Client-Side HTTP Cookie Security: Attack and Defense

David Wyde
DEF CON 22

Game Plan

- Why are HTTP cookies valuable to attackers?
- How do popular web browsers store cookies?
- How can cookies be stolen?
- How can cookies be protected?

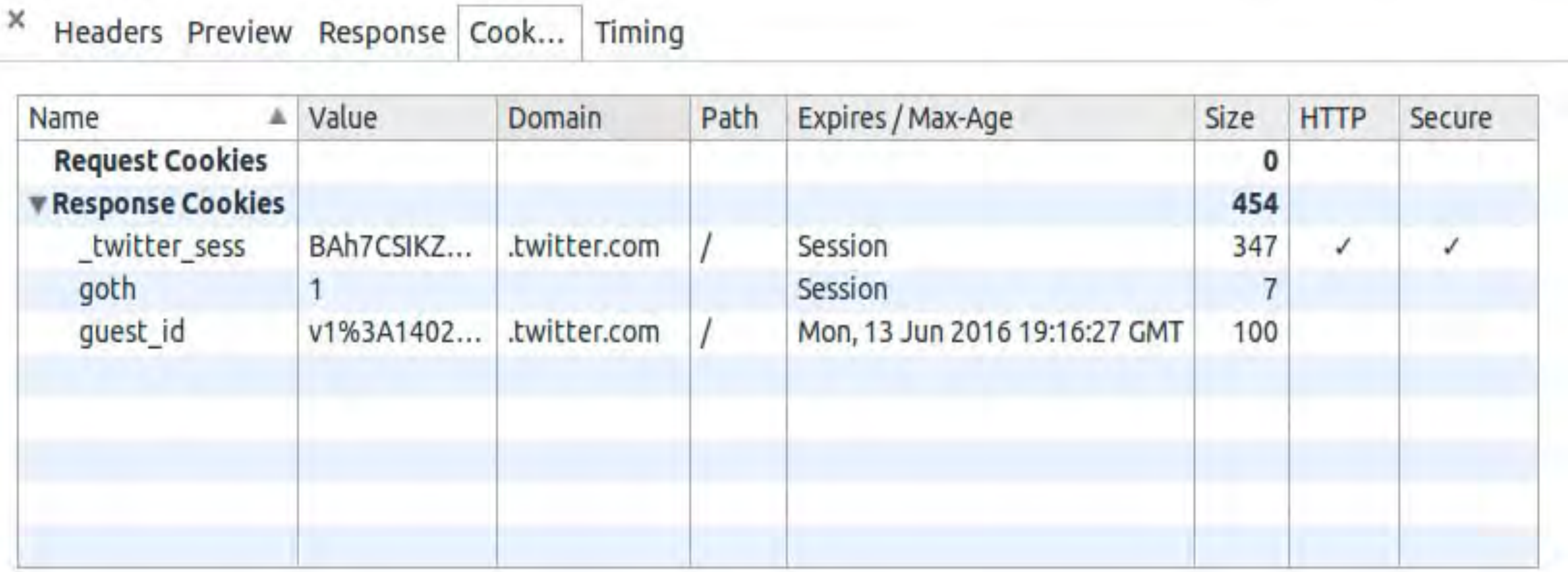
Disclaimers

- The opinions in this presentation are mine, and not my employer’s.
- The security issues I discuss are not specific to any one website, and are not vulnerabilities in the conventional sense.

What is an HTTP Cookie?

- Cookies are transmitted as HTTP headers
  - Name-value pairs
- HTTP clients store state using cookies
  - E.g., trade credentials for a session cookie

Cookies in Action



× Headers Preview Response Cook... Timing

Name	Value	Domain	Path	Expires / Max-Age	Size	HTTP	Secure
Request Cookies					**0**		
▼ Response Cookies					**454**		
_twitter_sess	BAh7CSIKZ...	.twitter.com	/	Session	347	✓	✓
goth	1			Session	7		
guest_id	v1%3A1402...	.twitter.com	/	Mon, 13 Jun 2016 19:16:27 GMT	100		

User-Readable Data

- Any process that runs as your user can read:
  - Your private keys
  - Some software saves passwords as plaintext
  - Web browser cookies

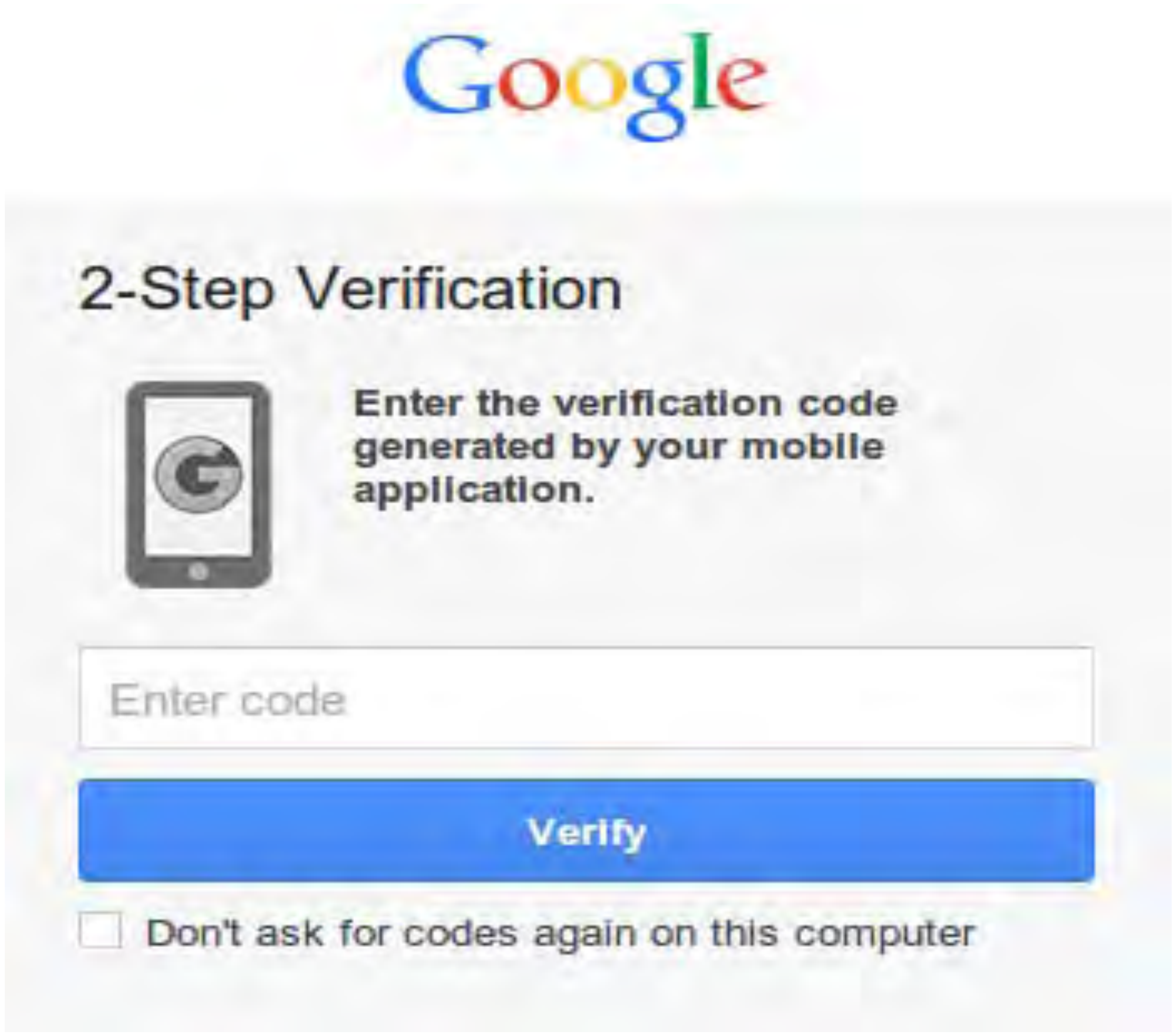
- Damage is done without privilege escalation

