minimum viable SECURITY

jacob@jacobian.org
about:me

Core developer

django

Director of Security

heroku
Who is this talk for?
Table of Contents
1. Train your staff

2. Develop an SDL

3. Plan for incident response

4. Governance, Risk, and Compliance

5. Brag about it!
Day 1:

TRAIN your staff
Security is a shared responsibility
Basic security awareness training

- **Passwords**
  Buy your staff a password manager (LastPass, 1Password), and train them in its use and in good password hygiene

- **Multifactor auth**
  Require it everywhere possible, and train staff to use it.
  Recommendations: Authy, Duo Security, Yubikeys

- **Privacy**
  “Keep customer data in production”
Phishing
“Phishing [is] a favorite tactic of state-sponsored threat actors and criminal organizations… for two years running, more than two-thirds of incidents that comprise the Cyber-Espionage pattern have featured phishing.”

— 2015 Data Breach Investigations Report
“23% of recipients [open] phishing messages and 11% [click] on attachments.

“A campaign of just 10 e-mails yields a greater than 90% chance that at least one person will become the criminal’s prey.”

— 2015 Data Breach Investigations Report
How can we combat phishing?

- Better email filtering
  Google is *really* good at this — use them

- Be prepared to detect and respond
  Enable Vault; encourage staff to forward suspicious emails

- Investing in training
  phishme.com; you can run your own campaigns fairly easily
Writing secure code
Who’s responsible for writing secure code?
Who’s responsible for writing secure code?

Who’s responsible for writing tests?
“Push decisions around security as far down as possible.”

— Parisa Tabriz [paraphrased]
Good news: writing secure code is easy!
Writing secure code

• OWASP Top 10
  https://www.owasp.org/index.php/Top_10_2013-Top_10

• Mozilla’s Secure Coding Guidelines
  https://wiki.mozilla.org/WebAppSec/Secure_Coding_Guidelines

• Microsoft’s “Writing Secure Code”

• Apple’s Introduction to Secure Coding
BONUS POINTS
Your own secure coding guides

1. The Big Three: SQLi, XSS, and CSRF
   Your web framework should be handling these for you!

2. “Preventing the OWASP Top 10 at YourCo”

3. Authentication and session guidelines

4. Cryptography handbook

BONUS POINTS
Day 1: Training

- Basic security awareness
  
  Passwords, multi-factor auth, phishing

- Secure coding guides
  
  Several good public guides

- Bonus: write your own secure coding guides
  
  The Big Three: XSS, CSRF, SQLi
  
  “Preventing the OWASP Top 10 at YourCo”
Day 2: develop an SDL (secure development lifecycle)
Now we know how to write secure code; how do we ensure these practices get followed?
The cycle of knowledge, best practices, development, and retrospective.
Minimum viable SDL

1. **When** do we do security?
2. **Who** needs to do security?
3. What is “**doing security**”? 
When do we do security?

- Throughout development, as much as possible
  Set up an internal security mailing list and/or chat channel, even if you don’t have dedicated security staff.

- Before development starts (security architecture), and before you ship (security review)
  This is much harder without dedicated security staff.

- Agile makes this much more complicated...
Who needs to do security?

- Everyone!
  
  “Push decisions around security as far down as possible”

- Dedicated security staff should serve as consultants
  Consult on architecture; answer questions; provide expert review in high-risk situations

- Decide how “far up” to push security based on risk
Initial Security Assessment

This questionnaire, to be completed when you kick off a new project, will determine the risk profile of the component based on your answers. This will help us determine just how closely Security needs to be involved in the project. Once you've answered these few questions, this app will give you the risk assessment and the next steps you'll need to take. If anything looks wonky, or if the outcome isn't what you'd expected, hit up security@heroku.com.

Do you want direct security involvement throughout the project?

If you think the project is risky and want help from Security from the beginning: -- trust your gut. We'll get involved and help however we can.

[Radio buttons] Yes, I would like Security involved, No, I don't think I need direct involvement
What is “doing security”?
“[A] critical care specialist at Johns Hopkins Hospital named Peter Pronovost decided to give a doctor checklist a try. ... He designed it to tackle just one of their hundreds of potential tasks: ... central line infections.

“For a year afterward, Pronovost and his colleagues monitored what happened. The results were so dramatic that they weren’t sure whether to believe them: the ten-day line-infection rate went from 11 percent to zero.”
“[There are] three different kinds of problems in the world: the simple, the complicated, and the complex. Simple problems… are ones like baking a cake from a mix…. Complicated problems are ones like sending a rocket to the moon. They can sometimes be broken down into a series of simple problems. But there is no straightforward recipe…. Complex problems are ones like raising a child….

“We are besieged by simple problems.”
Kickoff

- Add team information and technical contact information to this card
- Add links to spec, documentation, kickoff deck etc to this card
- Add links to the repository where development is taking place
- If your project received a HIGH risk rating, add an item...

