AppSec Reanimated

Webinar Series

This series explores modern appsec approaches. What tools and techniques improve security within the SDLC? How do we effectively combine manual analysis with automation? How do we adopt what works and avoid what doesn’t?

A Promethean Struggle: PCI's Lessons for Passwords

Mike Shema
mike@cobalt.io
March 29, 2017

“Plus, you'll be one of over 12 million customers who have safely shopped with us without credit card fraud. If you feel more comfortable, you can enter only your card's last 5 digits and its expiration date. Once you have fully submitted your order, you can call in the rest of your card number.”

– Amazon.com. October 13, 1999
(via archive.org)

“There's a possibility of electrocution! Do you understand?”

– Gene Wilder, *Young Frankenstein*

Premise: Controls & Iteration

Minimize an app's risk of cardholder data loss, impose consequences for noncompliance, improve and clarify over time.

https://www.pcisecuritystandards.org

Goals

Build and Maintain a Secure Network

Protect Cardholder Data

Maintain a Vulnerability Management Program

Implement Strong Access Control Measures

Regularly Monitor and Test Networks

Maintain an Information Security Policy

Have to Transport Secrets

HTTPS should be commonplace.

Its use remains haphazard.

PCI requirements have likely motivated faster adoption of good TLS for in-scope apps.

Delete, Delegate, Decrease

No card data? No PCI.

Use a third-party payment processor — tokenization instead of encryption.

Double-iframe payment flows — reduce scope.

16 Digits,
8 Characters,
A Few Lessons

Seems Familiar

Present to appropriate party only when needed.

Either rotate on every transaction, or maintain until suspicion of fraud/compromise.

Cost burden to replace out of cycle — loss of subscription revenue; loss of users.

Seems Worse

Card loss can drain an account, lead to unauthorized purchases. But loss is tied to that card like a 1-to-1 mapping and may be recoverable.

Password loss has escalating impacts, from one app, to many apps, to any app via password reset email.

Characters of Failure

Composition rules — they reduce entropy, will be cracked anyway.

Hints are horrible — even worse entropy.

Frequent rotation — not risk-based, reinforces poor decisions.

“Why Johnny Can’t Encrypt”
— Alma Whitten and J.D. Tygar, 1999. [link]

“The…process should be designed and implemented so it is easy for users to do the right thing, hard to do the wrong thing, and easy to recover when the wrong thing happens.”
— NIST SP 800-63A (DRAFT).

“Here’s Johnny!”

— Jack Nicholson, *The Shining*.

Concrete Steps

Tokenize identity; OAuth 2.0; SAML.

+ More multi-factor instead of just more hashing.

Recovery — e.g. Facebook's Delegated Recovery

Password managers enable better behavior.

Secure the browser before completing authentication

Cautious Steps

Delivery — notifications rather than email or SMS

Most identity remains email address and/or phone number

Delegation introduces additional attack models, different risks, and potential privacy problems.

More complex recovery and reset process with OAuth and service tokens.

Protect	Treat data (card, passwords) as a liability.
	Encrypt traffic, avoid storage, expose never.
	Identify data flows.
Control	Delegate where reasonable.
	Build processes to log, monitor, alert.
Isolate	Sandboxed transactions.
	Restricted to mutually-authenticated services.

Thank you!

mike@cobalt.io

Questions?

https://webinar.cobalt.io

Stay tuned for more **AppSec Reanimated.**

And check out the **AppSec Disrupted** series, too!