Cookies Are Valuable to Attackers

- Cookies can be more valuable than passwords
  - Gmail: bypass two-factor authentication
  - Facebook: don't warn of login from a new device

- Counterpoints
  - "Please re-enter your password”
  - Cookies expire

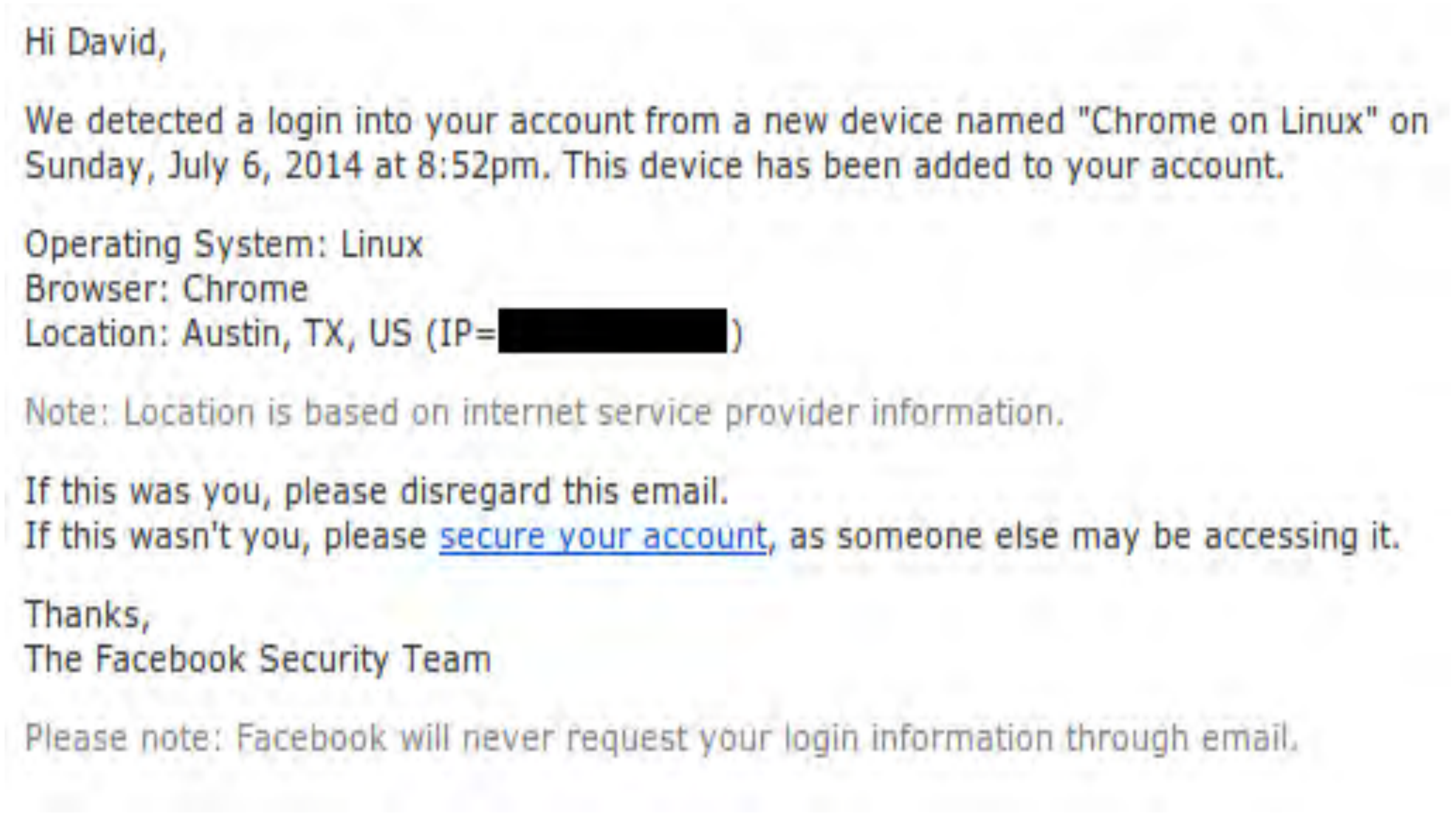
Gmail: Two-Factor Authentication


Google

2-Step Verification

Enter the verification code generated by your mobile application.

Facebook: New Login Email



Hi David,

We detected a login into your account from a new device named "Chrome on Linux" on Sunday, July 6, 2014 at 8:52pm. This device has been added to your account.

Operating System: Linux
Browser: Chrome
Location: Austin, TX, US (IP=[redacted])

Note: Location is based on internet service provider information.

If this was you, please disregard this email.
If this wasn't you, please secure your account, as someone else may be accessing it.

Thanks,
The Facebook Security Team

Please note: Facebook will never request your login information through email.

Browser Cookie Storage

Cookie Storage: Intro

- Almost all browsers store cookies as plaintext
- The HttpOnly and Secure flags apply inside browsers
    - Malware need not respect them

Firefox

- Stores cookies in an SQLite database
    - Cookies can be read using sqlite3, Python, etc.

