MANICODE

SECURE CODING EDUCATION

Authentication and Session Management

JIM MANICO Secure Coding Instructor *www.manicode.com*

A little background dirt…

jim@manico.net

@manicode

- OWASP Global Board Member
- Project manager of the OWASP Cheat Sheet Series and several other OWASP projects
- 18+ years of software development experience
- Author of "Iron-Clad Java, Building Secure Web Applications" from McGraw-Hill/Oracle-Press
- Kauai, Hawaii Resident

ORACLE

Iron-Clad Java: Building Secure Web Applications

Best Practices for Secure Java Web Application Development

Jim Manico
August Detlefsen
Contributing Author, Kevin Kenan
Technical Editor, Milton Smith
Oracle Senior Principal Security Product Manager, Java

Oracle Press

WARNING: Please do not attempt to hack any computer system without legal permission to do so. Unauthorized computer hacking is illegal and can be punishable by a range of penalties including loss of job, monetary fines and possible imprisonment.

ALSO: The *Free and Open Source Software* presented in these materials are examples of good secure development tools and techniques. You may have unknown legal, licensing or technical issues when making use of *Free and Open Source Software*. You should consult your company's policy on the use of *Free and Open Source Software* before making use of any software referenced in this material.

Authentication: Where are we going?

- Session Management
- Transport Security
- Password Storage
- Multi-Factor Authentication
- Forgot Password Workflow

Question:
What is authentication?

Answer: Verification that an entity is who it claims to be

Question:
What is the difference between authentication and authorization?

Answer: Authentication verifies the identity of a user. Authorization checks if an entity has privileges to perform a function or action.

Question:
What is an authentication session?

Answer: A session is an area of memory or storage that tracks certain aspects of a users. An authenticated session tracks the status of a user who is "logged in" to your system. A session identifier (ID) is supplied to the entity once they are authenticated.

Sessions and Session IDs

- A session is created by an application server to track the state of authenticated users and visitors
- A session includes a area of memory or storage on the server and a session ID to refer to that server side session
- A session ID is a random, unique, and difficult to guess string ASEIUHF849J283JE874GSJWOD2374DDEOFEFK93423H
- Sessions and therefor session ID's are valid for a finite period of time
- Sessions are used by the application server on any subsequent request to verify the identity of the sender

Session IDs are a "key" to a portion of memory on the server where data and state can be stored for the corresponding active user!

More on Sessions

- In some applications, the session is once a user identifies/authenticates themself.
- In other applications, the session is initiated even for anonymous users on first page visit.
- Session ID's are typically passed between the browser and server in an HTTP Cookie.
- The session ID is often all that is needed to prove authentication for the rest of the session.
- Session management is usually handled by the web framework, making it transparent to the developer.

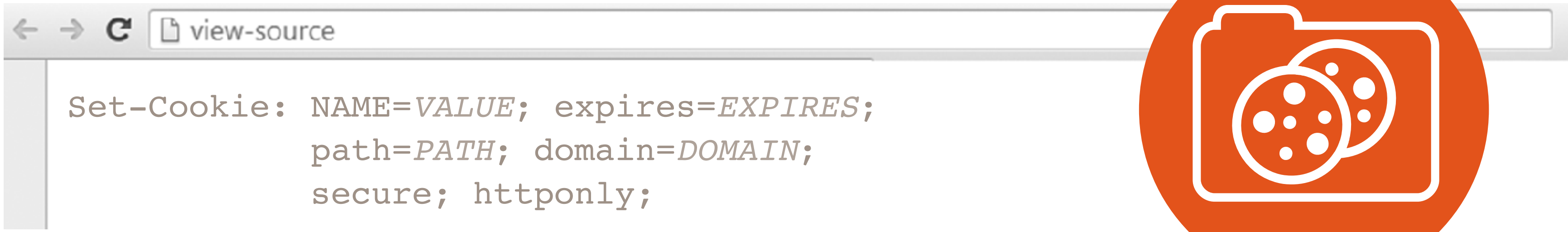
The session ID is often all that is needed to prove authentication for the rest of the session! We need to protect it!

Session Management Workflow

1. Start HTTPS, and deliver login form
2. Submit credentials
3. Create session, deliver cookie to user
4. Do cool things
5. Potential re-authentication
6. Logoff or idle session timeout
7. Absolute session timeout
8. Invalidate session

How do we manage cookies properly?

Cookie Options and Security



```
Set-Cookie: NAME=VALUE; expires=EXPIRES;
            path=PATH; domain=DOMAIN;
            secure; httponly;
```

Name	The name of the cookie parameter
Value	The parameter value
Expires	The date at which to discard the cookie. If absent, the cookie will not be persistent, and will be discarded when the browser is closed. If "-1", the cookie will be discarded immediately.
Domain	The domain that the cookie applies to
Path	The path that the cookie applies to
Secure	Indicates that the cookie can only be used over secure HTTPS. USE THIS!
HttpOnly	Indicates that the cookie can only be modified and accessed from the server. For example, JavaScript within the browser application will not be able to access the cookie. USE THIS FOR SESSION IDs!

Additional Cookie Security Defenses

- Avoid storing sensitive data in cookies
- Avoid using persistent cookies
- Any sensitive cookie data should be encrypted if not intended to be viewed/tampered by the user. Persistent cookie data not intended to be viewed by others should always be encrypted.
- Cookie values susceptible to tampering should be protected with an HMAC appended to the cookie, or a server-side hash in a session variable of the cookie contents.

So… what are some of the main attacks against authentication and session management mechanisms?

Authentication Dangers

Poor Password Management

- Stolen database revealing stored password data
- Brute force attack attempting many password guesses for a specific account
- Brute force attack attempting one password guess against many accounts: password123
- Simple password policy allowing faster guesses or unlimited guesses
- Password reuse: Attacks on one website effect others

Username Harvesting

- Registration page often makes this easy
- Leaked usernames and email addresses via timing attack

Weak "Forgot Password" Feature

- Plaintext password sent over email
- Reset links sent over email
- Original passwords sent over email

More Authentication Dangers

“Change Password” Feature

- Does not require existing password
- Allows for resetting of other users password
- Does not enforce good password policy

Session Management Dangers

- Forcing victims to use known session IDs (fixation)
- Weak or predictable session IDs
- Session Hijacking via XSS (HTTPOnly)
- Session Hijacking via network sniffing (secure cookie flag)
- Lack of session timeout; sessions that never expire

How do we deal with brute force attacks?

Brute Force Defense

Vertical

- Track TOTAL failed logins over time
- Detect when failed logins spike
- Rate limiting

Horizontal

- Multi-Factor authentication
- Account locking
- Obscure usernames
- Rate limiting
- Strong password policy

How do we protect usernames from being harvested?

Username Harvesting Attack Defense

- Send all usernames over well configured HTTPS/SSL/TLS.
- Develop generic failed login messages that do not indicate whether the user-id or password was incorrect, and implement timing-attack prevention.
- Ensure that good usernames and bad usernames take the same time to process for all login attempts.
  - Prevent Timing Attack
- Do not worry about this risk if your allow username verification via registration, forgot password or similar features.
- Consider making usernames obscure and assigned, instead of chosen by users.

When should we make our users re-authenticate?

Credential Security

- Credential security is used for authentication and re-authentication. It helps minimize CSRF and session hijacking attacks.
- Some of the actions that should require a user to provide their identity:
  - Login
  - Change Password
  - Change Email Address
  - Delete Account
  - Financial Transaction
  - Attestion
- Implement server-side enforcement of password syntax and strength
  - No common passwords
  - Minimum length
  - Numbers/Symbols
  - Uppercase/Lowercase

Find a balance. An overly strong policy is bad.

Password1!

Twitter Password Ban-List: August 2014

8675309
987654
nnnnnn
nop123
nop123
nopqrs
noteglh
npprff
npprff14
npgvba
nyoreg
nyoregb
nyrkvf
nyrwnaqen
nyrwnaqeb
nznaqn
nzngrhe
nzrevpn
naqern
naqerj
natryn
natryf
navzny
nagubal
ncbyyb
nccyrf

nefrany
neguhe
nfqstu
nfqstu
nfuyrl
nffubyr
nhthfg
nhfgva
onqobl
onvyrl
onanan
onearl
onfronyy
ongzna
orngevm
ornire
ornivf
ovtpbbx
ovtqnqql
ovtqvpx
ovtqbt
ovtgvgf
oveqvr
ovgpurf
ovgrzr
oynmre

oybaqr
oybaqrf
oybjwbo
oybjzr
obaq007
obavgn
obaavr
obbobb
obbtre
obbzre
obfgba
oenaqba
oenaql
oenirf
oenmvy
oebapb
oebapbf
ohyyqbt
ohfgre
ohggre
ohggurnq
pnyiva
pnzneb
pnzreba
pnanqn
pncgnva

pneybf
pnegre
pnfcre
puneyrf
puneyvr
purrfr
puryfrn
purfgre
puvpntb
puvpxra
pbpnpbyn
pbssrr
pbyyrtr
pbzcnd
pbzchgre
pbafhzre
pbbxvr
pbbcre
pbeirggr
pbjobl
pbjoblf
pelfgny
phzzvat
phzfubg
qnxbgn
qnyynf

qnavry
qnavryyr
qroovr
qraavf
qvnoyb
qvnzbaq
qbpgbe
qbttvr
qbycuva
qbycuvaf
qbanyq
qentba
qernzf
qevire
rntyr1
rntyrf
rqjneq
rvafgrva
rebgvp
rfgeryyn
rkgerzr
snypba
sraqre
sreenev
sveroveq
svfuvat

sybevqn
sybjre
sylref
sbbgonyy
sberire
serqql
serrqbz
shpxrq
shpxre
shpxvat
shpxzr
shpxlbh
tnaqnys
tngrjnl
tngbef
trzvav
trbetr
tvnagf
tvatre
tvmzbqb
tbyqra
tbysre
tbeqba
tertbel
thvgne
thaare

unzzre
unaanu
uneqpber
uneyrl
urngure
uryczr
uragnv
ubpxrl
ubbgref
ubearl
ubgqbt
uhagre
uhagvat
vprzna
vybirlbh
vagrearg
vjnagh
wnpxvr
wnpxfba
wnthne
wnfzvar
wnfcre
wraavsre
wrerzl
wrffvpn
wbuaal

wbuafba
wbeqna
wbfrcu
wbfuhn
whavbe
whfgva
xvyyre
xavtug
ynqvrf
ynxref
ynhera
yrngure
yrtraq
yrgzrva
yrgzrva
yvggyr
ybaqba
ybiref
znqqbt
znqvfba
znttvr
zntahz
znevar
znevcbfn
zneyobeb
znegva

