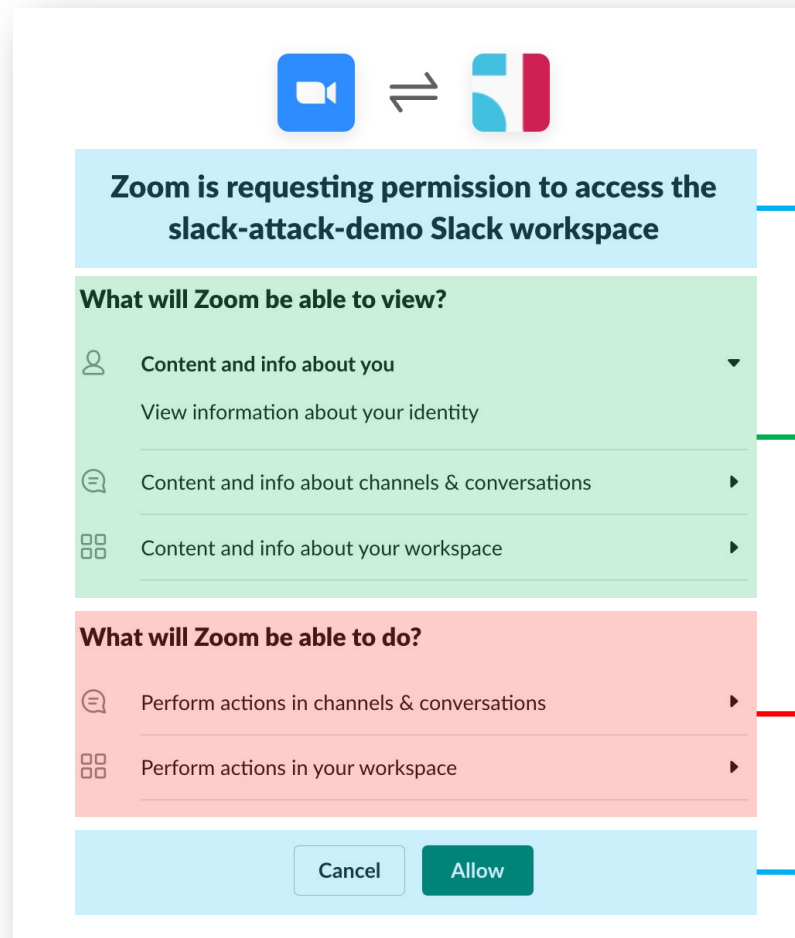




# Background: App Workflow



# Background: App Installation



1. App Requests Permissions

2. Read Permission Scopes

- *Read user identity*
- *Read public messages*

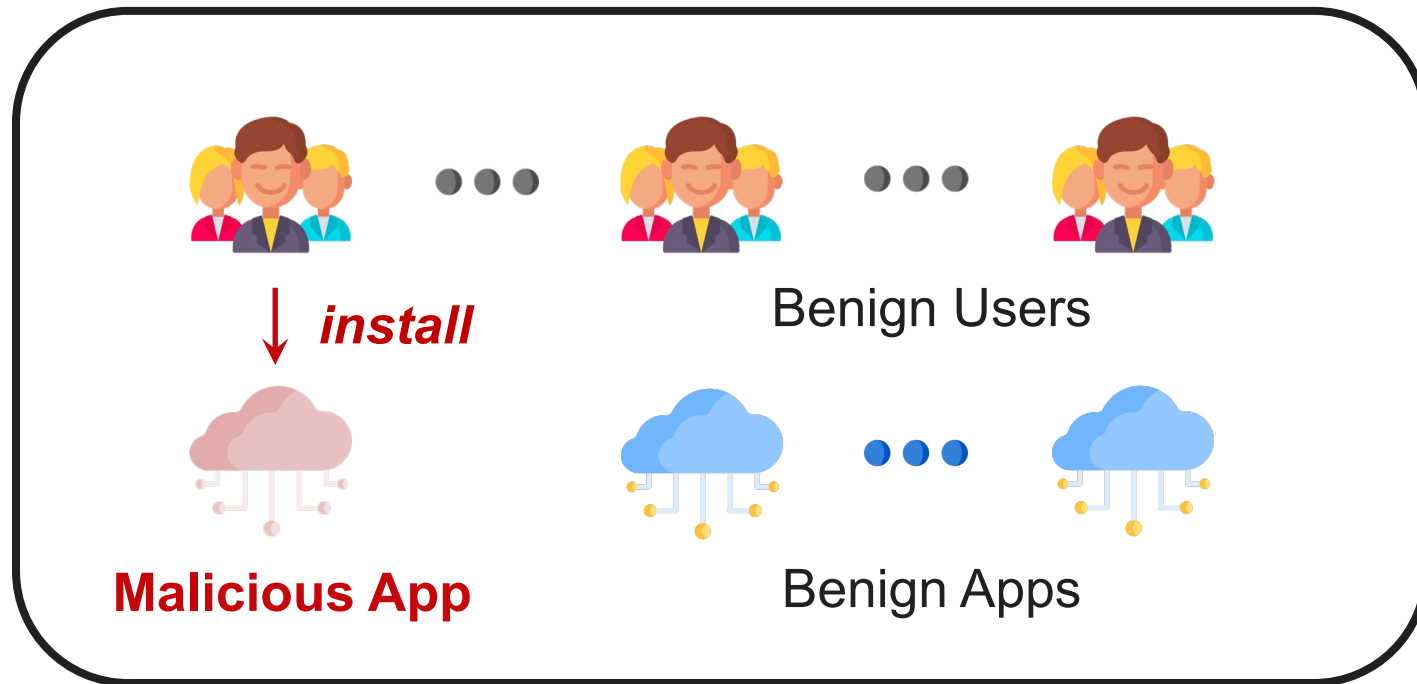
3. Write Permission Scopes

- *Post messages*
- *Post messages on behalf of users*

4. User Approves Permissions

# Threat Model: Malicious Apps in BCP

Target BCP Workspace



- Attacker tricks the user to install a malicious app