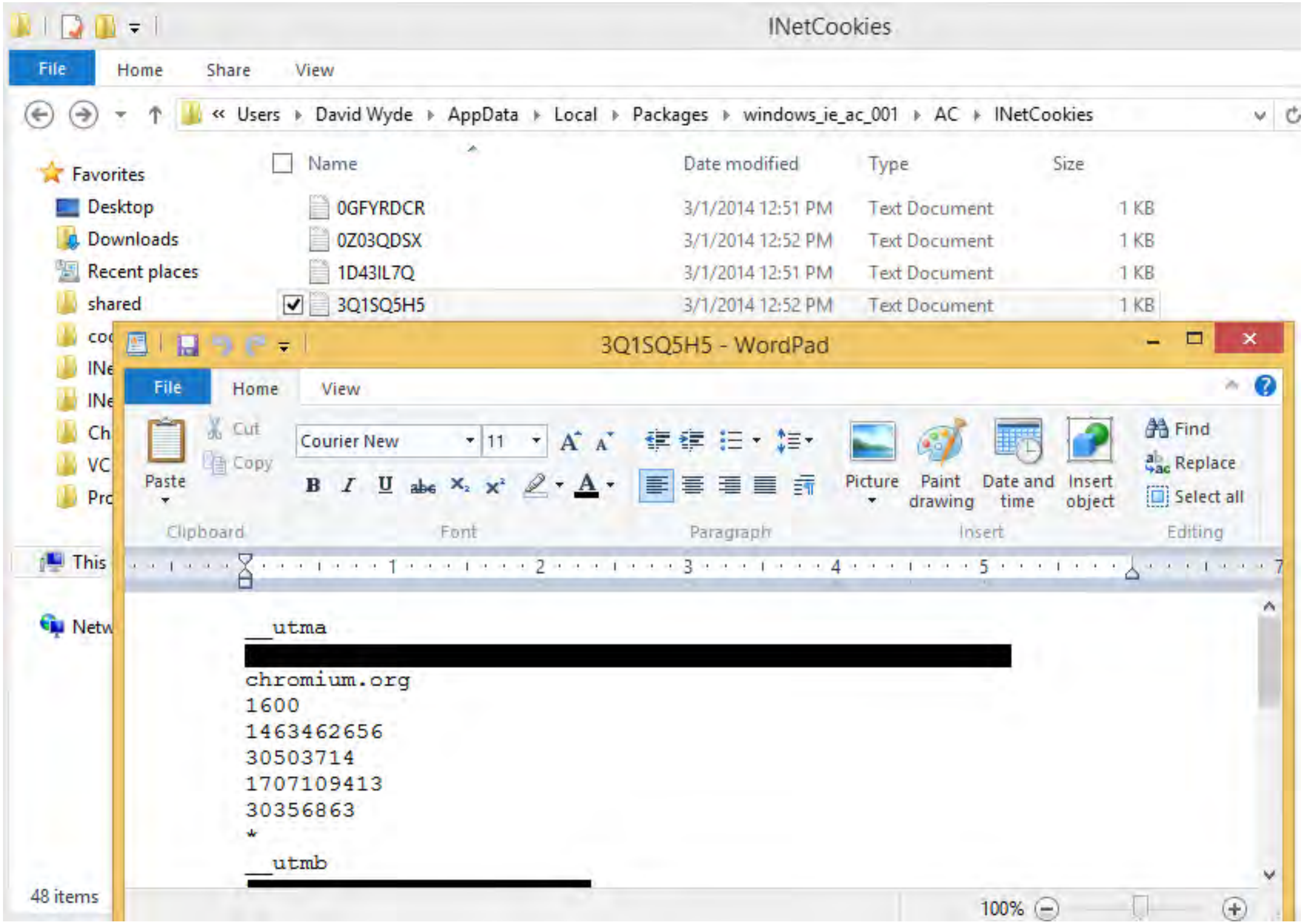
Reading Firefox Cookies

```
$ sqlite3 ~/Library/Application\ Support/Firefox/Profiles/*/cookies.sqlite
SQLite version 3.7.13 2012-07-17 17:46:21
Enter ".help" for instructions
Enter SQL statements terminated with a ";"

sqlite> .schema
CREATE TABLE moz_cookies (id INTEGER PRIMARY KEY, baseDomain TEXT, appId
INTEGER DEFAULT 0, inBrowserElement INTEGER DEFAULT 0, name TEXT, value
TEXT, host TEXT, path TEXT, expiry INTEGER, lastAccessed INTEGER,
creationTime INTEGER, isSecure INTEGER, isHttpOnly INTEGER, CONSTRAINT
moz_uniqueid UNIQUE (name, host, path, appId, inBrowserElement));
CREATE INDEX moz_basedomain ON moz_cookies (baseDomain, appId,
inBrowserElement);

sqlite> SELECT value FROM moz_cookies WHERE name='GX';
DQAAAPEAAABWYmsr2PFvwQi4XhQWYcw_5coZVfjh-efmKTNeLjyLx04sHi_Ih-
xMOsSRaZ6J38QzDGyCt5v6DKYkkoc6TeX8QKuaOPSAqqGTEo4v2Y6kvmzlS-SvdU4zTcuJ-
z4uCf7uiZ7Ic-
H6U5Mt7leqmsDhQeEoL01z5OF6iLoxUeCHU_91eWrA2bOpU8ppqVjutpi4WVhyqLV7WX6hgSnE
kWnpsN-XwcDF84V7u0DrlKCQFupzmCfa3nt_tARY-SxbyNrmY_0rH4YF-
xBVvPFXBQpKqUZrW_zMdGmWgmPER_7mBTGXtlh9PM5nCP_bw09oIqXrQb_OhHe7c3AnnIg2EIq
g
```

Internet Explorer

- Stores cookies as text files
- The folder varies depending on IE version
- Filenames are random: need to read the files

Reading Internet Explorer Cookies



Name	Date modified	Type	Size
0GFYRDCR	3/1/2014 12:51 PM	Text Document	1 KB
0Z03QDSX	3/1/2014 12:52 PM	Text Document	1 KB
1D43IL7Q	3/1/2014 12:51 PM	Text Document	1 KB
3Q1SQ5H5	3/1/2014 12:52 PM	Text Document	1 KB

```
__utma
[redacted]
chromium.org
1600
1463462656
30503714
1707109413
30356863
*
__utmb
[redacted]
```

Opera and Safari

- Custom binary formats
  - Can be parsed by free software tools
- Safari: Cookies.binarycookies
- Opera: cookies4.dat