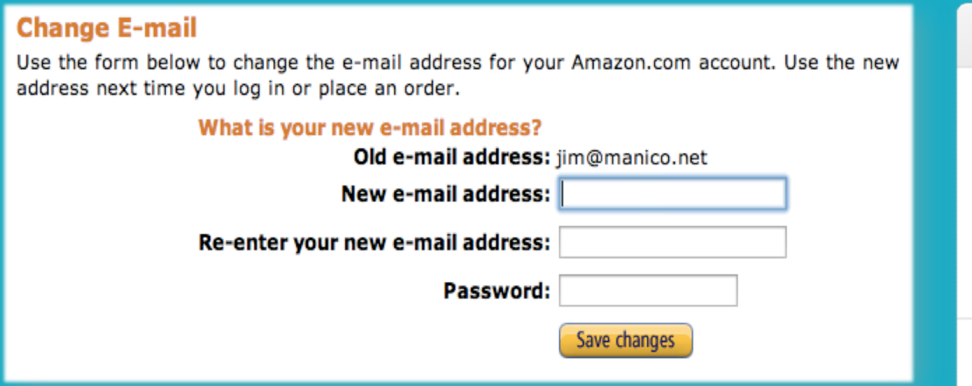
zneiva
znfgre
zngevk
znggurj
znirevpx
znkjryy
zryvffn
zrzore
zreprqrf
zreyva
zvpunry
zvpuryyr
zvpxrl
zvqavtug
zvyyre
zvfgerff
zbavpn
zbaxrl
zbaxrl
zbafgre
zbetna
zbgure
zbhagnva
zhssva
zhecul
zhfgnat

anxrq
anfpne
anguna
anhtugl
app1701
arjlbex
avpubynf
avpbyr
avccyr
avccyrf
byvire
benatr
cnpxref
cnagure
cnagvrf
cnexre
cnffjbeq
cnffjbeq
cnffjbeq1
cnffjbeq12
cnffjbeq123
cngevpx
crnpurf
crnahg
crccre
cunagbz

cubravk
cynlre
cyrnfr
cbbxvr
cbefpur
cevapr
cevaprff
cevingr
checyr
chffvrf
dnmjfk
djregl
djreglhv
enoovg
enpury
enpvat
envqref
envaobj
enatre
enatref
erorppn
erqfxvaf
erqfbk
erqjvatf
evpuneq
eboreg

eboregb
ebpxrg
ebfrohq
ehaare
ehfu2112
ehffvn
fnznagun
fnzzl
fnzfba
fnaqen
fnghea
fpbbol
fpbbgre
fpbecvb
fpbecvba
fronfgvna
frperg
frkfrk
funqbj
funaaba
funirq
fvreen
fvyire
fxvccl
fynlre
fzbxrl

Re-Authentication Examples



Change E-mail

Use the form below to change the e-mail address for your Amazon.com account. Use the new address next time you log in or place an order.

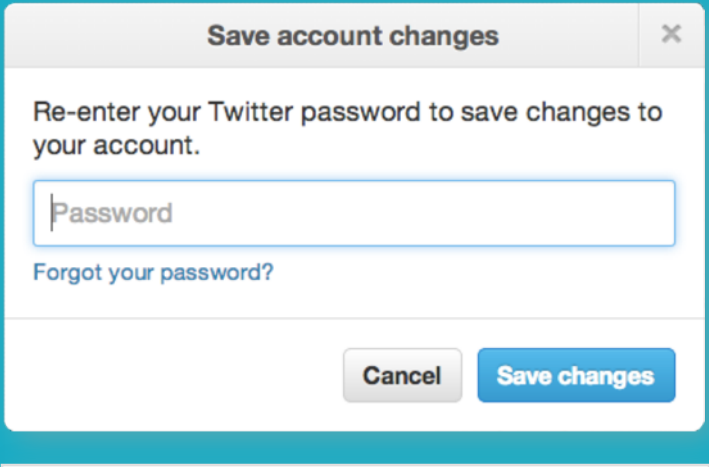
What is your new e-mail address?

Old e-mail address: jim@manico.net

New e-mail address:

Re-enter your new e-mail address:

Password:

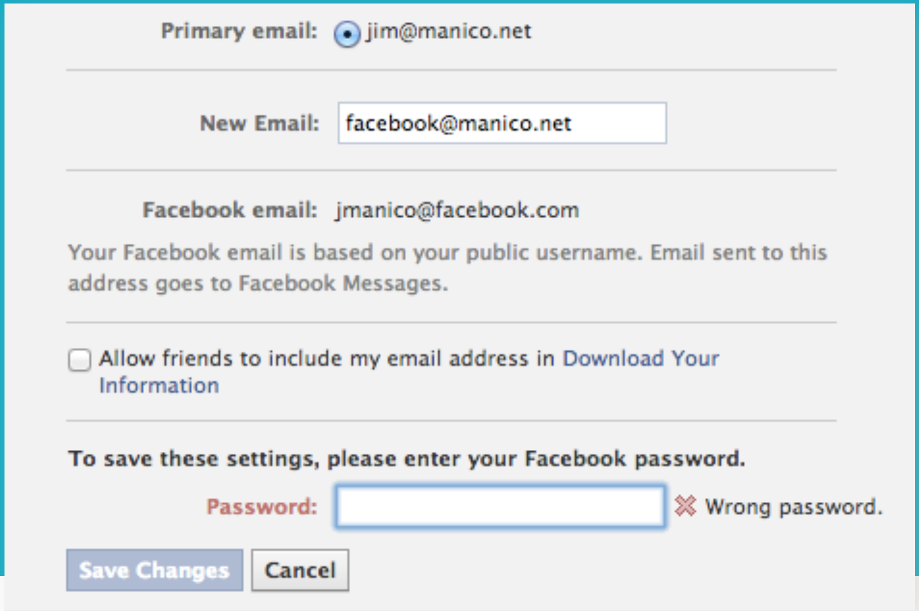


Save account changes

Re-enter your Twitter password to save changes to your account.

Password

Forgot your password?



Primary email: ● jim@manico.net

New Email: facebook@manico.net

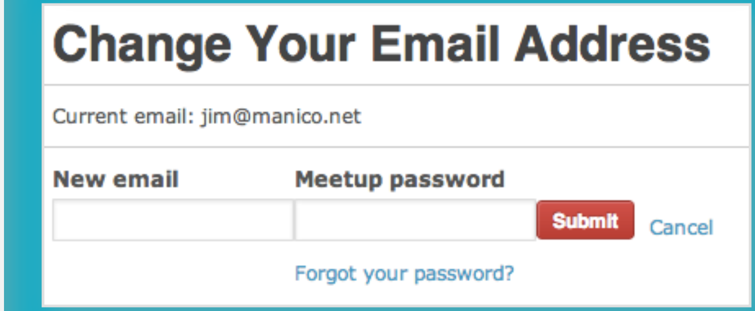
Facebook email: jmanico@facebook.com

Your Facebook email is based on your public username. Email sent to this address goes to Facebook Messages.

□ Allow friends to include my email address in Download Your Information

To save these settings, please enter your Facebook password.

Password: ☒ Wrong password.



Change Your Email Address

Current email: jim@manico.net

New email **Meetup password**

Forgot your password?

How do we deal with Session Fixation

Additional Session Defense

- Generate a new session ID at login to protect against session fixation.
- Disable URL session rewriting to protect against session fixation
- Example: Java/Tomcat 7
  - <session-config>
  - <tracking-mode>COOKIE</tracking-mode>
  - </session-config>
- Implement session timeouts and re-authentication to minimize session hijacking.

How do we deal with Logout correctly?

Logout/Session Defense

- Give users the option to log out of the application, and make the option available from every application page.
- When clicked, the logout option should prevent the user from requesting subsequent pages without re-authenticating to the application.
- The user's session should always be terminated during logout.
- JavaScript can be used to force logout during a window close event.

How should we store our users' passwords in the database?

Password Storage Defense Overview

Offline Attacks

- Avoid Hashing or Encryption
- Use proper key derivation functions and stretching configurations
- Use random and unique per-user salts
  - Less effective against targeted attacks, but use them anyhow
- Strict Password Policy
- Ban top X commonly used passwords

Online Attacks

- Ban top X commonly used passwords
- Rate limiting
- Multi-factor authentication
- Behavior Analysis
  - Trojan Combat
- Anti-Phishing
  - Early detection and takedown
- Good Network Security

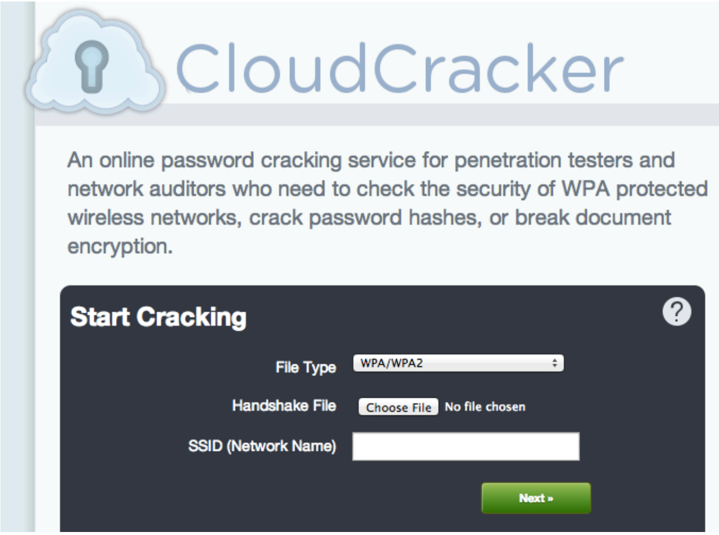
Reference: http://www.openwall.com/presentations

Estimated cost of hardware to crack password in 1 year

KDF	6 letters	8 letters	8 chars	10 chars	40-char text	80-char text
DES CRYPT	<$1	<$1	<$1	<$1	<$1	<$1
MD5	<$1	<$1	<$1	$1.1k	$1	$1.5T
MD5 CRYPT	<$1	<$1	$130	$1.1M	$1.4k	$\$1.5 \text{ x } 10^{15}$
PBKDF2 (100ms)	<$1	<$1	$18k	$160M	$200k	$\$2.2 \text{ x } 10^{17}$
Bcrypt (95 ms)	<$1	$4	$130k	$1.2B	$1.5M	$48B
Scrypt (64 ms)	<$1	$150	$4.8M	$43B	$52M	$\$6 \text{ x } 10^{19}$
PBKDF2 (5.0 s)	<$1	$29	$920k	$8.3B	$10M	$\$11 \text{ x } 10^{18}$
Bcrypt (3.0 s)	<$1	$130	$4.3M	$39B	$47M	$1.5T
Scrypt (3.8 s)	$900	$610k	$19B	$175T	$210B	$\$2.3 \text{ x } 10^{23}$

- Research by Colin Percival, https://www.tarsnap.com/scrypt/scrypt.pdf, *STRONGER KEY DERIVATION VIA SEQUENTIAL MEMORY-HARD FUNCTIONS*

Let’s Get Crackin’!