- The user is curious and installs a malicious app
- The benign app becomes malicious

# Challenges & Our Methodology

- Incomplete permission model description.

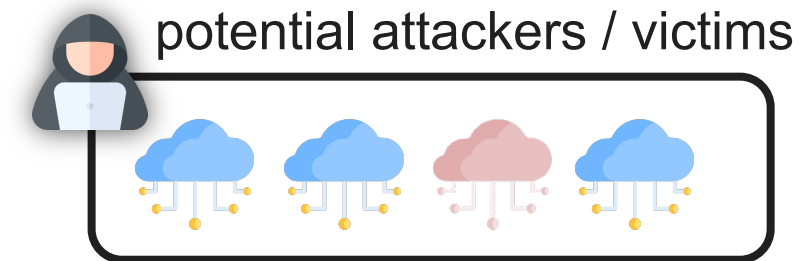
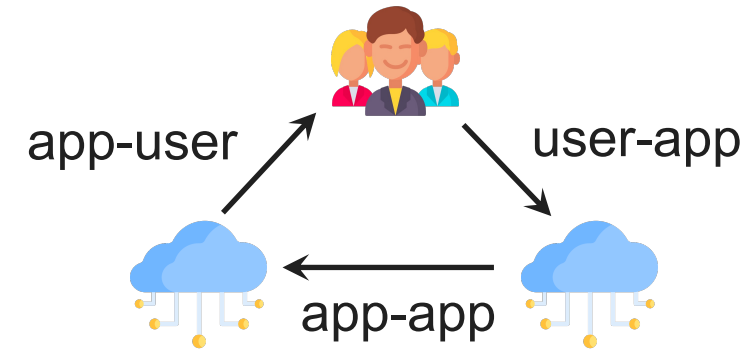
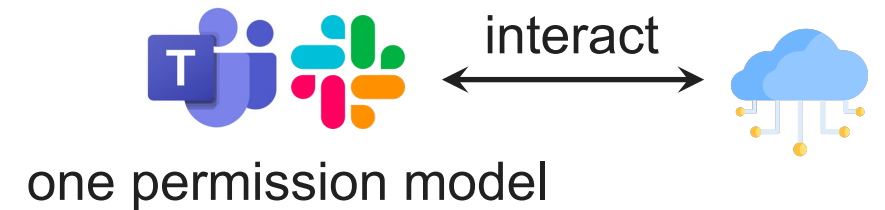
**We extract a unified abstraction.**

- Closed-source apps in the cloud.

**We examine all possible interactions.**

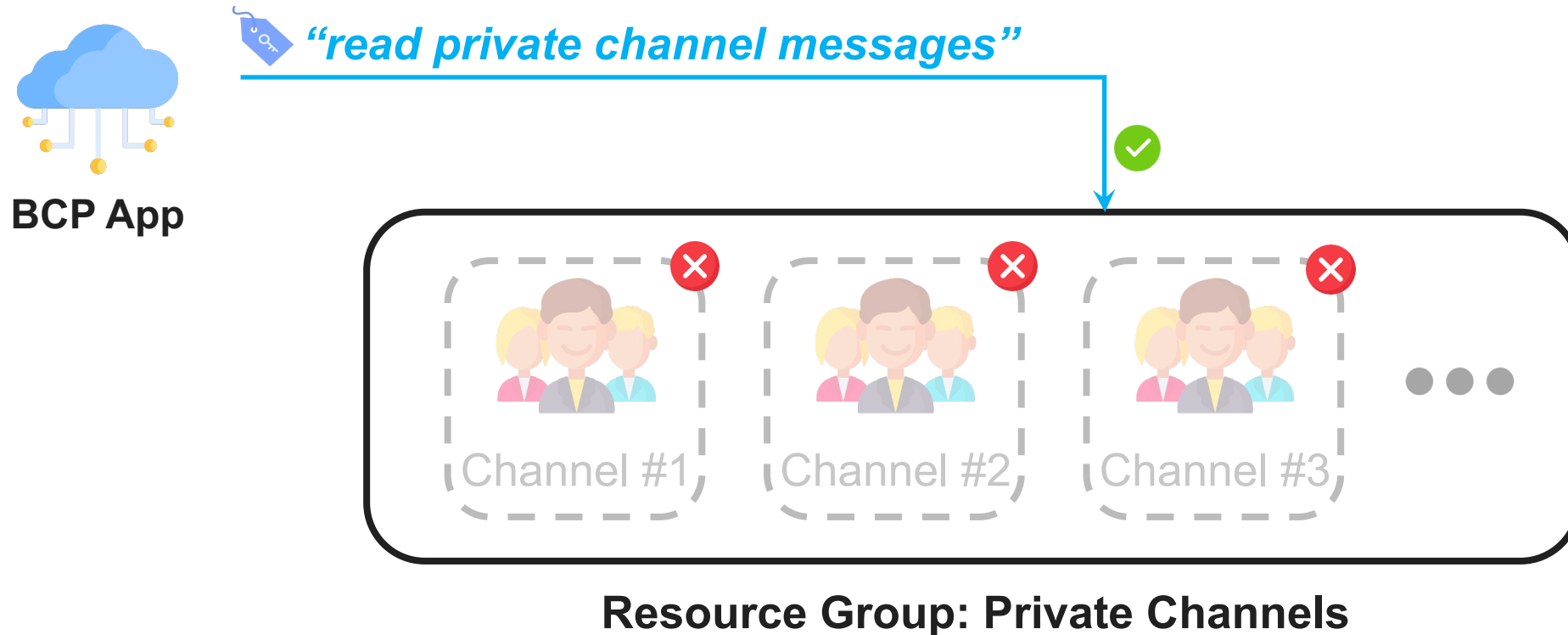
- Unscalable in-depth analysis.