Reading Safari Cookies

```
$ python ~/Desktop/BinaryCookieReader.py ~/Library/
Cookies/Cookies.binarycookies | grep yahoo

Cookie :
hpc=d=ItIKgZXDu9Pkv2_sEb7ygoVyN9bHZ2mmjnr8eBC8z9Ynw88Tayw
7ixgQfT4vleMQ56bGUussxMNmYBusbq3RHgXIkea3DhM.Yzckc.y6GAQE
iJoPoK1DzyvYg1cyBoMWlZccOkvv7wvPUmDHnNk1uyiJwon3_YjfMMyCX
stKdmUKmePy_Wn04tFoVbui1wlLTuSpqTw-&v=2;
domain=.www.yahoo.com; path=/; expires=Wed, 15 Jul 2015;

Cookie : B=2b26v3t9s955p&b=3&s=oh; domain=.yahoo.com;
path=/; expires=Fri, 15 Jul 2016;

Cookie : CRZY=%7B%221048616551%22%3A%7B%22expires
%22%3A1405564858541%2C%22data%22%3A%7B%22nv%22%3A1%2C
%22bn%22%3A0%7D%7D%7D; domain=.yahoo.com; path=/;
expires=Thu, 17 Jul 2014;
```

Reading Opera Cookies

```
$ python opera_reader.py ~/.opera/cookies4.dat

file_version_number 4096
app_version_number 8193
idtag_length 1
length_length 2
domain record
   [('0x1e', 'name of the domain part', 3, 'org')]
   end of path record
   domain record
      [('0x1e', 'name of the domain part', 8, 'slashdot')]
      cookie record
      [('0x10', 'name of the cookie', 6, '__gads'), ('0x11',
'value of the cookie', 69,
'ID=2628549bf6c27042:T=1405392507:S=ALNI_Maix2zTTIQ4159AfUM0tH
p7h_ODgQ'), ('0x12', 'expiry', 8, '2016-07-13 21:48:27'),
('0x13', 'last used', 8, '2014-07-14 21:49:28'), ('0x28',
'unknown cookie data id', 8,
'\x00\x00\x00\x00\x00\x00\x00\x00'), ('0xa9', 'unknown cookie
data id', 0, '')]
```

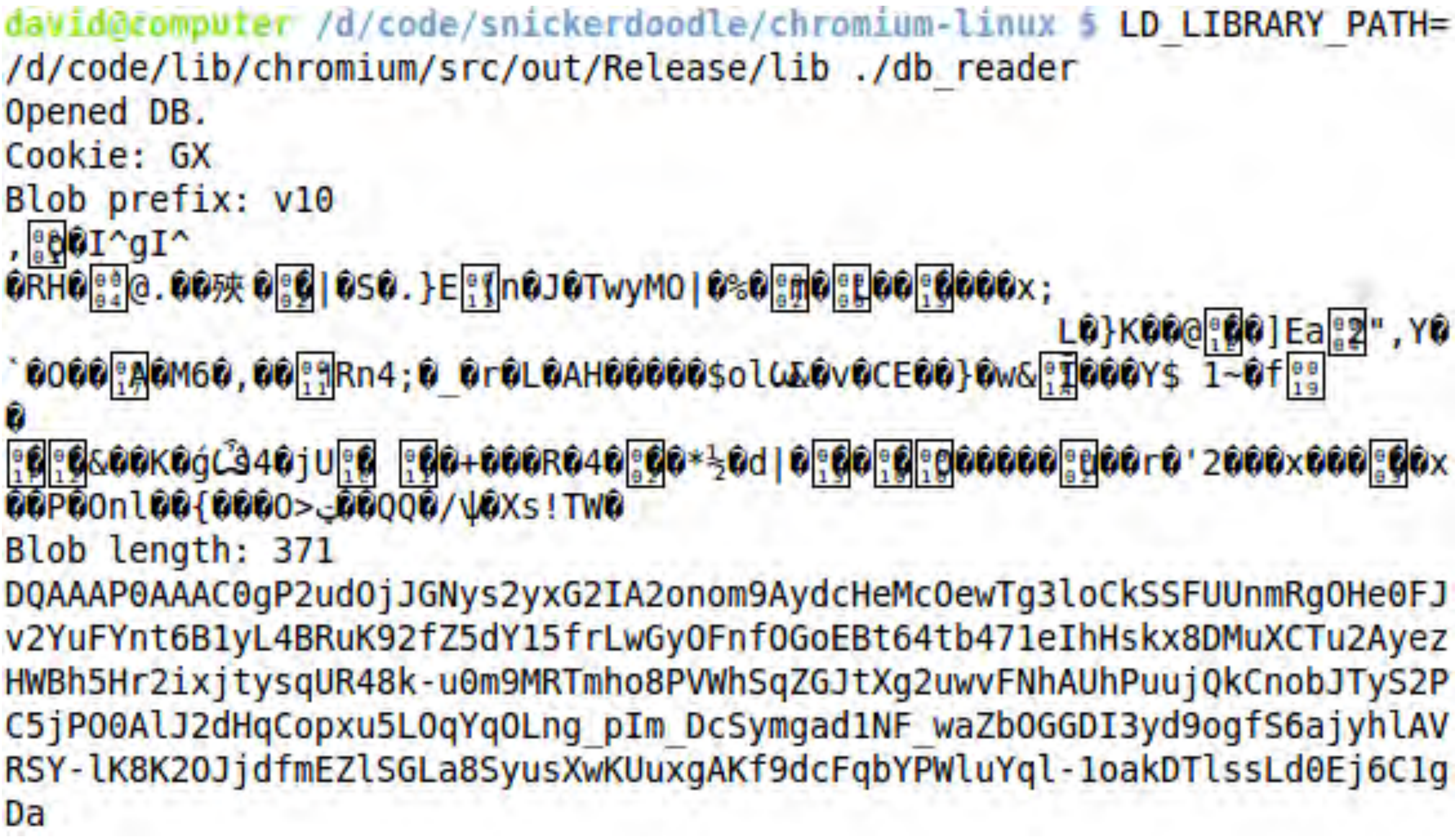
Chromium

- Encrypts cookies in recent versions
  - Implementation and security vary by platform
- Stores cookies in an SQLite database
  - BLOB field for encrypted cookie values

Chromium on Linux

- Linux has no single standard keyring mechanism
  - (KDE, Gnome, etc.)
- Cookies encrypted with AES (symmetric key)
  - Hard-coded key and salt
- Can be decrypted on any machine
  - Link against Chromium libs, call code to decrypt

