Ongoing Authentication Weaknesses

Steve Dispensa
Marsh Ray

PhoneFactor

What is authentication?

Wikipedia: *The process of proving that something is genuine or authentic*

Has been used to describe authentication of:

- People
- Machines
- Processes

Native authentication: biometrics

- The oldest of them all: your immune system!
  - Recognizes “you” and attacks “not you”

- Voice recognition
  - Babies know their mothers
  - Mothers can pick out babies’ cries
  - Telephone authentication – you know who you’re talking to

- Facial recognition
  - So built-in that we don’t think of it as authentication
  - But: Man Who Mistook His Wife for a Hat (Oliver Sachs)

Literary Authentication

Shakespeare described an authentication problem in Hamlet:

- King Claudius ships Hamlet off to England with a sealed request to the King to execute Hamlet on arrival
  - Tamper-evident; relies on the carrier not wanting to appear to have tampered with the cargo
- Hamlet un-seals the request in-flight (er, on-water)
- He forges a new letter asking for the execution of Guildenstern and Rosencrantz
- And seals it with an expired signet
  - revoked certificate?

Source Address Authentication

- Basis of some firewall ingress rules
- Most webservers do SA auth
  - Tivoli Access Manager, for another example
  - Internal users get better access
- Other daemons do this - 127.0.0.1
  - Local user, remote UDP service, …
- rhosts/hosts.equiv
- Only works as well as it does for TCP

How reliable is SA auth?

- Extends trust to every router on the path
  - Servers doing SA validation therefore trust every router on the path to every source address they're interested in
  - Incidentally, assumes that these addresses remain in the same topological place => unwarranted assumption

- The attack is simple
  - Any device that can see traffic can impersonate
  - Satellite guy from bh

- Reverse proxy problem (vpn gateways)
  - Anything in the DMZ is now considered semi-trusted; things like reverse proxies, nats, etc., break this assumption
  - Any kind of proxy that allows localhost proxying

Device authentication

- Two varieties:
  - "weak" - based on heuristics, etc
  - "strong" - TPM, etc.

- Ideally: The user proves he is authorized because he is submitting a transaction from a known-authorized device

- Really: Device authentication proves:
  - The device is on
  - It is connected to the internet
  - Someone who wants the transaction to complete is in control of it

Even protocol designers make this mistake

- TLS Renegotiation flaw (RFC 5746)
- Renegotiation was designed to upgrade an anonymous client connection
- Data from the anonymous part of the session was reused in forming the authenticated part of the session
- Anonymous sessions weren’t so anonymous after all, since the authenticated session depended on the anonymous session’s identity

NTLM too…

- Designed to be a replacement for plaintext passwords
  - Challenge/response
- Authenticates client to server
- Authenticates server to client only after client has provided its credentials
- No resistance to man-in-the-middle attacks
- Therefore: vulnerable to authentication forwarding