**We estimate potential attackers & victims.**



# A Two-Level Unified Permission Model

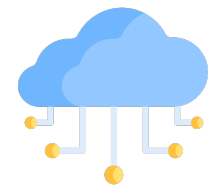
- **Level 1:** coarse-grained OAuth permissions scopes





# A Two-Level Unified Permission Model

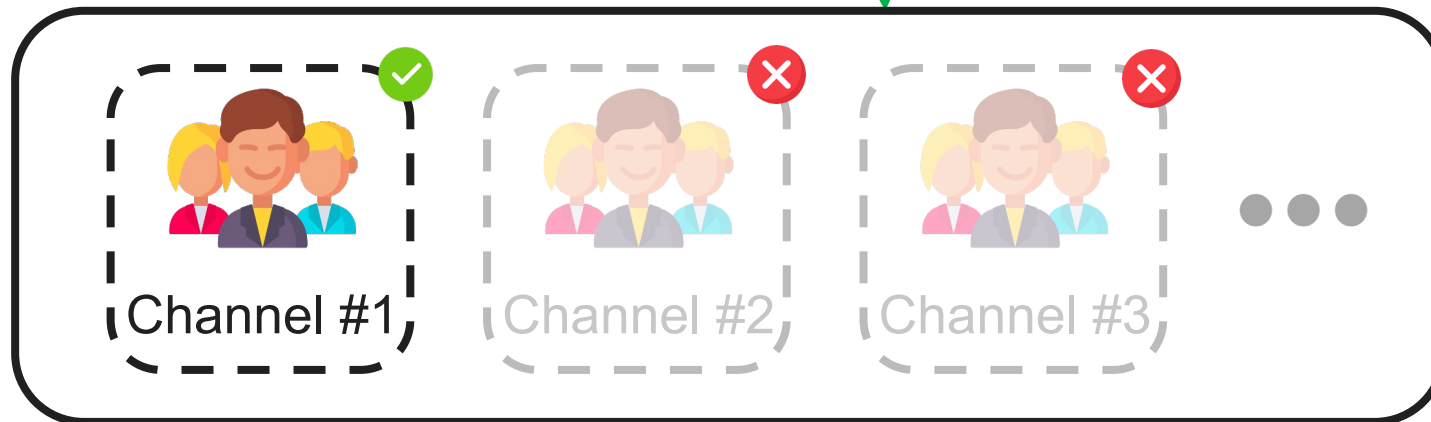
- **Level 1:** coarse-grained OAuth permissions scopes
- **Level 2:** fine-grained runtime policy checks



BCP App

 *“read private channel messages”*

 *“refers to private channel #1”*



Resource Group: Private Channels

# Violation of Security Principles

- **Least Privilege**

Runtime policies are ad-hoc and incomplete.



*“post messages to channels”*



*“post messages to users”*



*“only if the app joined this channel”*



*null*

- **Complete Mediation**

Provenance of resources are not properly tracked.

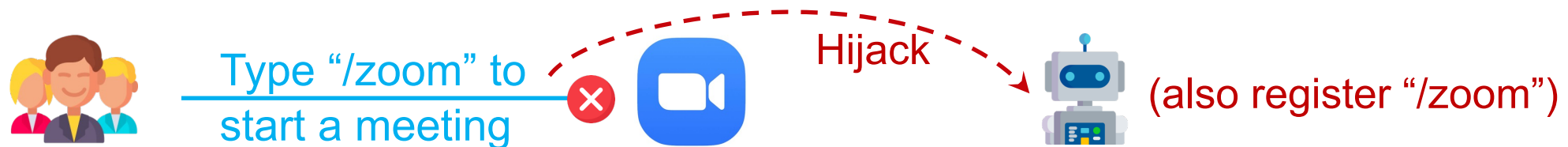


# All Types of Interactions Are Vulnerable

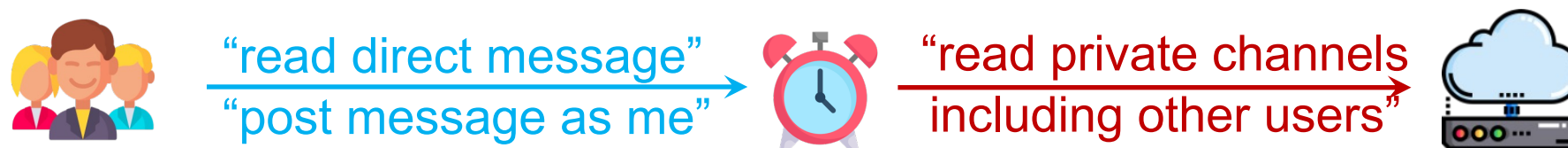
- **App-to-App Interaction** → *Delegation Attacks*