Reading Chromium Cookies: Linux



```
david@computer /d/code/snickerdoodle/chromium-linux $ LD_LIBRARY_PATH=
/d/code/lib/chromium/src/out/Release/lib ./db_reader
Opened DB.
Cookie: GX
Blob prefix: v10
,[illegible]��I^gI^
�RH�[illegible]@.��殃 �[illegible]|�S�.}E[illegible]n�J�TwyMO|�%�[illegible]m�[illegible]L��[illegible]����x;
                                                  L�}K��@[illegible]��]Ea[illegible]",Y�
`�O��[illegible]A�M6�,��[illegible]Rn4;�_�r�L�AH�����$olω&�v�CE��}�w&[illegible]I���Y$ 1~�f[illegible]
�
[illegible]�&��K�gĆ34�jU[illegible] [illegible]��+���R�4�[illegible]��*½�d|�[illegible]��[illegible]�[illegible]������[illegible]u��r�'2���x���[illegible]��x
��P�Onl��{����O>ـ��QQ�/Ѱ�Xs!TW�
Blob length: 371
DQAAAP0AAAC0gP2udOjJGNys2yxG2IA2onom9AydcHeMcOewTg3loCkSSFUUnmRgOHe0FJ
v2YuFYnt6B1yL4BRuK92fZ5dY15frLwGyOFnfOGoEBt64tb471eIhHskx8DMuXCTu2Ayez
HWBh5Hr2ixjtysqUR48k-u0m9MRTmho8PVWhSqZGJtXg2uwvFNhAUhPuujQkCnobJTyS2P
C5jPO0AlJ2dHqCopxu5LOqYqOLng_pIm_DcSymgad1NF_waZbOGGDI3yd9ogfS6ajyhlAV
RSY-lK8K2OJjdfmEZlSGLa8SyusXwKUuxgAKf9dcFqbYPWluYql-1oakDTlssLd0Ej6C1g
Da
```

Reading Chromium Cookies: Linux

```
[david@localhost Desktop]$ python chromium_b64_cookie_linux.py

djEwXgab42ZPnVqGRirZqEHsvEN8bC/
chT84CbmJxMSJDr6XA7mQLZdCuLwYSNA6srVf7NDn7rHdBOFJf8SX4jdCxlQhcrUGH
+0KzFz
+hUxUcgRzy6jWEZyAe4QDegh1YGtfdCGiZ2TgHkEifJ0Mojf4VpuKhFw7SVpCzCorz86JF
czNpco7LZwM/xng7UPmVEY4sIQwAGlTXoY9ThgaliP8HGviwkK0ozW9/FMUiGaxBIqDD
+FSfsGszckv9zRbK8XL2PbHVslRmG2ENQ8wESu2Czajb20BQ+L3dMRvOcVbW+gwt+H/
cBG23dnjnhFxGcvm9DSDyz87o5ssILocgMT+kddTBCG8ohvy7iNE3njT6WOFktK8Hd/
+rhSUarnCtZt9UB1EZtikWbpqn0PKrVCKn0wVpO4oyeDIe96xEesn/IM=
```

```
david@computer /d/code/snickerdoodle/chromium-linux $
LD_LIBRARY_PATH=/d/code/lib/chromium/src/out/Release/lib ./
base64_reader $(cat fedora-cookie.txt)

DQAAANMAAAD55DvOAnmlugeHzwGKs0asFxYtMfXl-
Xdg7MtLYmdj5GDI3iyPh70Ds6OKgogfATna2KV9d7JqZxJ5e7SA-
sbH1oxvQFs1WsFo_9WzEfj9VamEV5C0uml6tVuzhIGzrrKM0__0SI6QANb-y-
qyM3QJSKCB7QrXR_Ug7lFzjibDW7Fsfg15SUCTmfQz9YLBP4oYSOt_pJRVf5XZgbN_2J-
KQzBqtZznZwKVE4TatBaAucT-
R9jXnjM5aMdoJvr7ubghi0p1m7yvPevqNNRItPkeB5aV_cPXHKRMjwhAAk6_2w
```