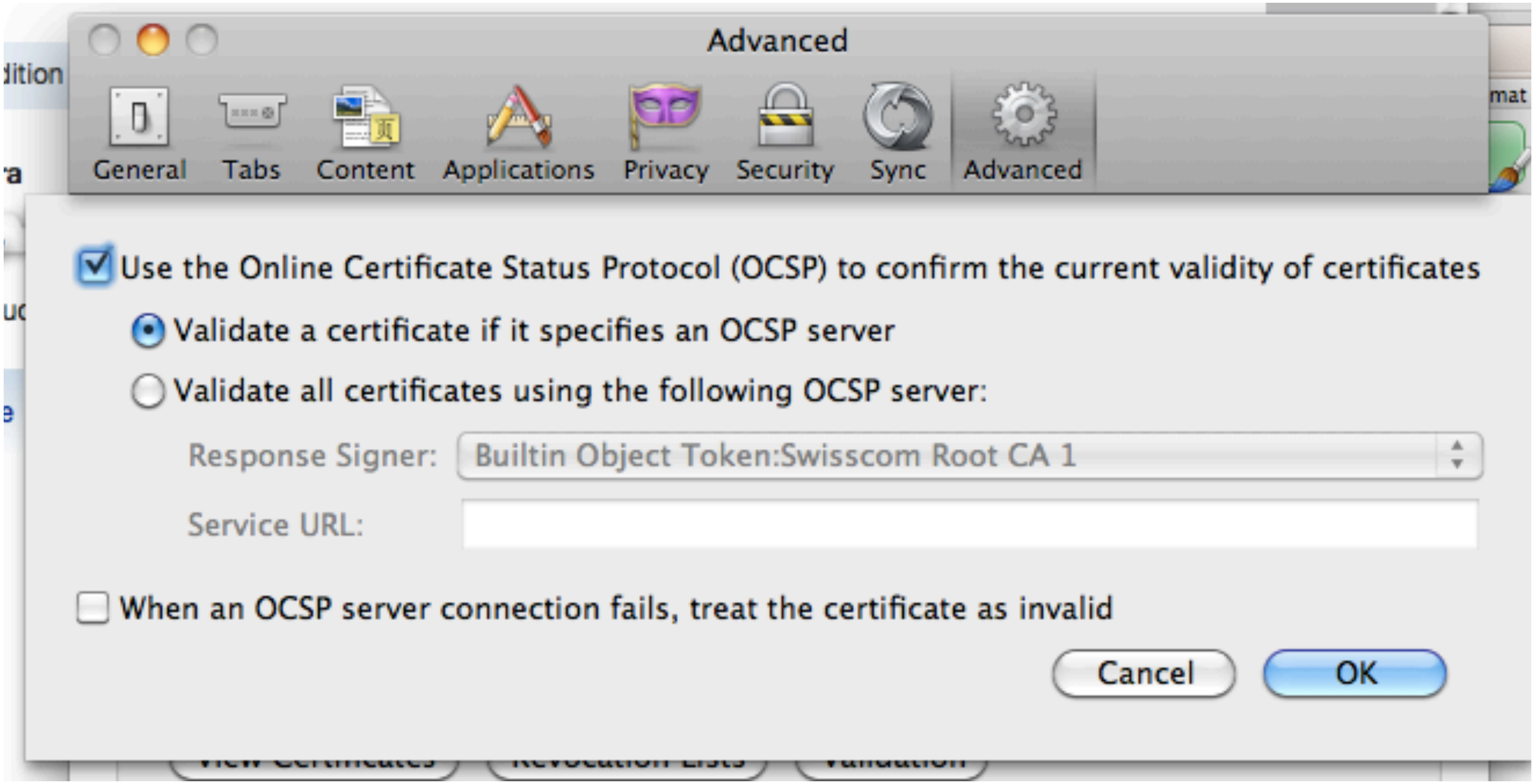
PKI is a vulnerable system

- Commodo issued signed certs inappropriately
  - Reseller able to sign certificates with only a username and password [until today?]
  - Username and password hardcoded into a DLL
  - Skr1pt k1dd33 gets hold of the DLL
  - Hilarity ensues

- No revocation
  - Dan Geer e-mail from keynote
  - Major browsers (all of them) fail open by default or always
- [screen shot of ff and/or error; maybe of mac keychain]
-

- ~1500 trusted CA certs (as of 3/11)

Firefox’s default



☑ Use the Online Certificate Status Protocol (OCSP) to confirm the current validity of certificates