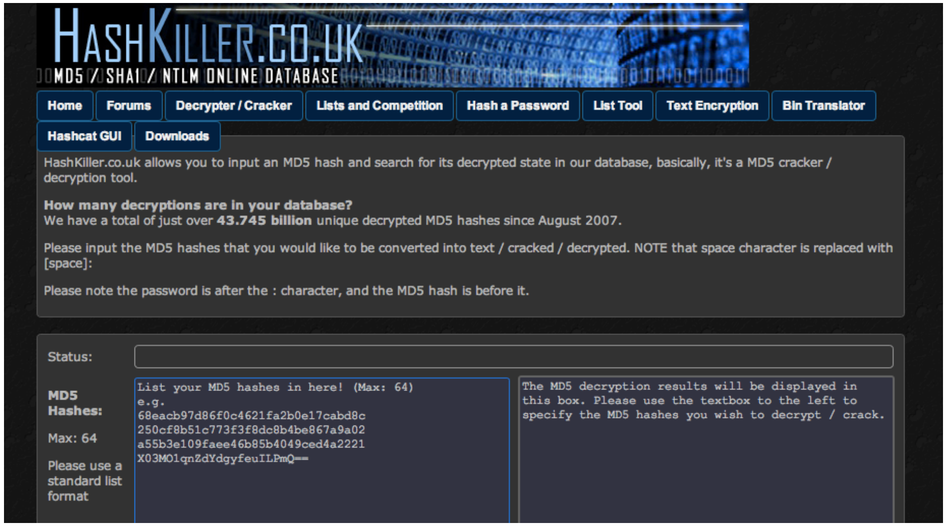
An online password cracking service for penetration testers and network auditors who need to check the security of WPA protected wireless networks, crack password hashes, or break document encryption.

Start Cracking

File Type: WPA/WPA2

Handshake File: No file chosen

SSID (Network Name):



HashKiller.co.uk allows you to input an MD5 hash and search for its decrypted state in our database, basically, it's a MD5 cracker / decryption tool.

How many decryptions are in your database?
We have a total of just over **43.745 billion** unique decrypted MD5 hashes since August 2007.

Please input the MD5 hashes that you would like to be converted into text / cracked / decrypted. NOTE that space character is replaced with [space]:

Please note the password is after the : character, and the MD5 hash is before it.

Status:

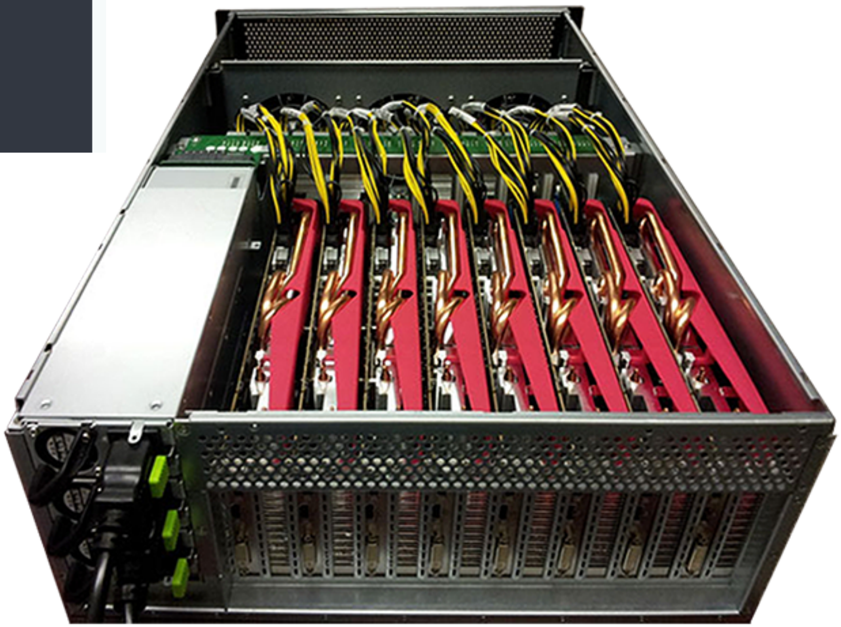
MD5 Hashes:

Max: 64

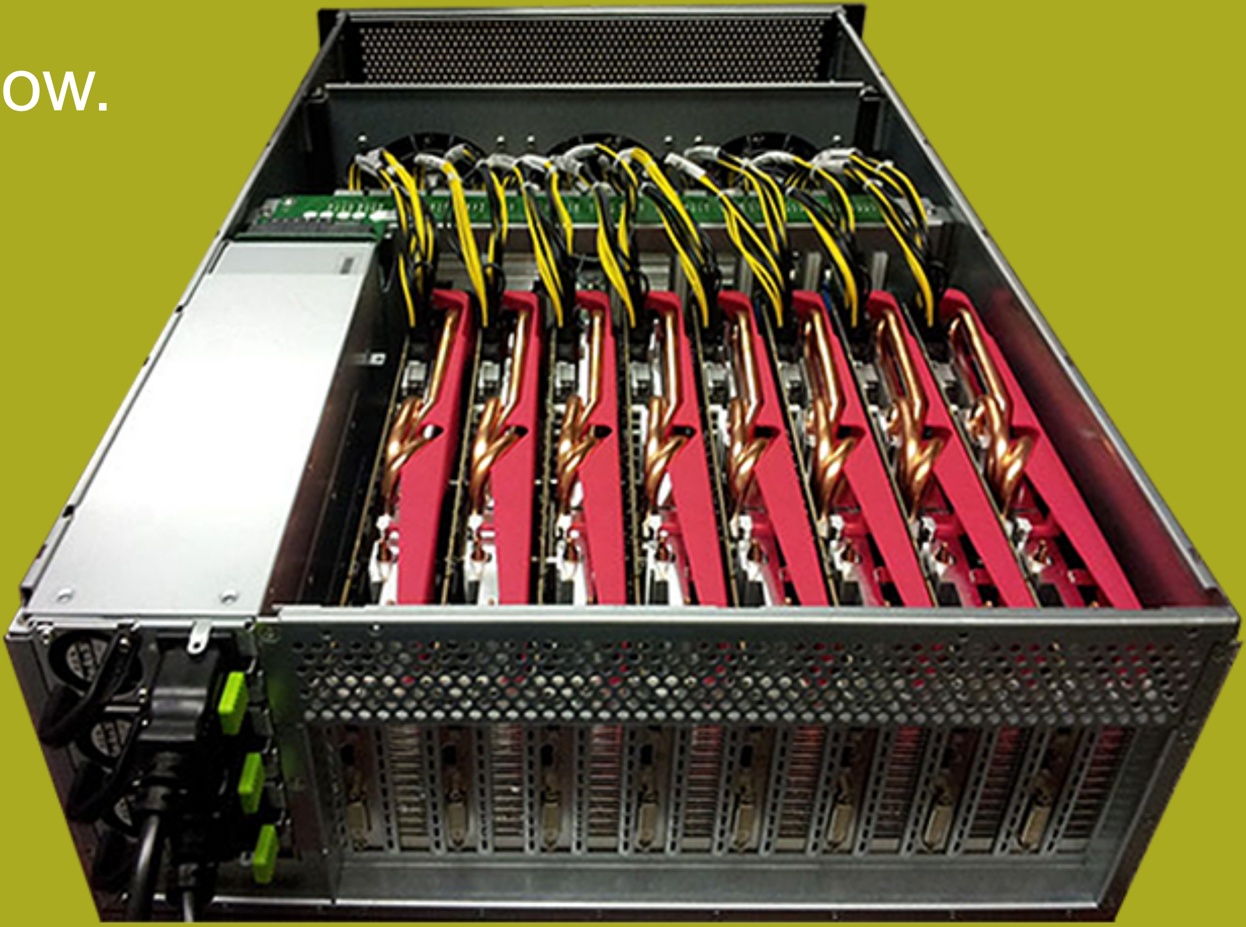
Please use a standard list format

```
List your MD5 hashes in here! (Max: 64)
e.g.
68eacb97d86f0c4621fa2b0e17cabd8c
250cf8b51c773f3f8dc8b4be867a9a02
a55b3e109faee46b85b4049ced4a2221
X03MO1qnZdYdgyfeuILPmQ==
```

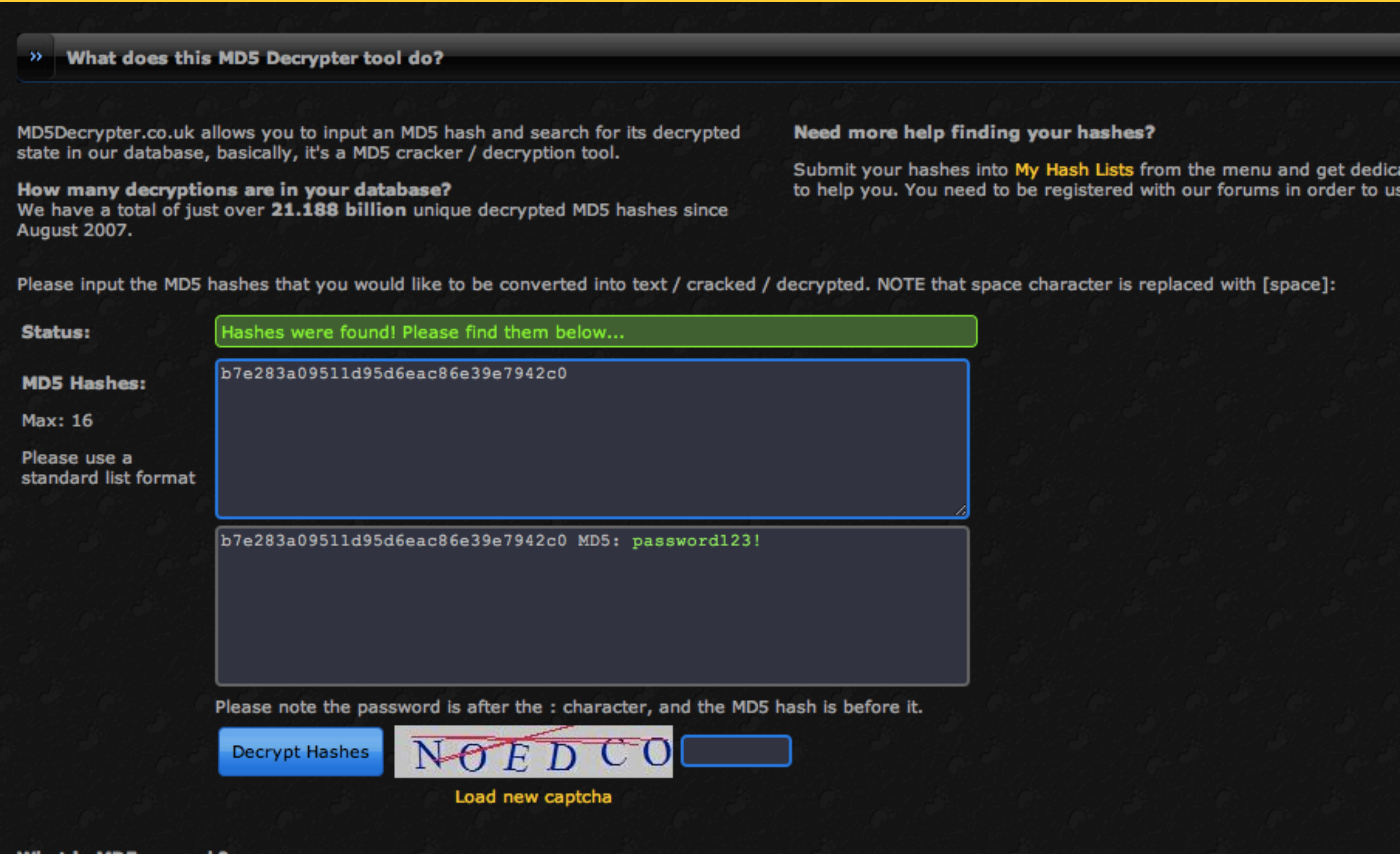
```
The MD5 decryption results will be displayed in
this box. Please use the textbox to the left to
specify the MD5 hashes you wish to decrypt / crack.
```



Wow.
Just… wow.



http://arstechnica.com/security/2012/12/25-gpu-cluster-cracks-every-standard-windows-password-in-6-hours



» What does this MD5 Decrypter tool do?