- **User-to-App Interaction** → *Command Hijacking*

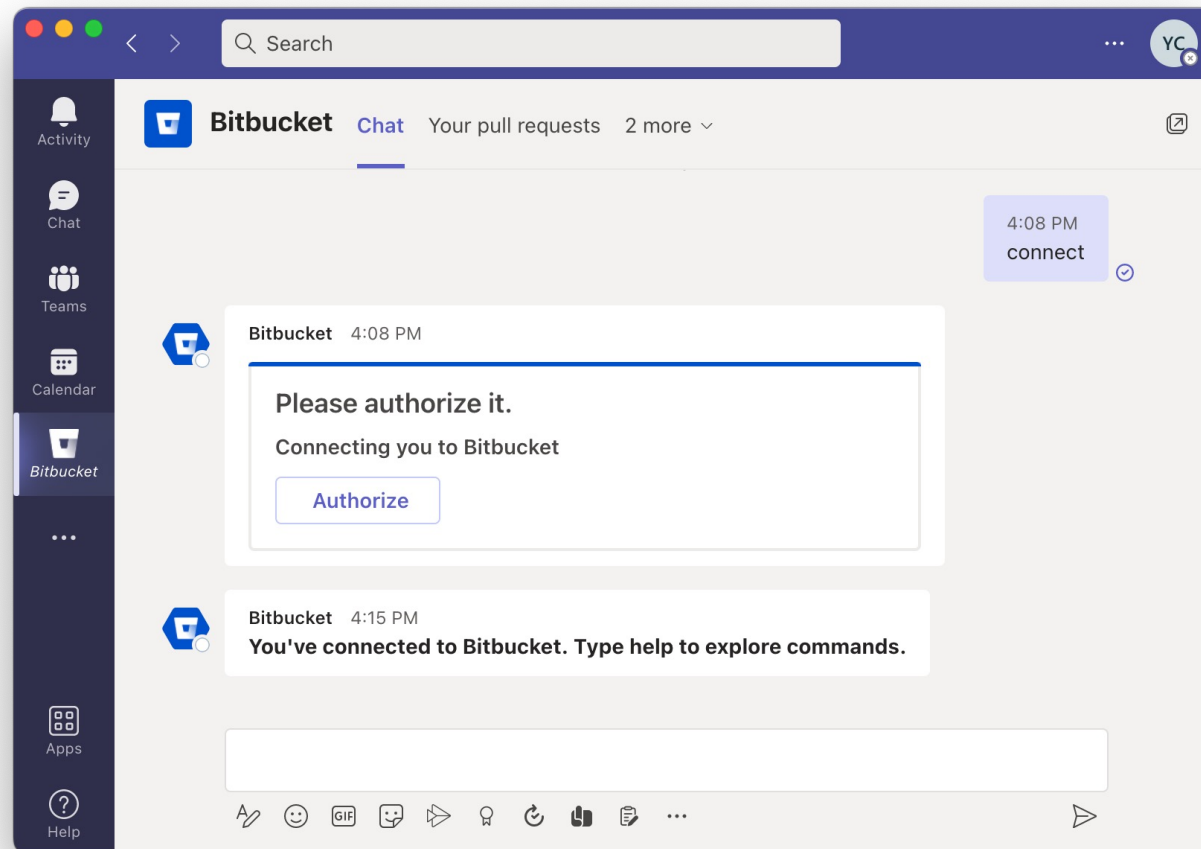


- **App-to-User Interaction** → *Privilege Escalation*



# Delegation Attack: Merge Malicious PRs

- **Step 1:** User installed Bitbucket app.



“Merge #1”



“Confirm merge #1?”