● Validate a certificate if it specifies an OCSP server

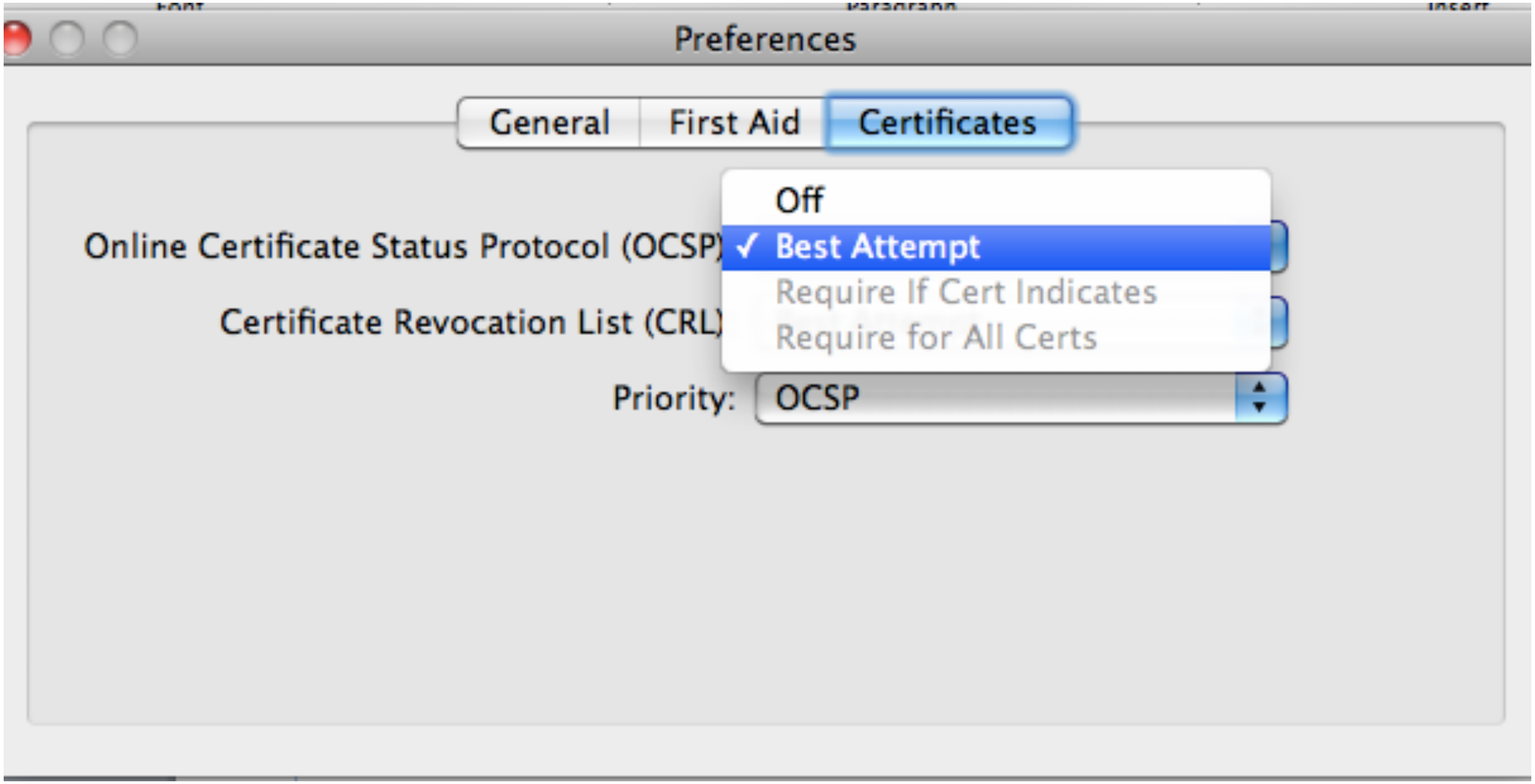
○ Validate all certificates using the following OCSP server:

Response Signer: Builtin Object Token:Swisscom Root CA 1

Service URL:

□ When an OCSP server connection fails, treat the certificate as invalid

Mac Snow Leopard default



Online Certificate Status Protocol (OCSP): Best Attempt

Certificate Revocation List (CRL)