MD5Decrypter.co.uk allows you to input an MD5 hash and search for its decrypted state in our database, basically, it's a MD5 cracker / decryption tool.

How many decryptions are in your database?
We have a total of just over **21.188 billion** unique decrypted MD5 hashes since August 2007.

Need more help finding your hashes?

Submit your hashes into My Hash Lists from the menu and get dedic
to help you. You need to be registered with our forums in order to us

Please input the MD5 hashes that you would like to be converted into text / cracked / decrypted. NOTE that space character is replaced with [space]:

Status: Hashes were found! Please find them below...

MD5 Hashes:

Max: 16

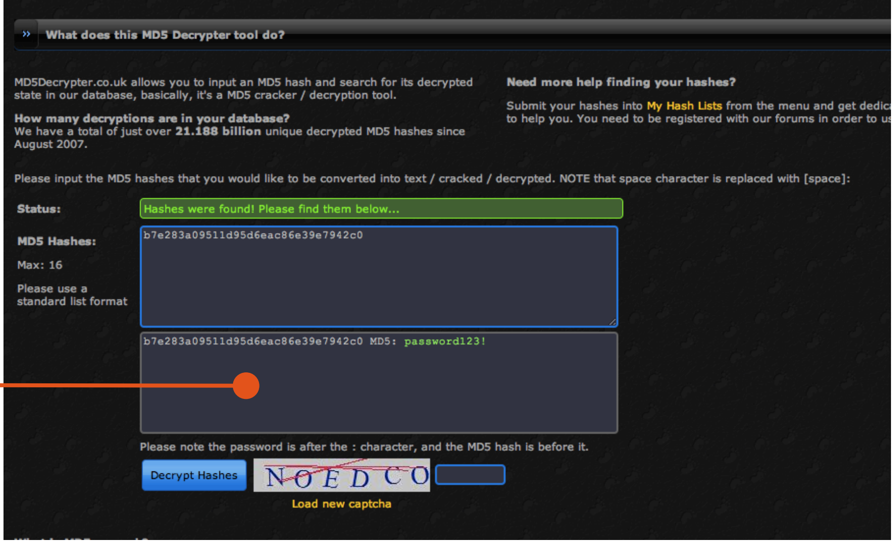
Please use a standard list format

```
b7e283a09511d95d6eac86e39e7942c0
```

```
b7e283a09511d95d6eac86e39e7942c0 MD5: password123!
```

Please note the password is after the : character, and the MD5 hash is before it.

Online Hashcracking Services



MD5Decrypter.co.uk allows you to input an MD5 hash and search for its decrypted state in our database, basically, it's a MD5 cracker / decryption tool.

How many decryptions are in your database?
We have a total of just over **21.188 billion** unique decrypted MD5 hashes since August 2007.

Need more help finding your hashes?

Submit your hashes into My Hash Lists from the menu and get dedica
to help you. You need to be registered with our forums in order to us

Please input the MD5 hashes that you would like to be converted into text / cracked / decrypted. NOTE that space character is replaced with [space]:

Status: Hashes were found! Please find them below...

MD5 Hashes:

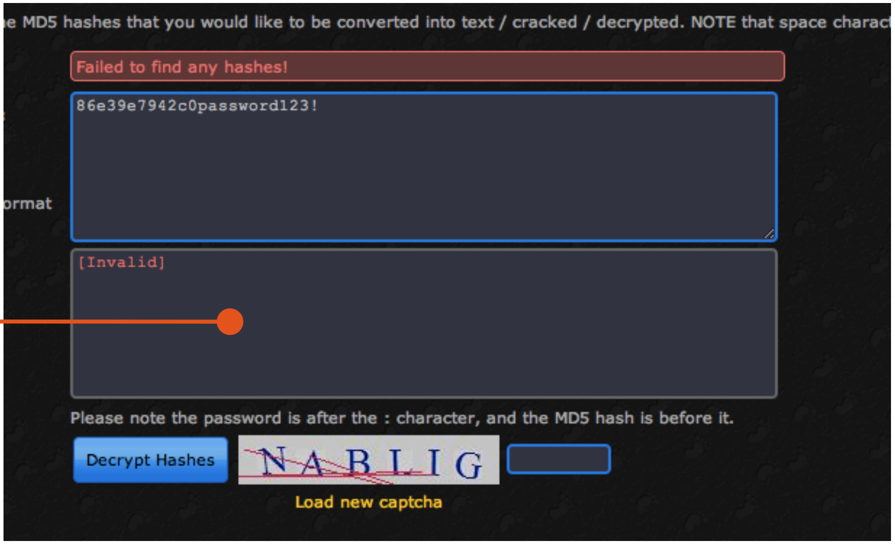
Max: 16

Please use a standard list format

```
b7e283a09511d95d6eac86e39e7942c0
```

```
b7e283a09511d95d6eac86e39e7942c0 MD5: password123!
```

Please note the password is after the : character, and the MD5 hash is before it.



e MD5 hashes that you would like to be converted into text / cracked / decrypted. NOTE that space charact

Failed to find any hashes!

```
86e39e7942c0password123!
```

```
[Invalid]
```

Please note the password is after the : character, and the MD5 hash is before it.

md5("86e39e7942c0password123!") = f3acf5189414860a9041a5e9ec1079ab

md5("password123!") = b7e283a09511d95d6eac86e39e7942c0

Basic Password Defenses

Disable browser autocomplete

- Chrome, Opera, and IE11+ will ignore the autocomplete attribute for password fields.

```
<form autocomplete="off">
  <input autocomplete="off">
</form>
```

Only send passwords over HTTPS POST Body

```
<form action="https://mybank.example/" method="POST">
```

Never display password in the browser

```
<input type="password">
```

Store passwords so that they are quickly verifiable and are not reversible

– Use a salt – Use SCRYPT/PBKDF2 – Use HMAC

Password Storage Best Practices

1

Do not limit the characters or length of user password

2

Do not allow users to use common passwords

3

Use a user-specific random and unique salt

4

Use BCRYPT, SCRYPT or PBKDF2

5

Store passwords as an HMAC + good key management as an alternative

1

Do Not Limit the Password Strength

- Limiting passwords to protect against injection is doomed to failure
- Use proper encoding and other defenses instead
- Very long passwords can cause DoS
- Do not allow common passwords

Password1!

2

Use a User-Specific Salt

- Protect (salt, password);
- Use a 32+ byte salt
- Do not depend on hiding, splitting, or otherwise obscuring the salt
- Consider hiding, splitting or otherwise obscuring the salt anyway as a extra layer of defense
- Salt should be both cryptographically random AND unique per user!

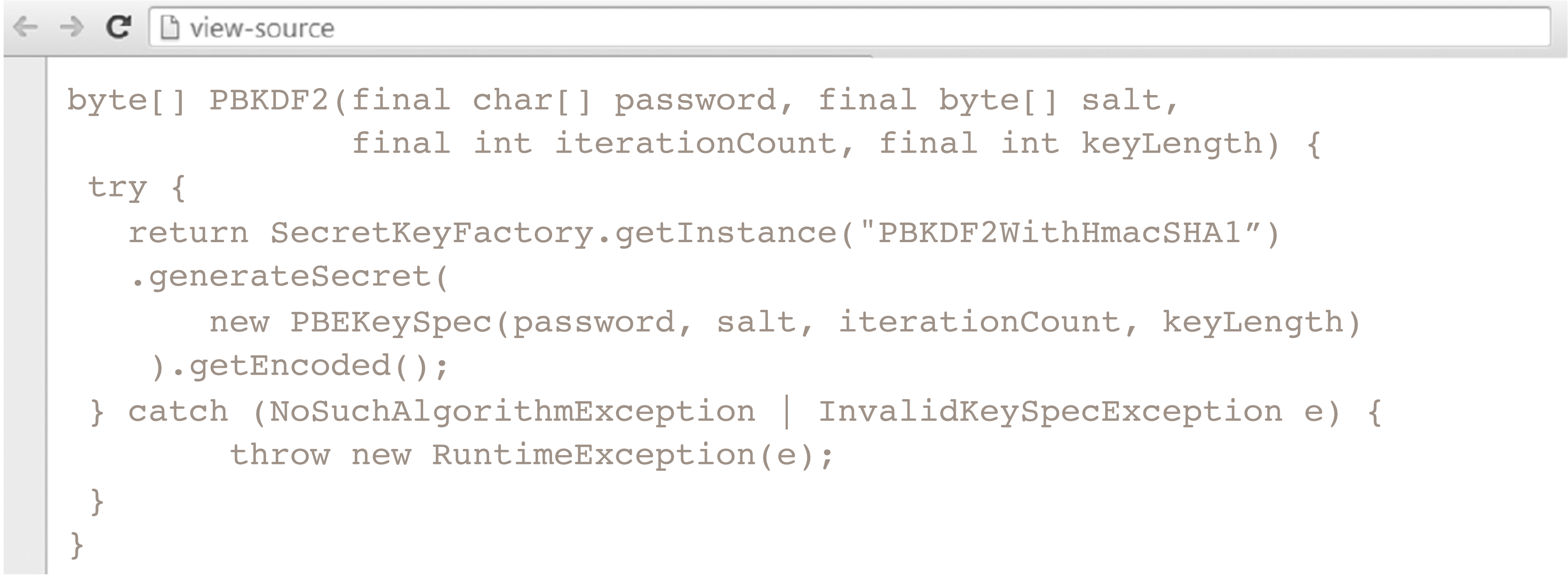
3

Leverage an Adaptive KDF

- PBKDF2 (salt, password,128000);
- PBKDF2 when FIPS certification or enterprise support on many platforms is required
- bcrypt where resisting most hardware accelerated attacks is necessary but enterprise support isn’t
- scrypt where resisting any/all hardware accelerated attacks is necessary but enterprise support isn’t

Imposes difficult verification on the attacker
and defender!