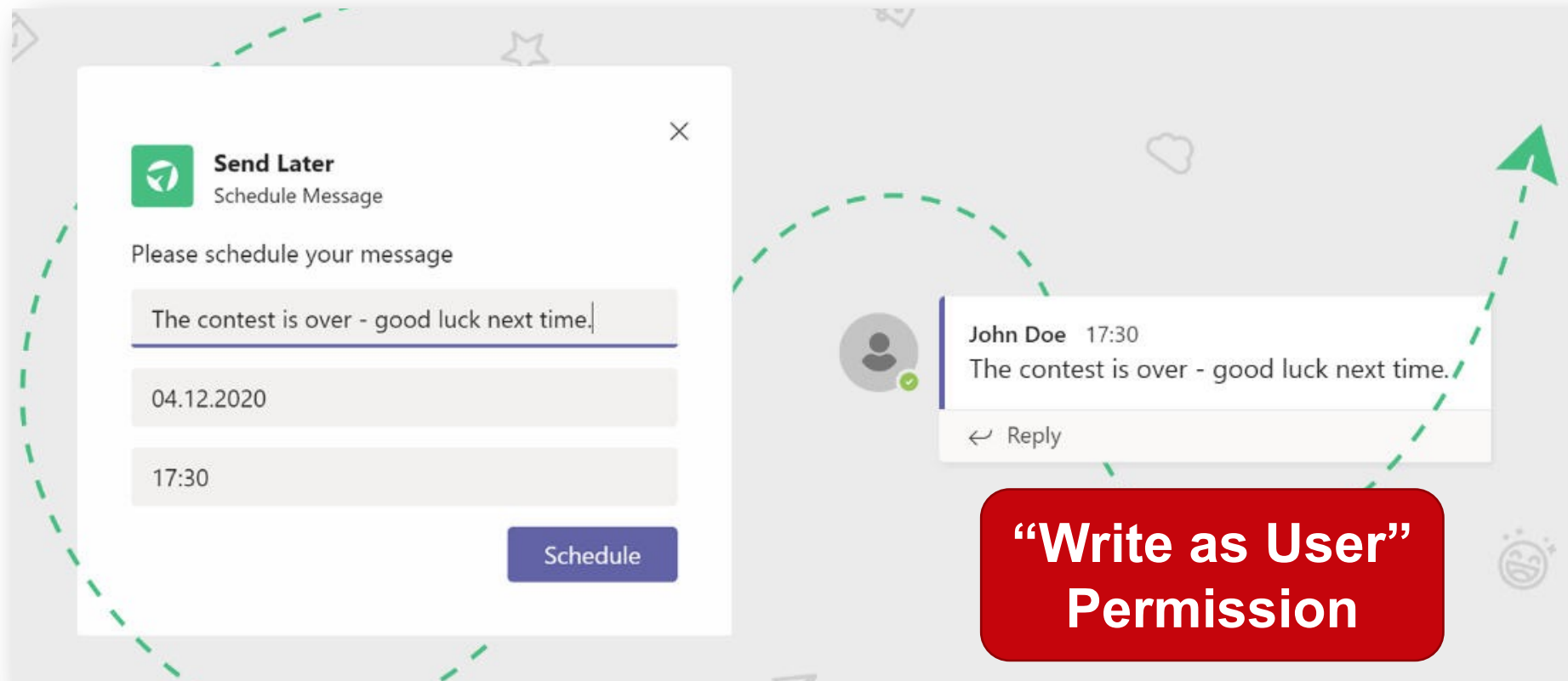
“Yes”



“Merged!”

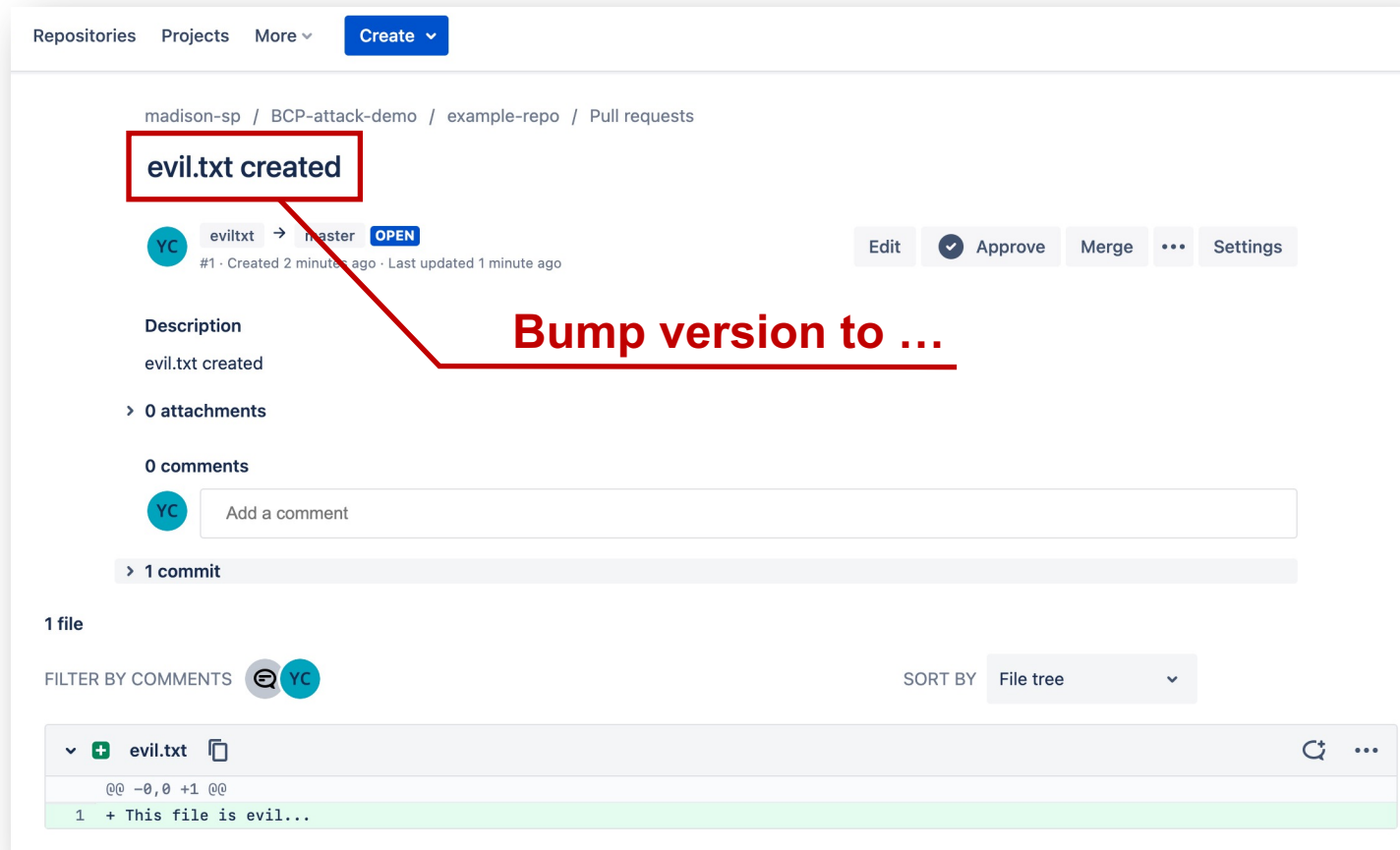
# Delegation Attack: Merge Malicious PRs