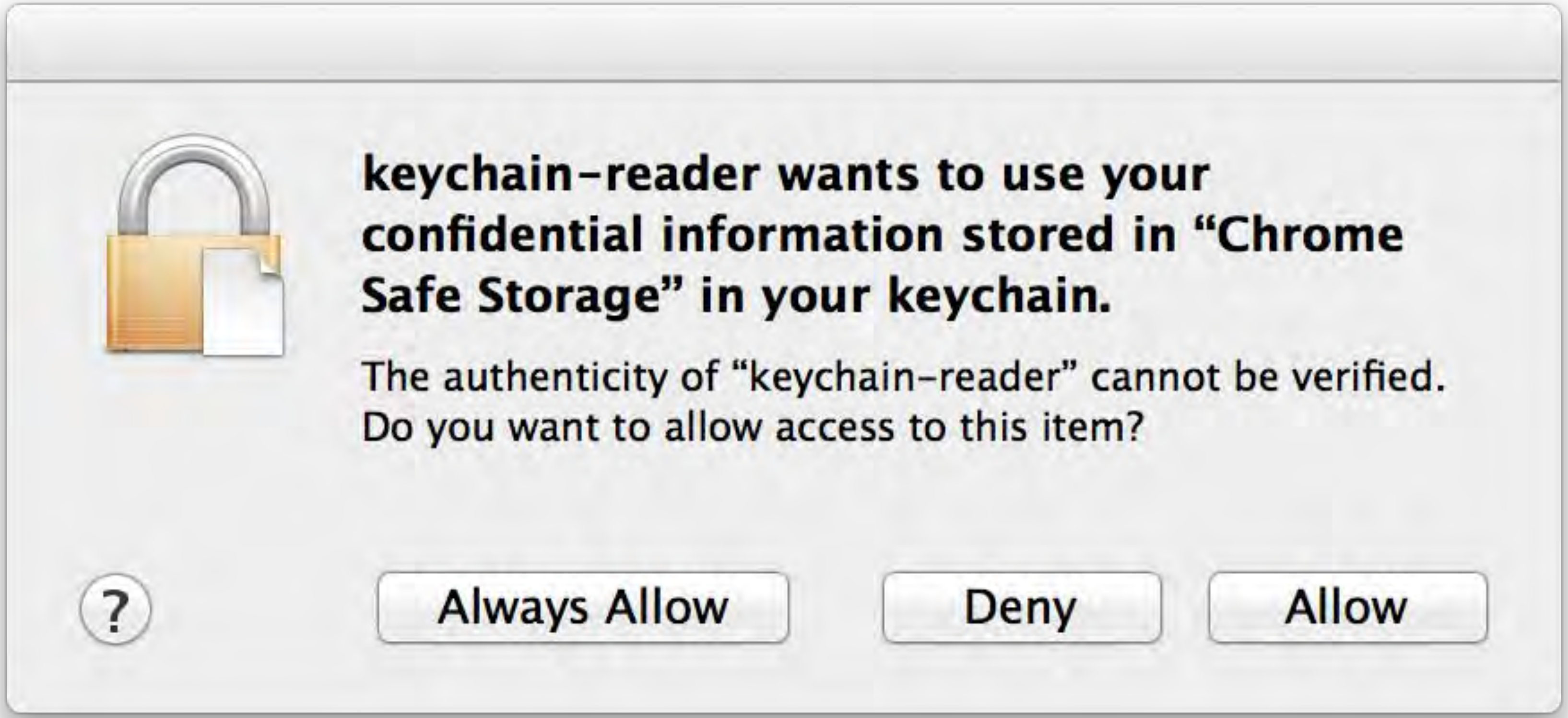
Chromium on Windows

- *CryptProtectData* is used to encrypt
  - A Windows cryptography API
  - Uses login credentials as part of the encryption
- *CryptUnprotectData* is used to decrypt
  - Must be called by the user that encrypted, on the same machine

Chromium on Mac

- Store an encryption key in the system keychain
  - If no key exists, a random one is generated
- AES is used to encrypt/decrypt
- Keychain prompts when accessed from unsigned apps

Reading Chromium Cookies: Mac



keychain-reader wants to use your confidential information stored in "Chrome Safe Storage" in your keychain.

The authenticity of "keychain-reader" cannot be verified. Do you want to allow access to this item?

Browser Cookie Storage: Summary

- Chromium encrypts cookies on Windows and Mac
- Chromium obfuscates cookies on Linux
- Other popular browsers store cookies as plaintext

Attack Vectors

Physical Access

- Cookies are there for the taking with most browsers
- Chromium protects you on Windows and Mac

Social Engineering

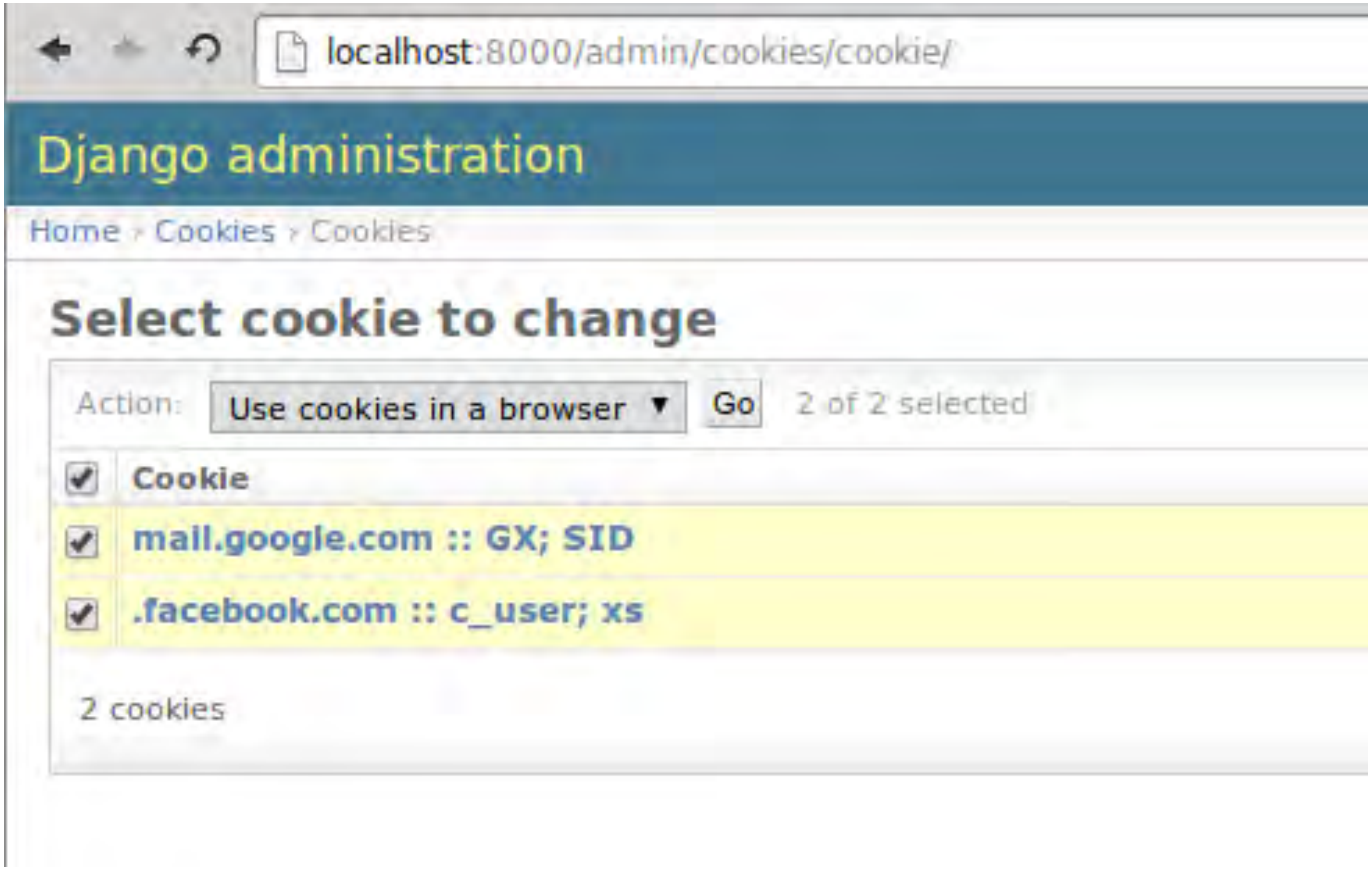
- Excel/Word macros
- Malicious executables
- Don't need to install anything - just run once

Malware

- Drop and run an executable to extract cookies
  - Metasploit
  - Any process that runs as your user