Java 7 PBKDF2

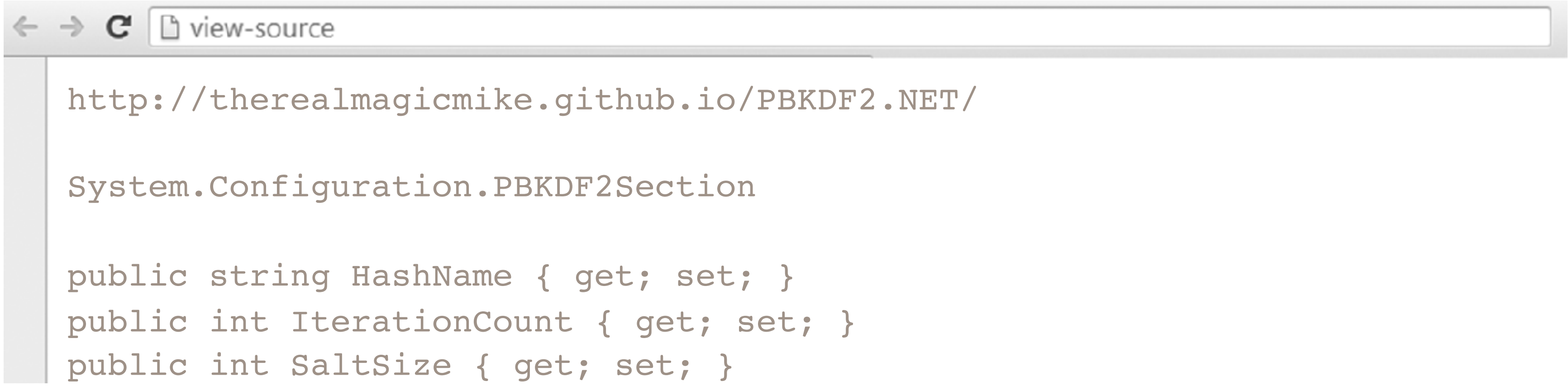


```
byte[] PBKDF2(final char[] password, final byte[] salt,
              final int iterationCount, final int keyLength) {
 try {
   return SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1”)
   .generateSecret(
       new PBEKeySpec(password, salt, iterationCount, keyLength)
    ).getEncoded();
 } catch (NoSuchAlgorithmException | InvalidKeySpecException e) {
        throw new RuntimeException(e);
 }
}
```

keyLength: length of HmacSHA1

iterationCount: 128,000 at LEAST (2014)

.NET PBKDF2



```
http://therealmagicmike.github.io/PBKDF2.NET/

System.Configuration.PBKDF2Section

public string HashName { get; set; }
public int IterationCount { get; set; }
public int SaltSize { get; set; }
```

```
keyLength: length of HmacSHA1
iterationCount: 128,000 at  LEAST (2014)
hashName: PBKDF2-HMAC-SHA-512
```

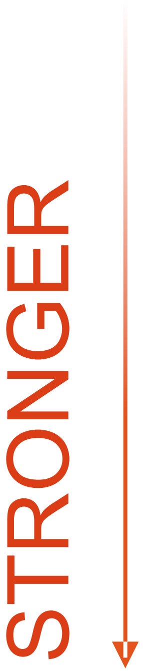
Bcrypt in PHP

- string password_hash
  (string $password , integer $algo [, array $options])
- Uses the bcrypt algorithm (default as of PHP 5.5.0)

bcrypt in .NET

- https://www.nuget.org/packages/BCrypt-Official/

GPU Attacks on Modern Password Hashes



STRONGER

PBKDF2-HMAC-SHA-1

PBKDF2-HMAC-SHA-256

PBKDF2-HMAC-SHA-512

Bcrypt

scrypt

Reference: Openwall and http://www.openwall.com/presentations/

ASIC/FPGA Attacks on Modern Password Hashes

STRONGER

PBKDF2-HMAC-SHA-1

PBKDF2-HMAC-SHA-256

PBKDF2-HMAC-SHA-512

scrypt below 16 MB

bcrypt (uses 4 KB)

scrypt at 16 MB

scrypt above 32 MB

Reference: Openwall and http://www.openwall.com/presentations/

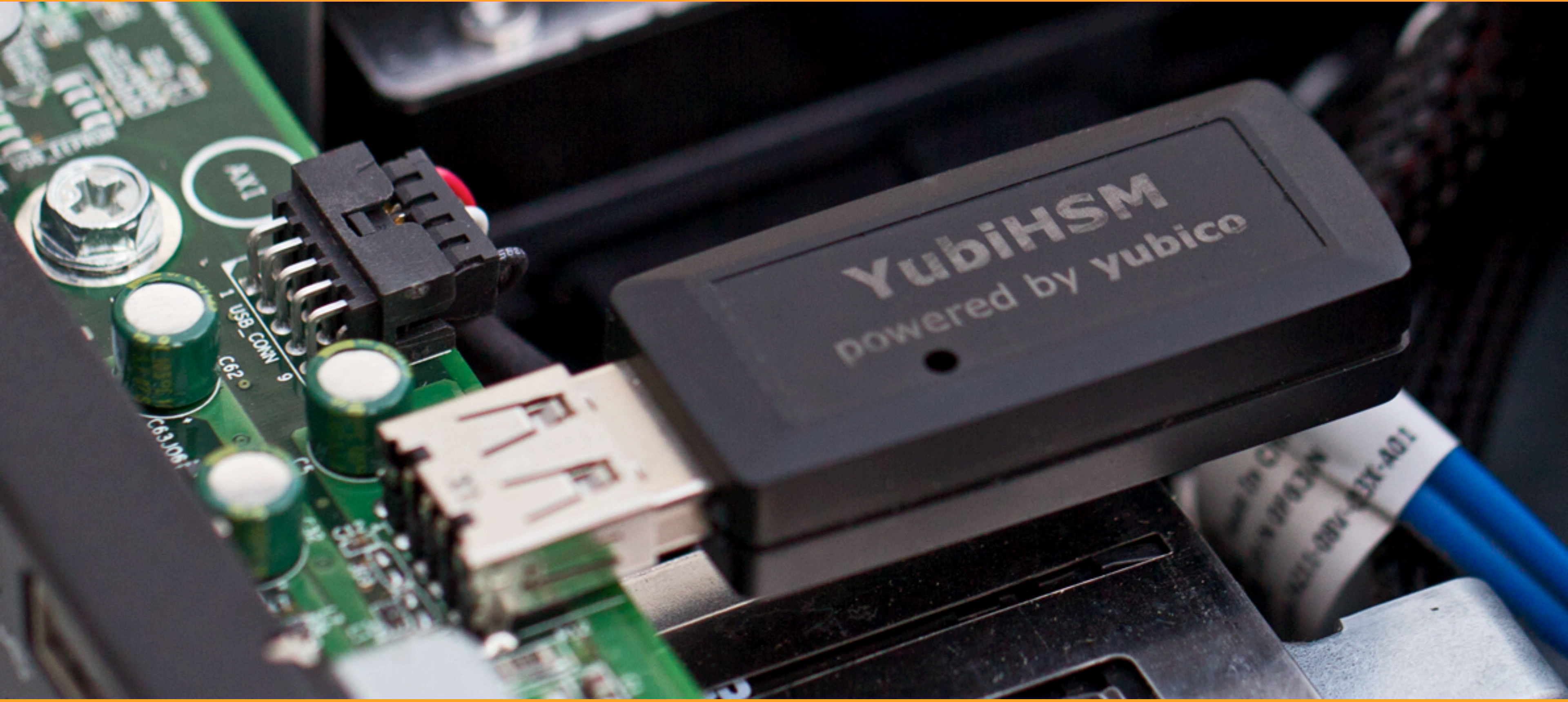
4

Leverage Keyed Protection Solution

- HMAC-SHA-256([key], [salt] + [credential])
- Protect this key as any private key using best practices
- Store the key outside the credential store
- Isolate this process outside of your application layer

Imposes difficult verification on the *attacker only!*

YubiHSM: a USB Dongle for Servers



YubiHSM in a server's internal USB port.

Photo © Yubico, reproduced under the fair use doctrine.

HMAC's in Action for YubiHSM

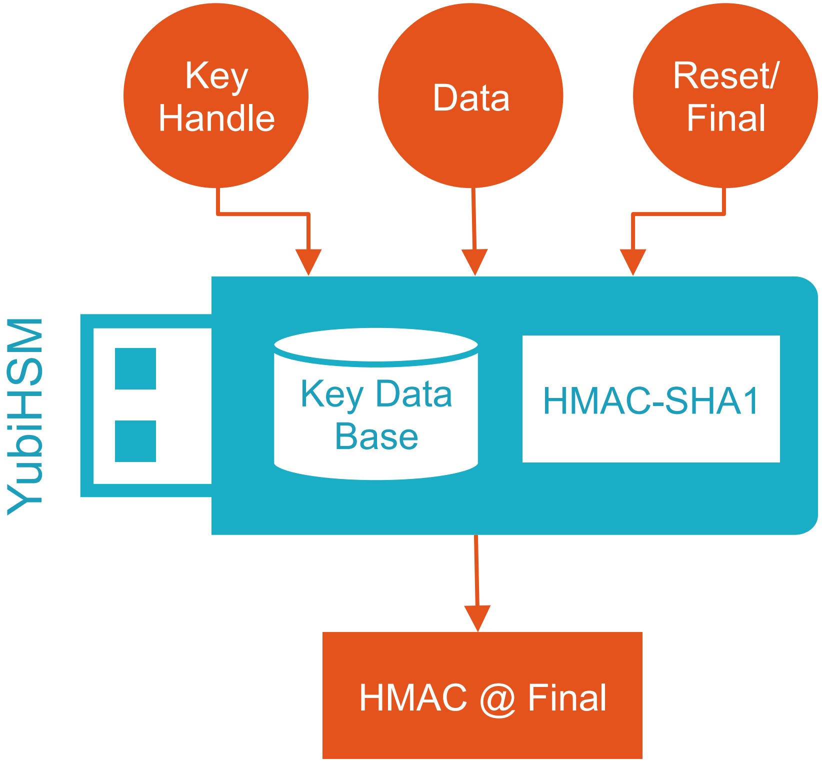


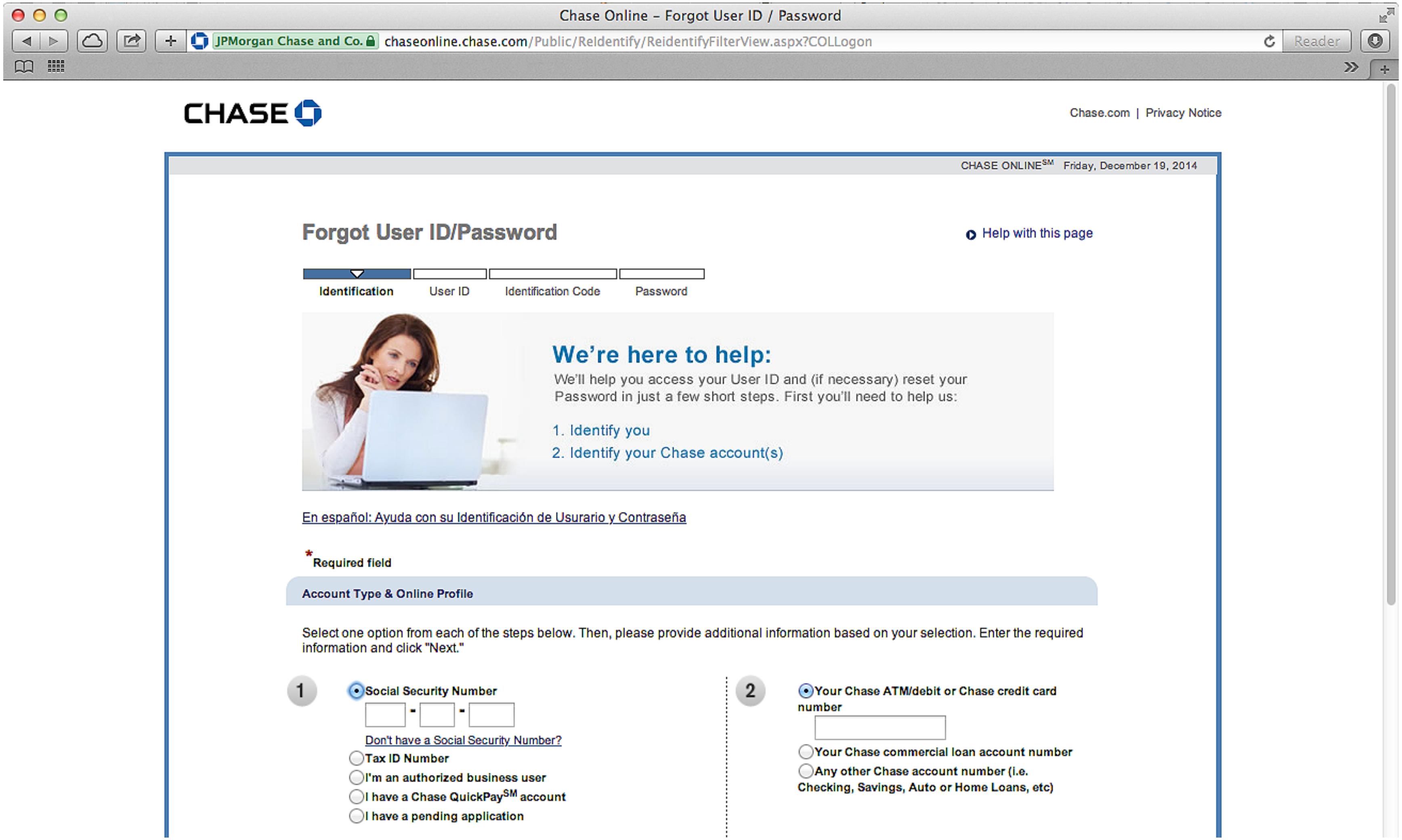
Diagram © Yubico, reproduced under the fair use doctrine.