Priority: OCSP

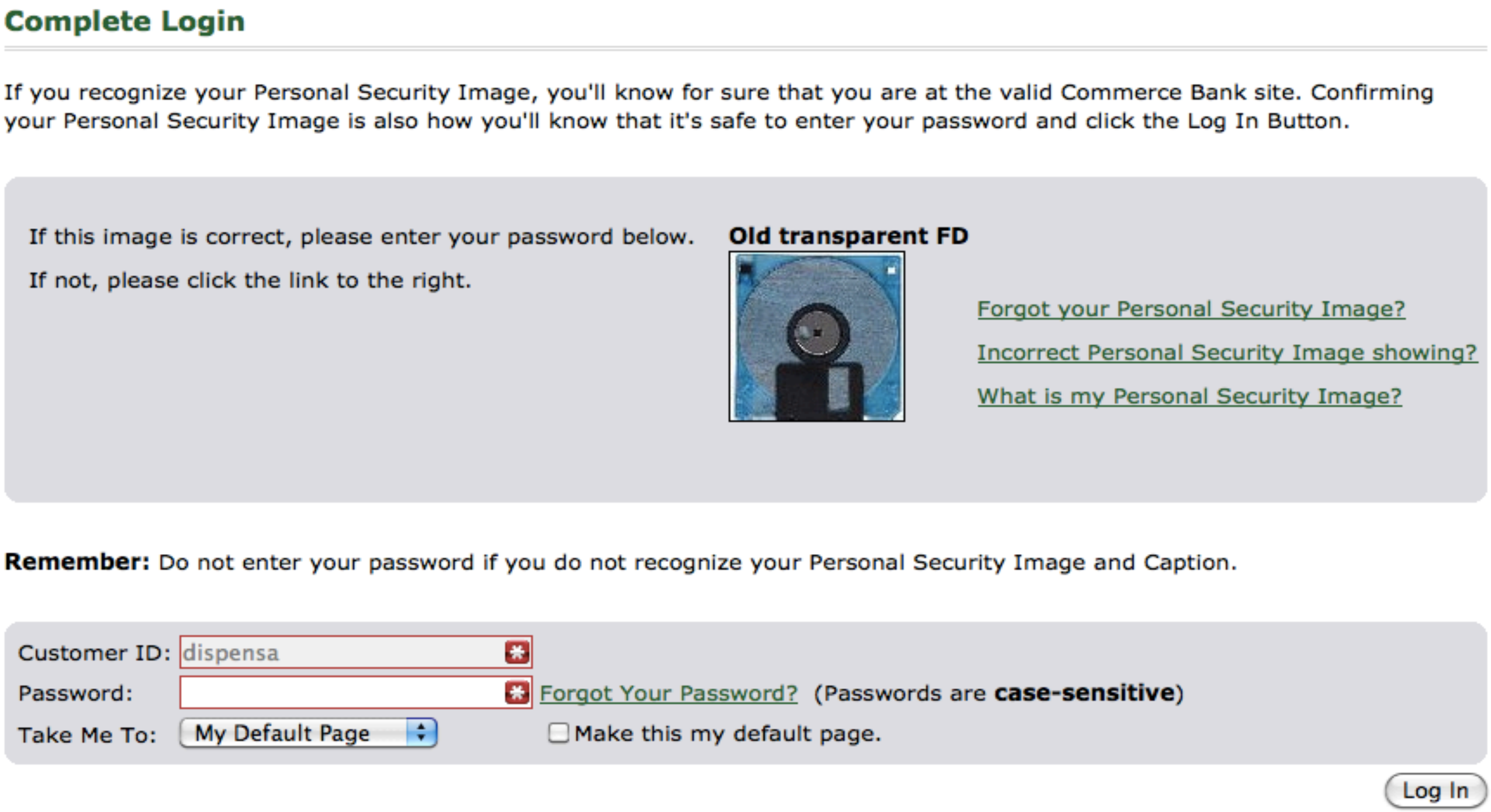
EV-SSL

- EV-SSL is an attempt at improving the system
- Some might say that EV-SSL is an attempt to do what everyone thought CA’s were doing all along
- Like all authentication technologies:
  - Useless if the bad guy gets to choose the level of security
  - Useless until users refuse to use blue-bar websites

Client trust

- SSL sites generally handle anonymous clients
- That means that only the client can ensure that there is no man-in-the-middle
- Therefore, the SSL server trusts the client's list of CA's completely
  - What about MitM corporate proxy servers?
- Ask yourself: do you, as a server, have an interest in the security of the session?
  - This may be an unwise delegation of responsibility

PassMark



Complete Login

If you recognize your Personal Security Image, you'll know for sure that you are at the valid Commerce Bank site. Confirming your Personal Security Image is also how you'll know that it's safe to enter your password and click the Log In Button.