- **Step 2:** User installed “Send Later” app (*our malicious demo*).



# Delegation Attack: Merge Malicious PRs

- **Step 3:** Attacker creates a **malicious** Pull Request #1.



The screenshot shows a GitHub Pull Request interface for a repository named 'example-repo'. The pull request is titled 'evil.txt created' and is in the 'OPEN' state. The description of the pull request is 'evil.txt created'. The interface shows 0 attachments, 0 comments, and 1 commit. The file list shows a single file named 'evil.txt' with a diff view showing a single line of code: '1 + This file is evil...'. A red box highlights the title 'evil.txt created', and a red arrow points from this box to the text 'Bump version to ...'.

Repositories Projects More **Create**

madison-sp / BCP-attack-demo / example-repo / Pull requests

**evil.txt created**

YC eviltxt → master **OPEN** Edit Approve Merge ... Settings

#1 · Created 2 minutes ago · Last updated 1 minute ago

**Description**  
evil.txt created

> 0 attachments

0 comments

YC Add a comment

> 1 commit

1 file

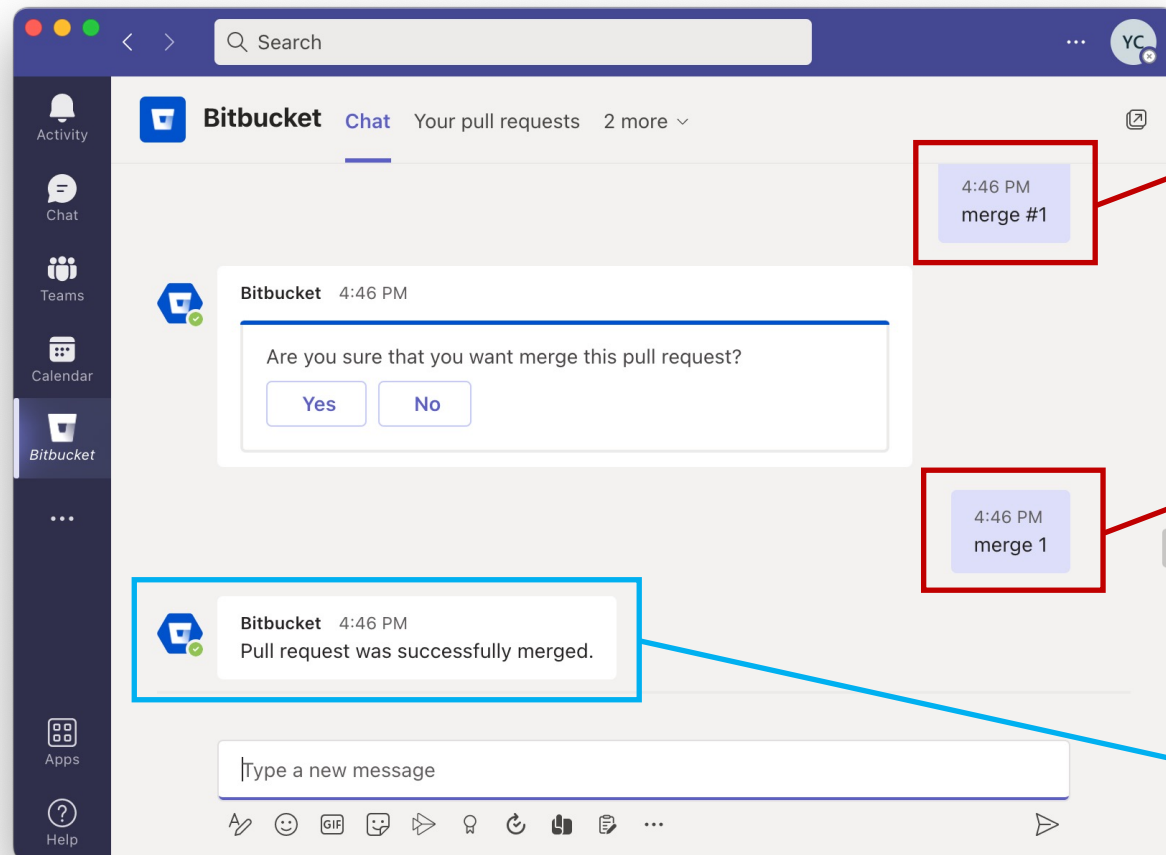
FILTER BY COMMENTS YC SORT BY File tree

evil.txt

```
@@ -0,0 +1 @@
1 + This file is evil...
```

# Delegation Attack: Merge Malicious PRs

- **Step 4:** Malicious **“Send Later”** app talks to the **Bitbucket** app.