- KEY for HMAC stored in local key database only, not retrievable
- Key handle is the HSM ID
- Data is password or KDF of Password
- HMAC @ Final is final computed password hash

Forgot Password Secure Design

Require identity question	▪ Last name, account number, email, Social Security #, DOB ▪ Enforce lockout policy or throttling
Ask one or more good security questions	▪ https://www.owasp.org/index.php/Choosing_and_Using_Security_Questions_Cheat_Sheet
Send the user a randomly generated token via out-of-band communication	▪ SMS, mobile app or dedicated multi-factor token
Verify code in *same web session*	▪ Enforce lockout policy
Change password	▪ For more info see https://www.owasp.org/index.php/Forgot_Password_Cheat_Sheet

Example of Forgotten Password



Chase Online – Forgot User ID / Password
JPMorgan Chase and Co.
chaseonline.chase.com/Public/ReIdentify/ReidentifyFilterView.aspx?COLLogon
Reader
CHASE
Chase.com | Privacy Notice

CHASE ONLINE^{SM} Friday, December 19, 2014

Forgot User ID/Password

Help with this page

Identification | User ID | Identification Code | Password

We're here to help:

We'll help you access your User ID and (if necessary) reset your Password in just a few short steps. First you'll need to help us:

1. Identify you
2. Identify your Chase account(s)

En español: Ayuda con su Identificación de Usurario y Contraseña

*Required field

Account Type & Online Profile

Select one option from each of the steps below. Then, please provide additional information based on your selection. Enter the required information and click "Next."

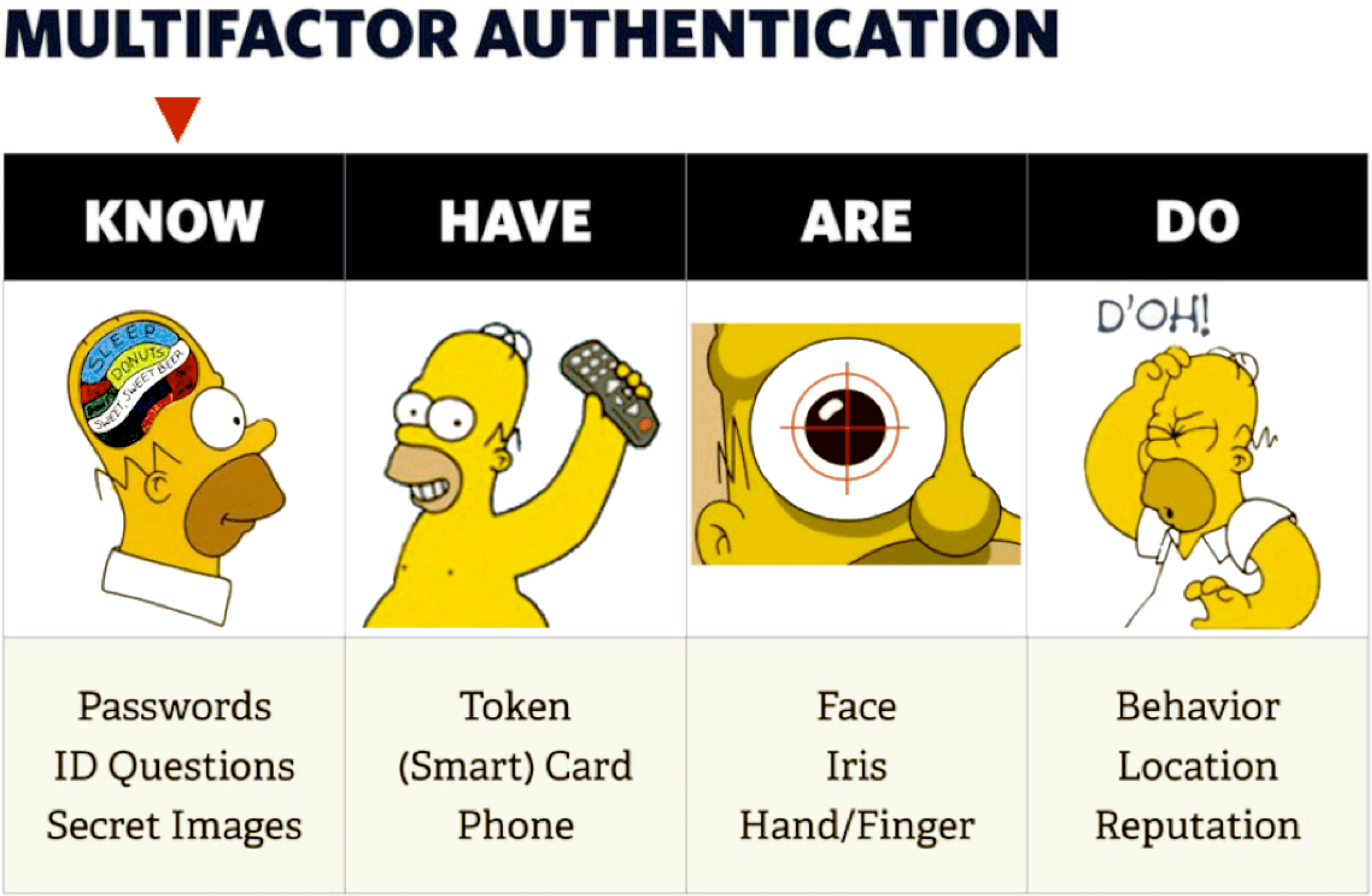
1

● Social Security Number
□ - □ - □
Don't have a Social Security Number?
○ Tax ID Number
○ I'm an authorized business user
○ I have a Chase QuickPay^{SM} account
○ I have a pending application

2

● Your Chase ATM/debit or Chase credit card number
□
○ Your Chase commercial loan account number
○ Any other Chase account number (i.e. Checking, Savings, Auto or Home Loans, etc)

Multi-Factor Authenticaion



MULTIFACTOR AUTHENTICATION

KNOW	HAVE	ARE	DO
Passwords ID Questions Secret Images	Token (Smart) Card Phone	Face Iris Hand/Finger	Behavior Location Reputation

A slide from "Modern Two-Factor Authentication: Defending Against User-Targeted Attacks" by Dug Song and Jon Oberheide, Duo Security, 2012

2000+MFA Goes Mainstream

- Many online services and especially banks start to treat trojans and phishing seriously
- They deployed 2-factor authentication where passwords are augmented with one-time codes or some other second factor
- Passwords remain relevant as one factor
- But is MFA effective?

“"Two factor authentication isn't our savior. It won't defend against phishing. It wont protect against identity theft. It's not going to secure accounts from fraudulent transactions. It solves the problems we had ten years ago, not today".” ***— Bruce Scneier***

- "The Future of Password Hashing" – Password-hashing.net

Multi-Factor Authentication

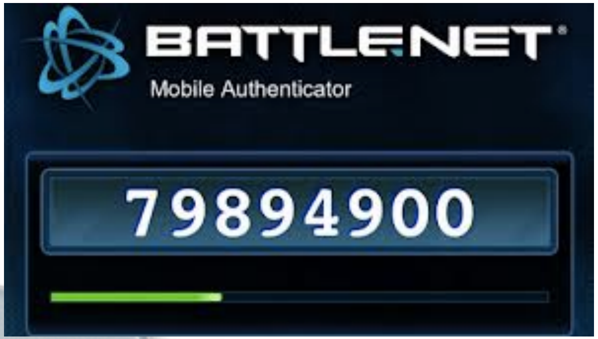
- There are 3 methods of identifying an individual

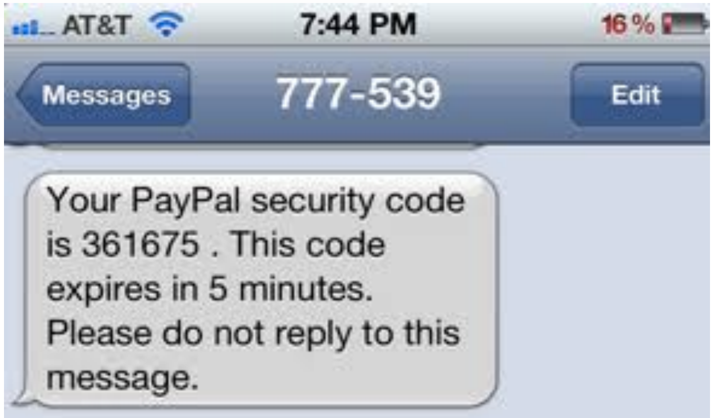
  Something you have – e.g. token, certificate, cell

  Something you are – e.g. biometrics

  Something you know – e.g. password.
- Protects against brute force attacks, minimizes impact of password theft
- Financial services applications are moving towards "stronger authentication"
- Google/Facebook/World-Of-Warcraft support consumer-centric multi-factor authentication

Multi-factor Token Generation Options





Your PayPal security code is 361675 . This code expires in 5 minutes. Please do not reply to this message.