If this image is correct, please enter your password below.

If not, please click the link to the right.

Old transparent FD

Forgot your Personal Security Image?

Incorrect Personal Security Image showing?

What is my Personal Security Image?

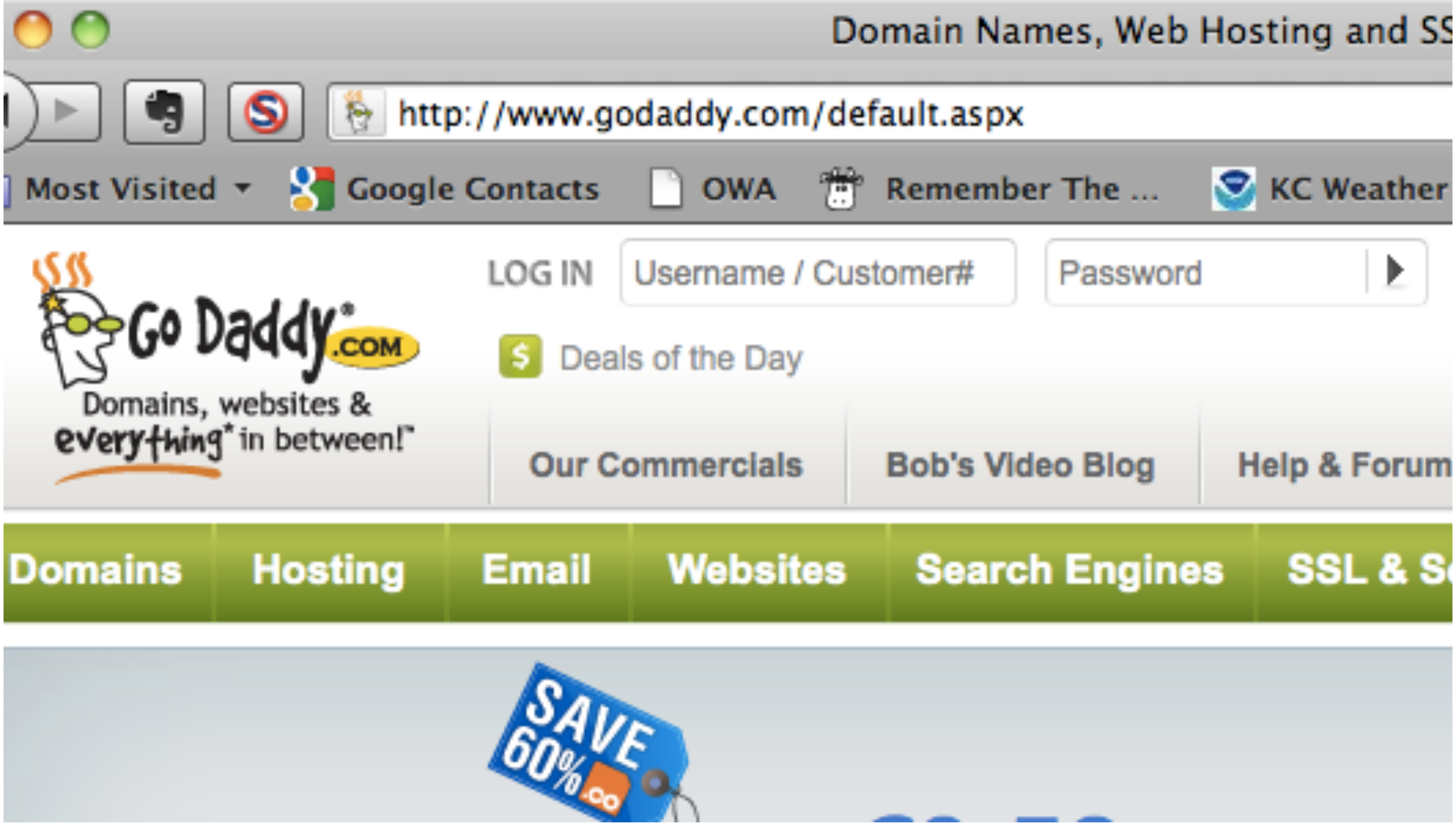
Remember: Do not enter your password if you do not recognize your Personal Security Image and Caption.