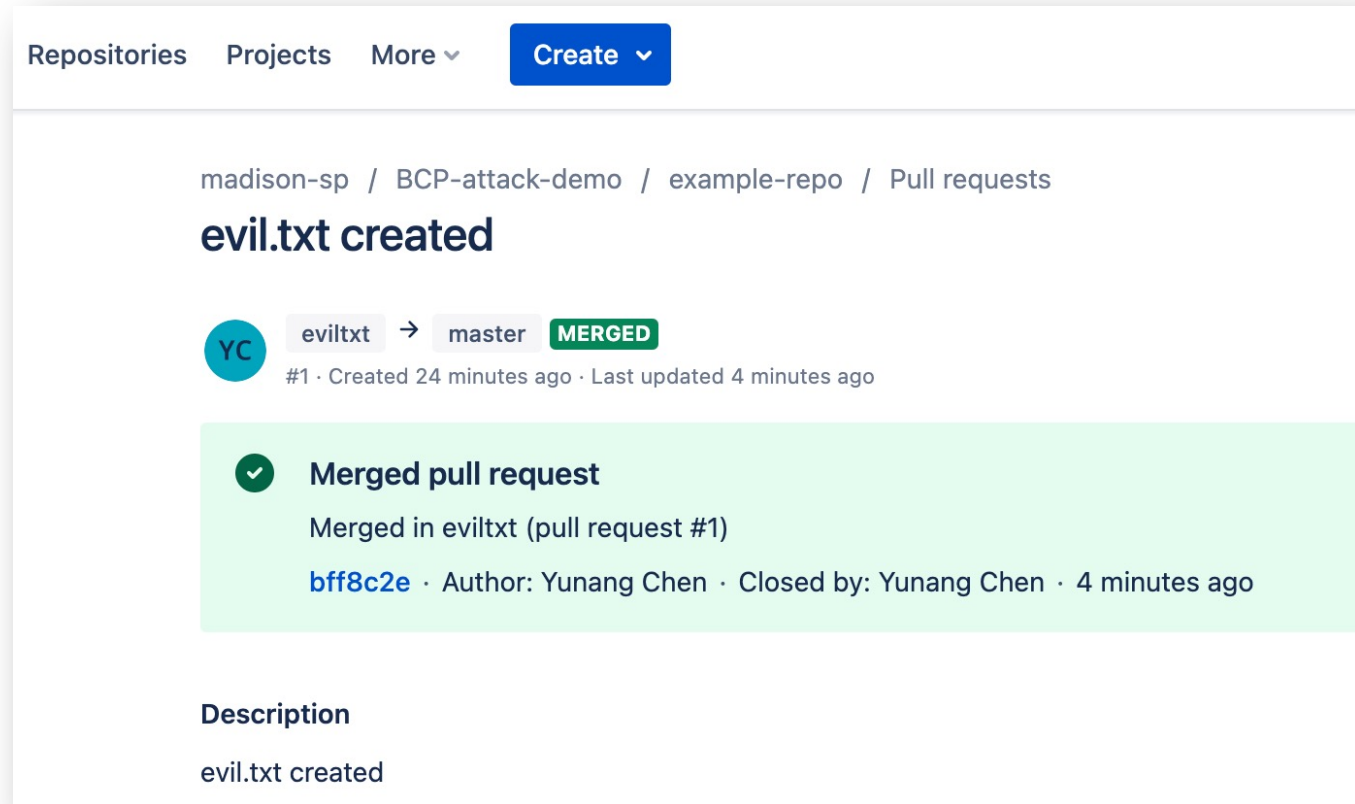
First request “merge #1”

Confirm “merge #1”

“Pull request was successfully merged.”

# Delegation Attack: Merge Malicious PRs

- **Step 5:** Bitbucket merges the malicious pull request.



The screenshot shows the Bitbucket interface for a pull request. At the top, there are navigation links for "Repositories", "Projects", "More", and a blue "Create" button. The breadcrumb path is "madison-sp / BCP-attack-demo / example-repo / Pull requests". The main heading is "evil.txt created". Below this, a pull request is shown from user "YC" (profile picture) to the "master" branch, with a green "MERGED" badge. The pull request details include "#1 · Created 24 minutes ago · Last updated 4 minutes ago". A green notification box contains a checkmark icon and the text "Merged pull request", followed by "Merged in eviltxt (pull request #1)" and "bff8c2e · Author: Yunang Chen · Closed by: Yunang Chen · 4 minutes ago". At the bottom, the "Description" section shows "evil.txt created".