Multi-Factor Authentication

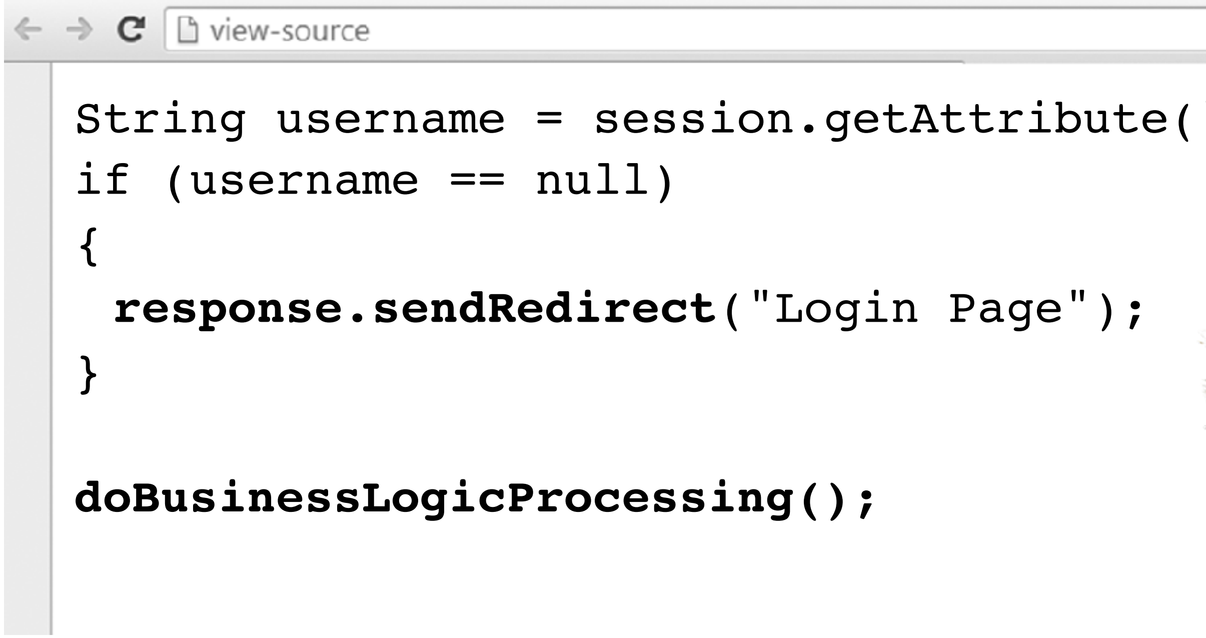
Changed Channel: [1. General - Redridge Mountains]
Changed Channel: [3. LocalDefense - Redridge Mountains]
[Mallice] has come online.

http://twofactorauth.org

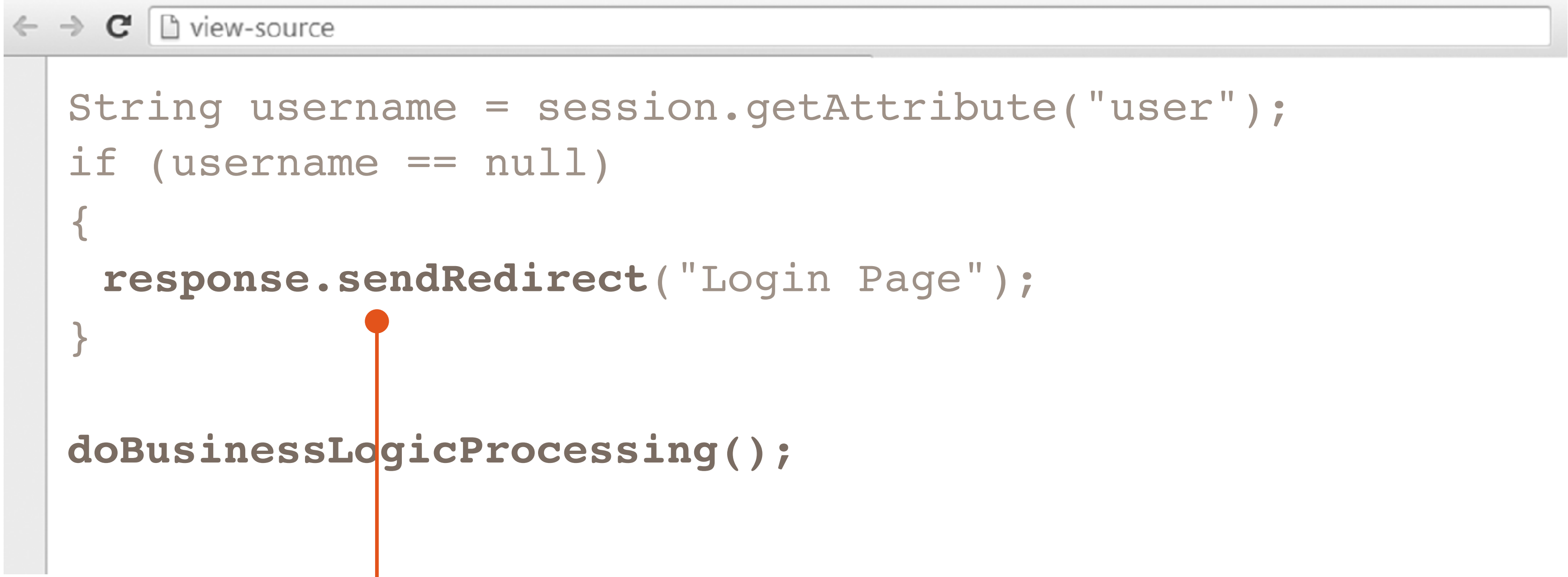
- Google
- Facebook
- PayPal
- Apple
- AWS
- Dropbox
- Twitter
- Blizzard's Battle.Net
- Valve's Steam
- Yahoo

Authentication Control Flow Flaws

Does this code look safe to you?



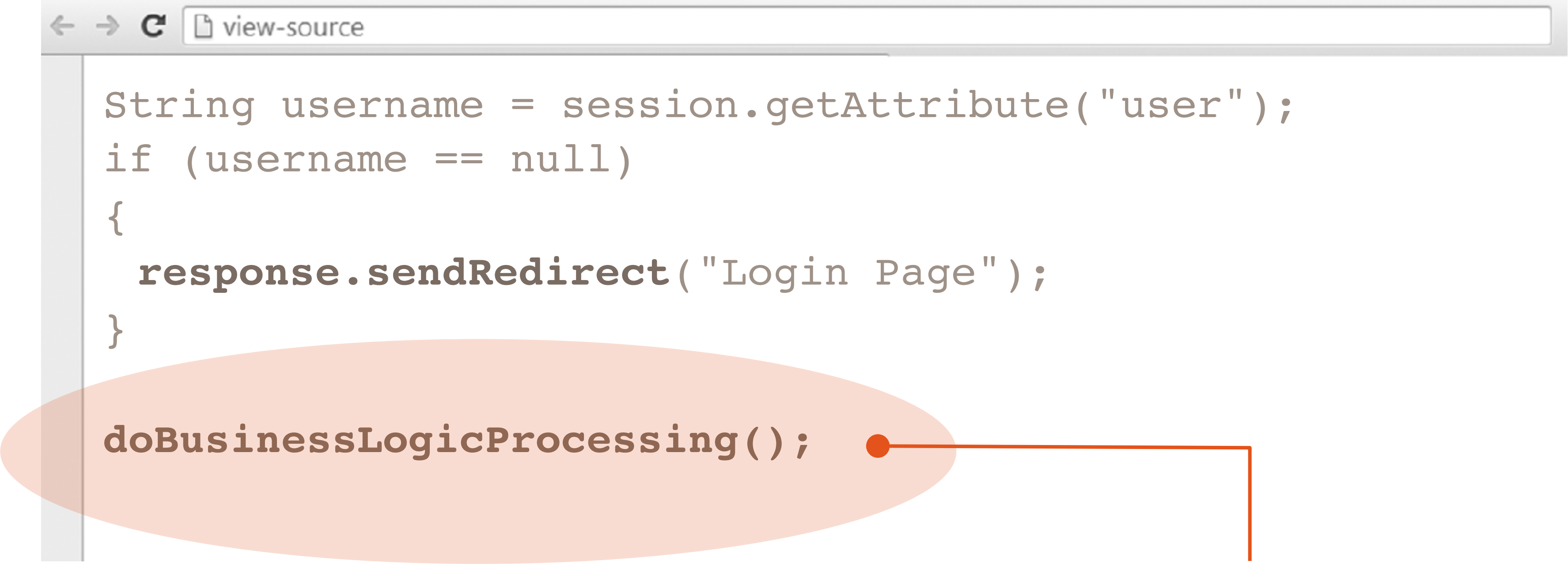
```
String username = session.getAttribute("user");
if (username == null)
{
  response.sendRedirect("Login Page");
}

doBusinessLogicProcessing();
```



```
String username = session.getAttribute("user");
if (username == null)
{
  response.sendRedirect("Login Page");
}

doBusinessLogicProcessing();
```

What if the execution did not stop here?

Business logic would execute for an unauthenticated request



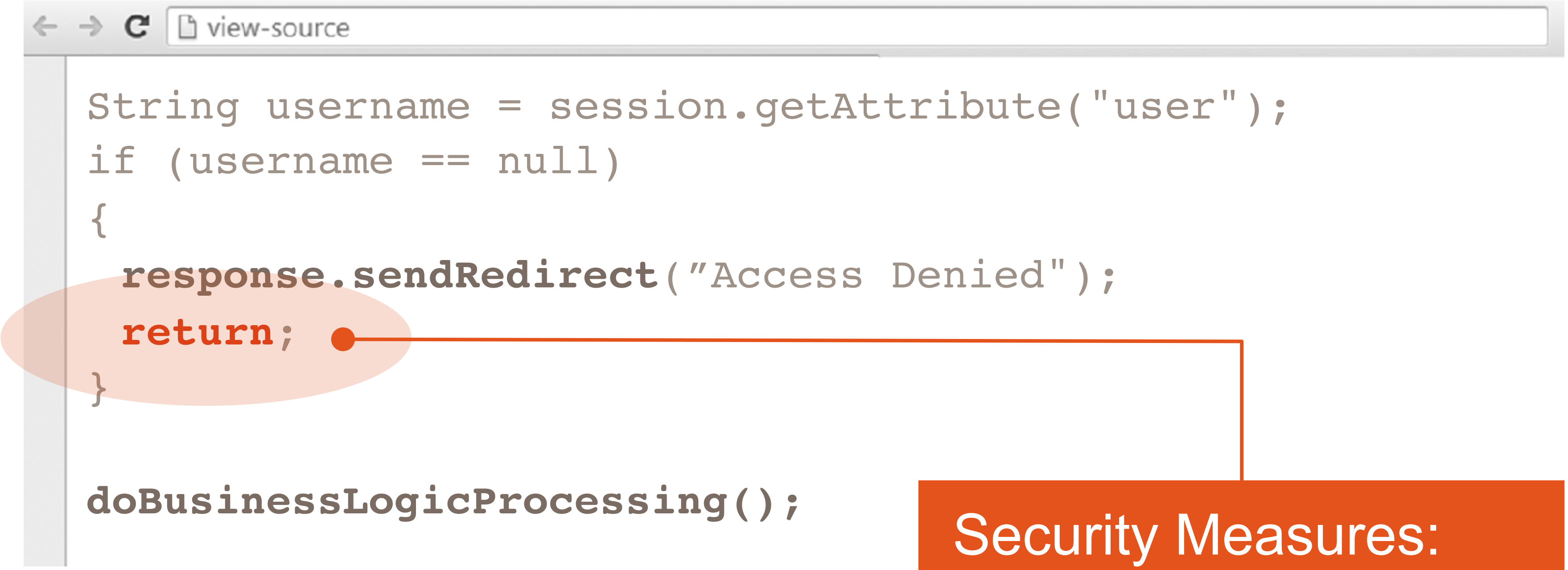
```
String username = session.getAttribute("user");
if (username == null)
{
  response.sendRedirect("Login Page");
}

doBusinessLogicProcessing();
```

This is
NOT PROTECTED

What does this mean?