- HTTP POST cookies to a malicious server

Proof of Concept


localhost:8000/admin/cookies/cookie/

Django administration

Home › Cookies › Cookies

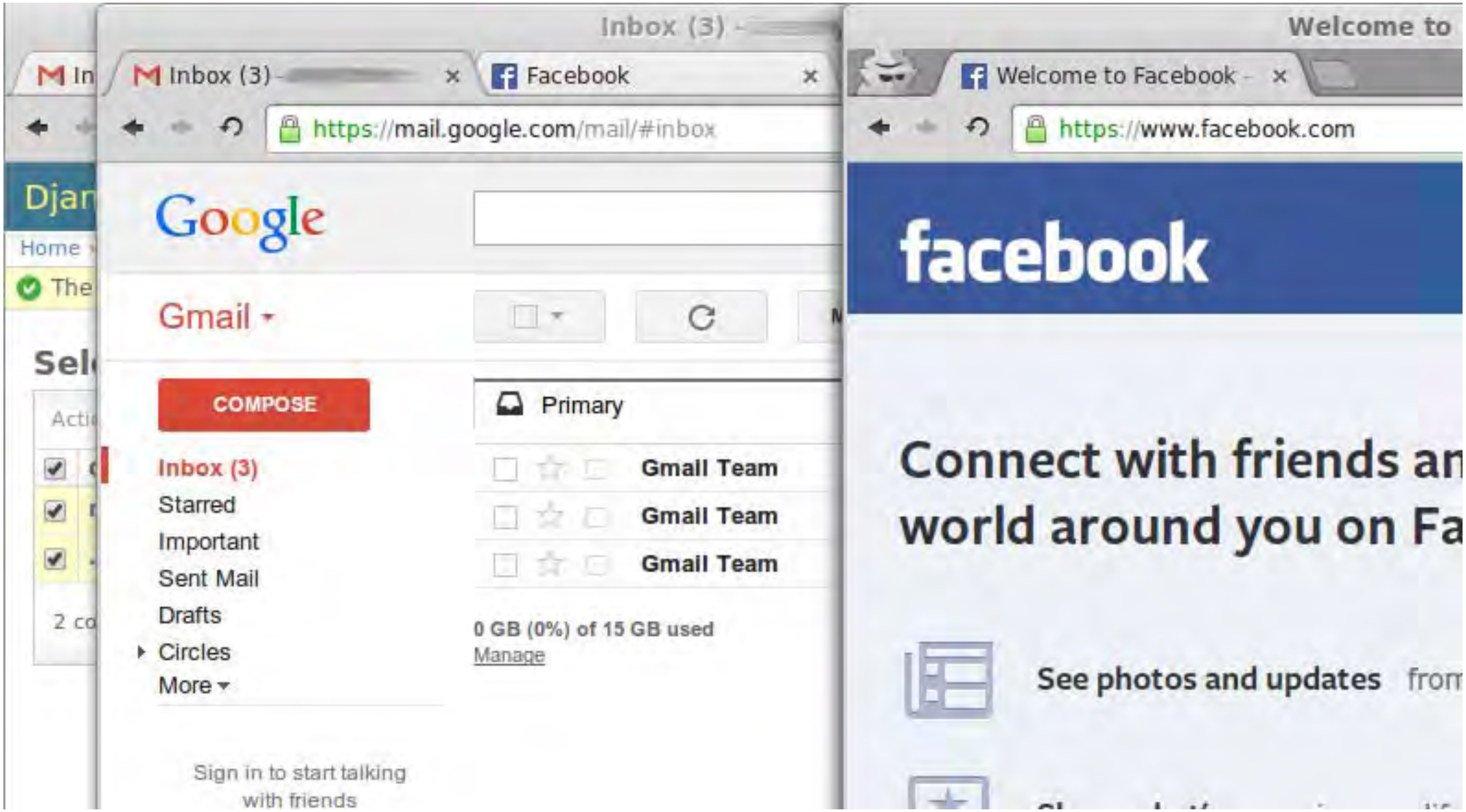
Select cookie to change

Action: Use cookies in a browser ▾ Go 2 of 2 selected

✓	Cookie
✓	mail.google.com :: GX; SID
✓	.facebook.com :: c_user; xs

2 cookies

Proof of Concept: Login



Defenses

Disk Encryption

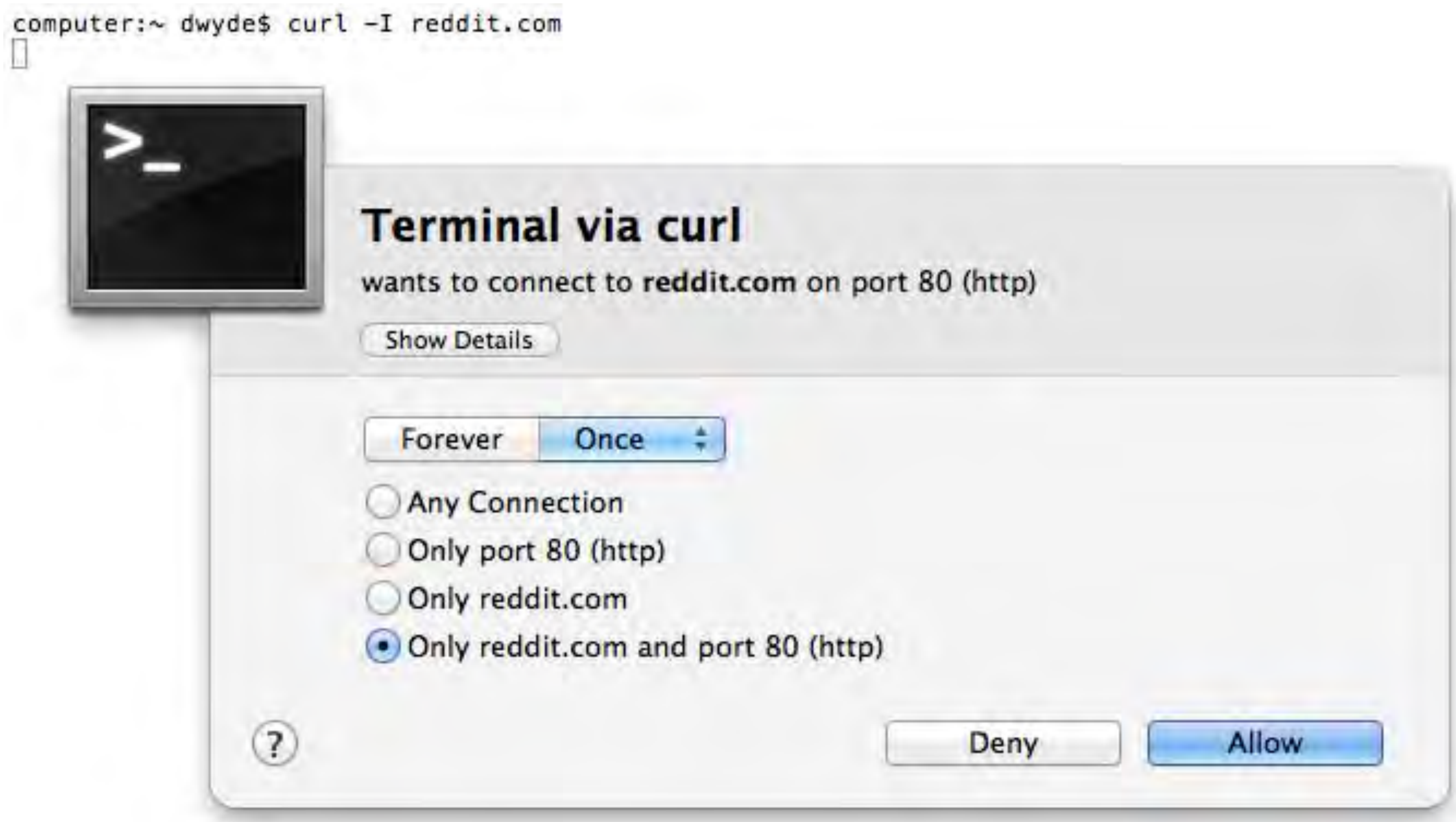
- Protect against physical access to plaintext cookies