# Potential Prevalence Analysis

- Collect each app's requested permissions.
- **Capable Apps** — Have write permissions needed for attacks.
- **Susceptible Apps** — Have read permissions affected by attacks.

Attacks	# Capable Apps (MS Teams)	# Capable Apps (Slack)	# Susceptible Apps (Slack)
Delegation Attacks	427 (33%)	563 (23%)	1,493 (61%)
Command Hijacking	77 (6%)	270 (11%)	1,266 (52%)
Privilege Escalation	n/a	11	n/a

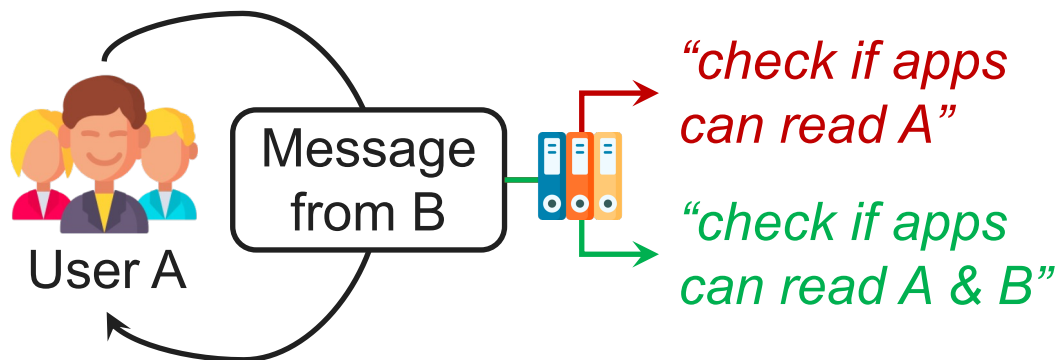
# Countermeasures: Improve Permission Models

## Better Design

- *Finer-grained Scopes*

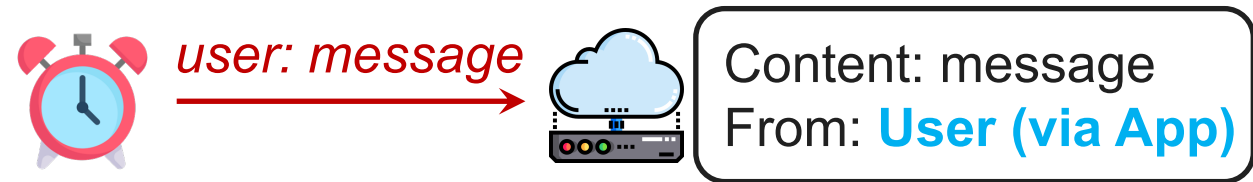


- *Stricter Runtime Policies*

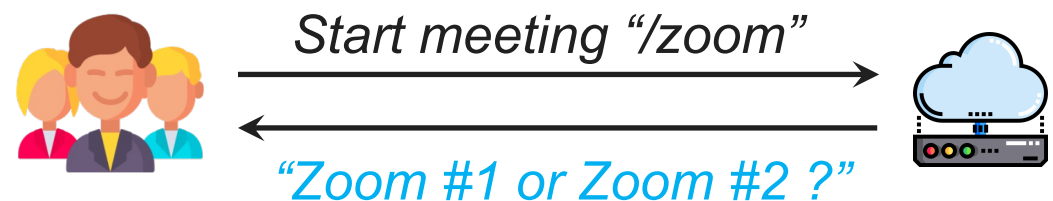


## Better Execution

- *Track Provenance of Actions*



- *Explicit User Confirmation*





# Disclosure & Responses

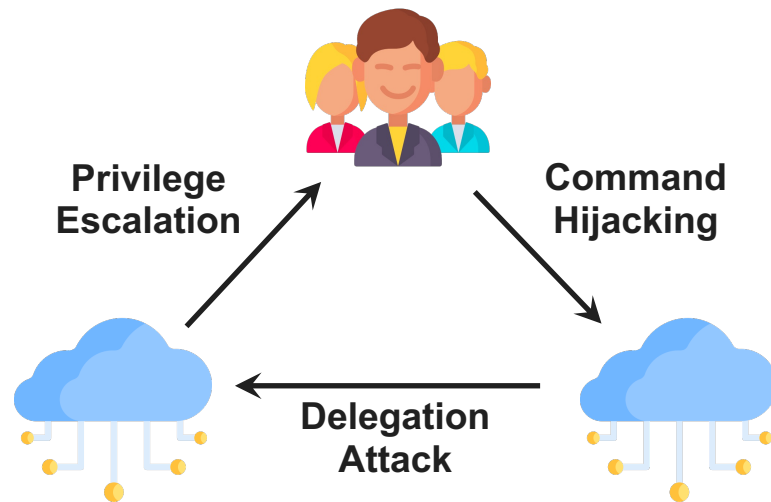
---

- Confirmed attacks
- Workspace → a trusted environment
- Administrator → will correctly manage apps
  
- Our tips for administrators
  - Consider limiting users from installing apps
  - Actively monitor the behavior of installed apps
  - Only approve delegation permissions from trusted apps

# Summary

---

- BCPs have become a hub for **sensitive third-party resources**.
- We provide **security analysis** under malicious apps.
- All types of interactions are **vulnerable & potentially prevalent**.



Paper



Demo