- The execution flow does not stop after the *response.sendRedirect* call
- Entire page is processed and then the user is redirected to error page
- Thus, the business logic remains unprotected

Return after redirecting



```
String username = session.getAttribute("user");
if (username == null)
{
  response.sendRedirect(”Access Denied");
  return;
}

doBusinessLogicProcessing();
```

Security Measures: Terminate the execution flow after redirection call.z

ASVS 2 Authentication Requirements

ASVS 2 Authentication Requirements: Easy to Discover

V2.1 Verify all pages and resources require authentication except those specifically intended to be public (Principle of complete mediation).

V2.2 Verify all password fields do not echo the user's password when it is entered.

V2.4 Verify all authentication controls are enforced on the server side.

V2.6 Verify all authentication controls fail securely to ensure attackers cannot log in.

V2.16 Verify that credentials, and all other identity information handled by the application(s), do not traverse unencrypted or weakly encrypted links.

V2.17 Verify that the forgotten password function and other recovery paths do not reveal the current password and that the new password is not sent in clear text to the user.

V2.18 Verify that username enumeration is not possible via login, password reset, or forgot account functionality.

V2.19 Verify there are no default passwords in use for the application framework or any components used by the application (such as "admin/password").

ASVS 2 Authentication Requirements: Intermediate Part 1

V2.7 Verify password entry fields allow or encourage the use of passphrases, and do not prevent long passphrases or highly complex passwords being entered, and provide a sufficient minimum strength to protect against the use of commonly chosen passwords.

V2.8 Verify all account identity authentication functions (such as registration, update profile, forgot username, forgot password, disabled / lost token, help desk or IVR) that might regain access to the account are at least as resistant to attack as the primary authentication mechanism.

V2.9 Verify users can safely change their credentials using a mechanism that is at least as resistant to attack as the primary authentication mechanism.

V2.12 Verify that all authentication decisions are logged. This should include requests with missing required information, needed for security investigations.

V2.13 Verify that account passwords are salted using a salt that is unique to that account (e.g., internal user ID, account creation) and use bcrypt, scrypt or PBKDF2 before storing the password.

ASVS 2 Authentication Requirements: Intermediate Part 2

V2.20 Verify that a resource governor is in place to protect against vertical (a single account tested against all possible passwords) and horizontal brute forcing (all accounts tested with the same password e.g. "Password1"). A correct credential entry should incur no delay. Both these governor mechanisms should be active simultaneously to protect against diagonal and distributed attacks.

V2.21 Verify that all authentication credentials for accessing services external to the application are encrypted and stored in a protected location (not in source code).

V2.22 Verify that forgot password and other recovery paths send a link including a time-limited activation token rather than the password itself. Additional authentication based on soft-tokens (e.g. SMS token, native mobile applications, etc.) can be required as well before the link is sent over.

V2.23 Verify that forgot password functionality does not lock or otherwise disable the account until after the user has successfully changed their password. This is to prevent valid users from being locked out.

V2.24 Verify that there are no shared knowledge questions/answers (so called "secret" questions and answers).

V2.25 Verify that the system can be configured to disallow the use of a configurable number of previous passwords.

ASVS 2 Authentication Requirements: Advanced

V2.5 Verify all authentication controls (including libraries that call external authentication services) have a centralized implementation.

V2.26 Verify re-authentication, step up or adaptive authentication, SMS or other two factor authentication, or transaction signing is required before any application-specific sensitive operations are permitted as per the risk profile of the application.

OWASP
The Open Web Application Security Project

Application Security Verification Standard (2014)

Web Application Standard

Creative Commons (CC) Attribution Share-Alike
Free version at http://www.owasp.org

ASVS 2 Session Management Requirements: Easy to Discover

V3.1 Verify that the framework's default session management control implementation is used by the application.

V3.2 Verify that sessions are invalidated when the user logs out.

V3.3 Verify that sessions timeout after a specified period of inactivity.

V3.5 Verify that all pages that require authentication to access them have logout links.

V3.6 Verify that the session id is never disclosed other than in cookie headers; particularly in URLs, error messages, or logs. This includes verifying that the application does not support URL rewriting of session cookies.

V3.14 Verify that authenticated session tokens using cookies sent via HTTP, are protected by the use of "HttpOnly".

V3.15 Verify that authenticated session tokens using cookies are protected with the "secure" attribute and a strict transport security header (such as Strict-Transport-Security: max-age=60000; includeSubDomains) are present.

ASVS 2 Session Management Requirements: Intermediate

V3.4 Verify that sessions timeout after an administratively-configurable

V3.7 Verify that the session id is changed on login to prevent session fixation.

V3.8 Verify that the session id is changed upon re-authentication.

V3.10 Verify that only session ids generated by the application framework are recognized as valid by the application.

V3.11 Verify that authenticated session tokens are sufficiently long and random to withstand session guessing attacks.

V3.12 Verify that authenticated session tokens using cookies have their path set to an appropriately restrictive value for that site. The domain cookie attribute restriction should not be set unless for a business requirement, such as single sign on.

V3.16 Verify that the application does not permit duplicate concurrent user sessions, originating from different machines.

Conclusion

Authentication: Summary

- Session Management
- Transport Security
- Password Storage
- Multi-Factor Authentication
- Forgot Password Workflow

It’s been a pleasure.

jim@manicode.com

Basic MFA Considerations

Where do you send the token?

- Email (worst)
- SMS (ok)
- Mobile native app (good)
- Mobile native app, push notification (great)
- Dedicated token (ideal)
- Printed Tokens (interesting)

How do you handle thick clients?

- Email services, for example
- Dedicated and strong per-app passwords

How do you handle unavailable MFA devices?

- Printed back-up codes
- Fallback mechanism (like email)
- Call in center

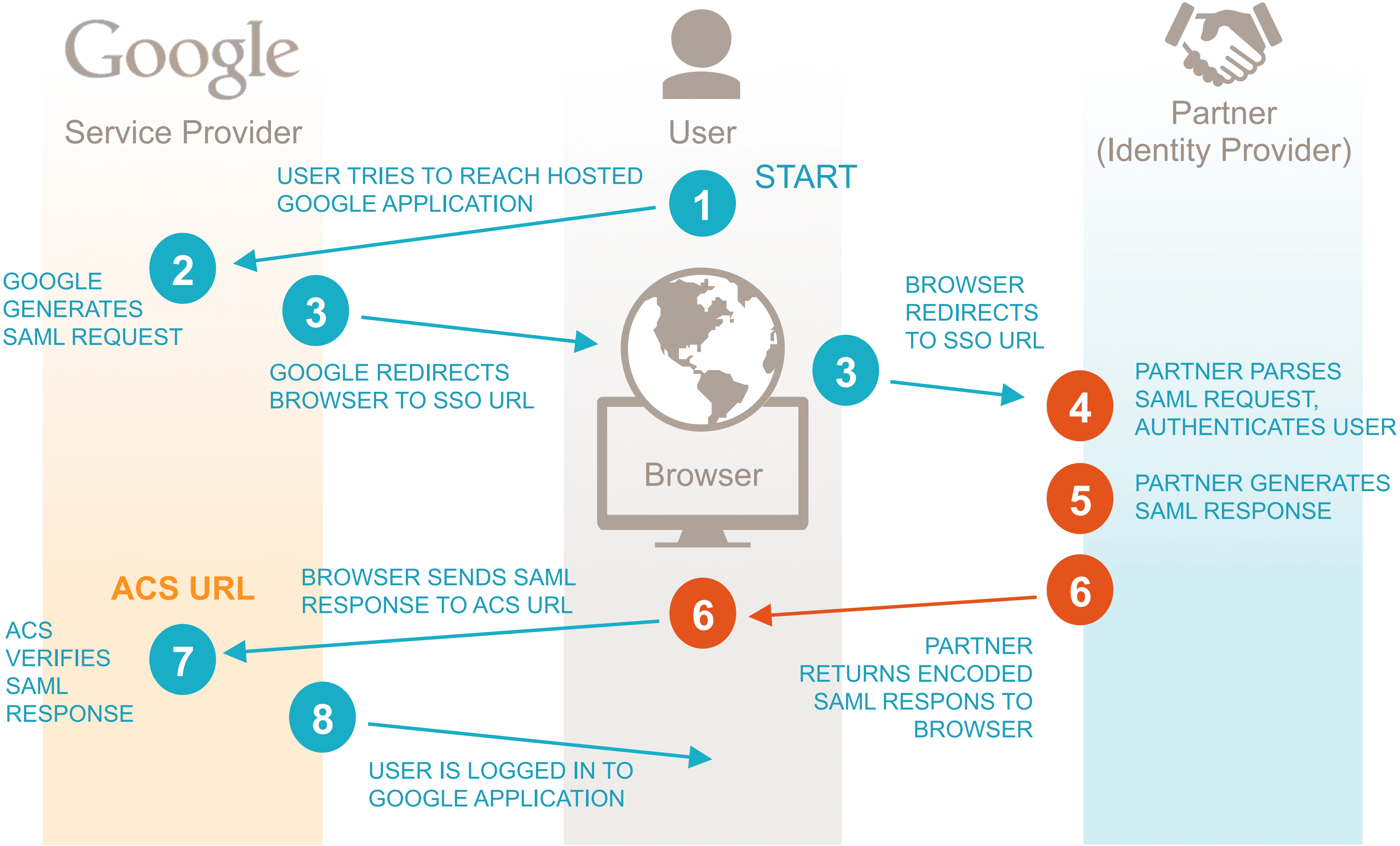
How do you handle mobile apps?

When is MFA not useful in mobile app scenarios?

Federated Identity and SAML

- XML-based identity management between different businesses
- Centralized Authentication Authority
- Single Sign-on / Log-out
- Assertions and Subjects
- Authentication Assertion Types
- Attribute Assertion Types
- Entitlement Assertion Types

SAML Transaction Steps


Google
Service Provider
User
Partner
(Identity Provider)
START
1
USER TRIES TO REACH HOSTED GOOGLE APPLICATION
2
GOOGLE GENERATES SAML REQUEST
3
GOOGLE REDIRECTS BROWSER TO SSO URL
Browser
3
BROWSER REDIRECTS TO SSO URL
4
PARTNER PARSES SAML REQUEST, AUTHENTICATES USER
5
PARTNER GENERATES SAML RESPONSE
6
PARTNER RETURNS ENCODED SAML RESPONS TO BROWSER
6
BROWSER SENDS SAML RESPONSE TO ACS URL
ACS URL
7
ACS VERIFIES SAML RESPONSE
8
USER IS LOGGED IN TO GOOGLE APPLICATION

Source: https://developers.google.com/google-apps/sso/saml_workflow_vertical.gif