Application Firewalls

- Block/allow (server, port) pairs for each application
  - Chromium can access www.google.com on port 443

- Examples
  - Mac: Little Snitch
  - Windows: NetLimiter?
  - Linux: ?

Little Snitch



```
computer:~ dwyde$ curl -I reddit.com
```

Terminal via curl

wants to connect to **reddit.com** on port 80 (http)

[✓ Once]

- Any Connection
- Only port 80 (http)
- Only reddit.com
- [✓ Only reddit.com and port 80 (http)]

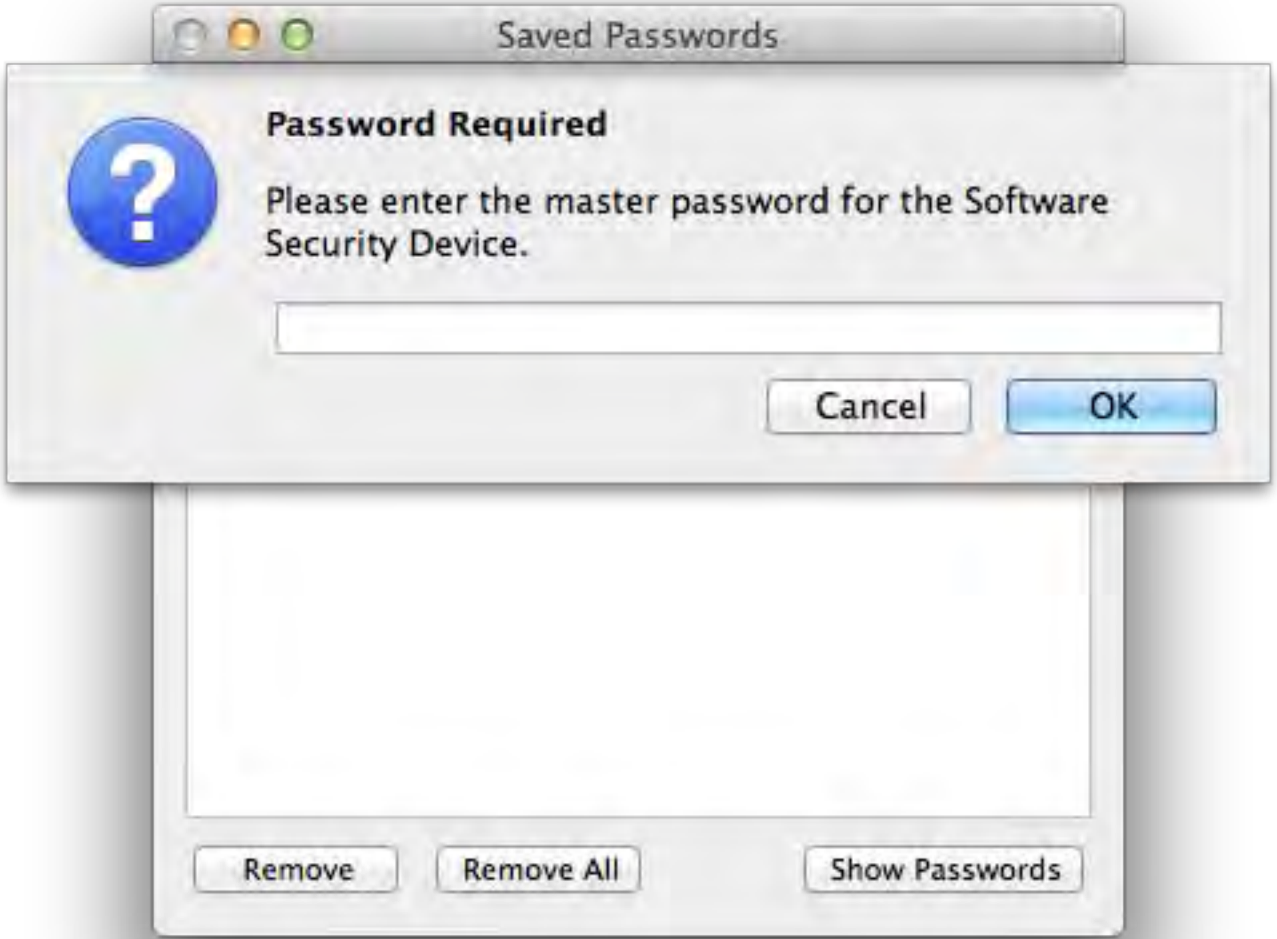
SELinux

- Security-Enhanced Linux
- Separate from standard Unix permissions
- Can isolate a user’s applications from each other

Idea: Master Password for Cookies

- Type in a password to decrypt your cookies
- Firefox has this to protect passwords

Firefox: Master Password



Password Required

Please enter the master password for the Software Security Device.

Server-Side Defenses

- Tie a session cookie to the login IP
  - The cPanel web hosting tool can optionally enforce this
  - Kind of annoying in a world of mobile clients

- Warn users, rather than force them to log in again
  - “You’ve logged in from X and Y countries this month”

Conclusions

- Cookies should be handled with care
- Client-side cookie security is not a solved problem

References

- Opera reader: https://gist.github.com/gwarser/1324501#file-readcookies-py
- Safari reader: http://www.securitylearn.net/2012/10/27/cookies-binarycookies-reader/
- Firefox master password: http://kb.mozillazine.org/Master_password
- cPanel cookie IP validation: http://www.cpanelkb.net/cpanel-security-settings-checklist/
- CryptProtectData (Microsoft documentation): http://msdn.microsoft.com/en-us/library/aa922939.aspx