Customer ID:	dispensa	
Password:		Forgot Your Password? (Passwords are **case-sensitive**)
Take Me To:	My Default Page	□ Make this my default page.

Opt-in security

- 97% of study participants failed to notice the absence of the SiteKey image; only 2/3 of those didn’t log in
- “The Emperor’s New Security Indicators” - Harvard/MIT study, 2007
- Moral: opt-in security is worse than no security
  - A false sense of security
- Promiscuous authentication, again

But wait, there’s more!



Domain Names, Web Hosting and SS

http://www.godaddy.com/default.aspx

Login forms delivered over HTTP

- Anyone on the path between the server and the client can re-point the form action

- It’s trivial to own someone on public wi-fi (“coffee shop” attack)

- Advice: never submit a web form (with anything serious) if it arrived unencrypted.

- Whether or not the form action is HTTPS is irrelevant.

And then there’s this:

```
$ curl https://google.com
<HTML><HEAD><meta http-equiv="content-
type" content="text/html;charset=utf-8">
<TITLE>302 Moved</TITLE></HEAD><BODY>
<H1>302 Moved</H1>
The document has moved
<A HREF="http://www.google.com">here</A>.
</BODY></HTML>
```

The obligatory RSA slide

- Anyone can be the victim of a sophisticated attack – not a story.
- Two interesting stories here:
  - Ownable architecture – no obvious way to recover
  - Be careful when you have secrets
- Disclosure story
  - One wonders what RSA’s big customers are hearing
  - I know what the small customers are hearing
  - Can’t help but assume it’s pretty bad
- OATH tokens - unaffected

Multifactor authentication

- Two-factor is generally:
  - Something you know
  - Something you have

- It may also be:
  - Something you are

- Or, otherwise put:
  - Something they steal
  - Something they beat out of you
  - Something they chop off

Out-of-band authentication

- Leverages another network (usually the phone network) and another end device (e.g., a phone)
- Makes the attacker’s job more difficult: requires pwning two different endpoints on two different networks

Stored credentials

- They're everywhere
  - web services APIs
  - Phones
  - every web browser

- Notion of refusing to store credentials is disappearing
  - Twitter
  - Facebook
  - Lots of other dynamic websites

- How do you apply two-factor to phone-based e-mail?

Sessions

- History: Serial lines
- Then: a cached credential, or even just a TCP SEQ#
- Now: some crypto state
- TLS sessions: strongly authenticated, but not used

The real problem: login sessions

- Login sessions are blank check to do whatever the authenticated user could do, regardless of what he wants to do

For example:

- Steve wants to do *something*
- *someone* wants to transfer $100K

Abusing login sessions

- Firesheep by Eric Butler
- Basic idea: steal unencrypted session cookies
- http://codebutler.com/firesheep

More abusing of sessions

- [Phone browsers]

Solution: event authentication

- What exactly is being authenticated to whom
- Tightly bound together:
  - Steve wants to transfer $100K
- Current use cases: Banking and financial services
- Coming soon: enterprise

Event authentication mechanisms

- Smart cards with readers
  - EMV
  - Chip & Pin
- Phone verification
  - Phone call
  - Text message
- Phone apps
- Others
  - One-time card numbers, etc.

Authentication 2.0

Login sessions are dead. Now what?

- Multi-factor
  - Passwords suck
- Out-of-band
  - Guard against owned endpoints
- Event-based, not session-based
  - Control session risk

Questions?

Steve Dispensa
dispensa@phonefactor.com
@dispensa / kernelmustard.com

Marsh Ray
marsh@extendedsubset.com
@marshray / extendedsubset.com