Basics

- Has the paranoid flag set (for apps that deal with sensitive data)
- Use an automated code quality tool, if available
- Use approved authentication methods, if applicable
- Component inventory entry is up-to-date, including any potential risk indicators
- If you interact with AWS directly, use a proper AWS account and permissions. If you need deletion, consult your local AWS admin
- Add an item...

Investigation

- Perform vulnerability analysis
- Assign and notify team(s)
- Create due date(s) based on severity
- Add remediation steps (with links to tracking cards, as appropriate) to the remediation list below
- Move this card to "remediating"
- Add an item...

Remediation

- Add an item...

Cryptographic Usage

- Follows cryptography best practices. PCI/DSS
- Only serves web traffic over https

Post-remediation

- Verify fix is live
- Post public disclosure (if necessary)
- Inform external reporter(s) of resolution (if applicable)
- Schedule retrospective
### A CHECKLIST FOR CHECKLISTS

<table>
<thead>
<tr>
<th>Development</th>
<th>Drafting</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do you have clear, concise objectives for your checklist?</td>
<td>Does the Checklist:</td>
<td>Have you:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>□ A critical safety step and in great danger of being missed?</td>
<td>□ Utilize natural breaks in workflow (pause points)?</td>
<td>□ Trialed the checklist with front line users (either in a real or simulated situation)?</td>
</tr>
<tr>
<td>□ Not adequately checked by other mechanisms?</td>
<td>□ Use simple sentence structure and basic language?</td>
<td>□ Modified the checklist in response to repeated trials?</td>
</tr>
<tr>
<td>□ Actionable, with a specific response required for each item?</td>
<td>□ Have a title that reflects its objectives?</td>
<td></td>
</tr>
<tr>
<td>□ Designed to be read aloud as a verbal check?</td>
<td>□ Have a simple, uncluttered, and logical format?</td>
<td></td>
</tr>
<tr>
<td>□ One that can be affected by the use of a checklist?</td>
<td>□ Fit on one page?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Minimize the use of color?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Is the font:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Sans serif?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Upper and lower case text?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Large enough to be read easily?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Dark on a light background?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Are there fewer than 10 items per pause point?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>□ Is the date of creation (or revision) clearly marked?</td>
<td></td>
</tr>
<tr>
<td>Have you considered:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>□ Adding items that will improve communication among team members?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>□ Involving all members of the team in the checklist creation process?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Day 2: SDL

- Create a virtuous cycle
- Minimum Viable SDL:
  When do we do security?
  Who needs to do security?
  What is “doing security”?
- Checklists are an amazingly powerful tool
Day 3:

plan for

INCIDENT RESPONSE
“But I would not feel so all alone
Everybody must get owned”

— Bob Dylan*

*more or less
Data breaches are fast becoming just a "fact of life." Companies, thus, are judged as much for their response as for the breach itself.

You can’t effectively plan your emergency response during the emergency!
Minimum viable IR plan

1. Initiate
   Where should a (potential) incident be reported?
   How will incidents be tracked?
   What are the roles and responsibilities during a incident?

2. Communicate
   Where will comms happen? Who will be involved?
   Who will send situation updates? To whom? How often?
Minimum viable IR plan

3. Assess
   Where do we collect information? Who follows up?
   How do we determine severity?

4. Remediate
   Given that severity, what is our response SLA?
   How do we prioritize remediation (short vs long-term)?
   How do we communicate and track remediation?
   When and how do we notify customers?
5. Retrospective

How do we explore causes? Five Whys? Infinite Hows?
How will we piece together a timeline for the retro?
What metrics do we need to collect?
How will we track long-term follow-up?
More reading

• Incident Response at Heroku

• The Incident Command System
  https://en.wikipedia.org/wiki/Incident_Command_System

• Security Breach 101
  https://medium.com/@magoo/security-breach-101-b0f7897c027c
BONUS POINTS
Run a tabletop exercise
Day 3: Incident Response

• Bad things **do** happen to good people
  Better to be prepared than to be in denial

• Create an incident response plan
  Initiate, Communicate, Assess, Remediate, Retrospective

• Bonus: run a tabletop exercise
Day 4: Governance, Risk, & Compliance
Formal risk programs

- Aren’t worth the investment at small scale
  Startups are more concerned about traction than compliance — and this is fine.

- However, at scale they become increasingly vital

- Don’t shoot yourself in the foot
  You can save a world of effort in the future by laying the groundwork now.
GRC 101

• Document everything
  Make a decision about security, privacy, or company policy? Write it down.
  Don’t worry about formal language: it’s not necessary.

• Track as much as you can
  Good: “Hi Mary, this email is to confirm that I’ve given you commit access github.com/myorg/myrepo”
  Better: build systems to track security work, access, etc.
Snap #6121 audit hipchat about 6 hours ago

Hipchat Admin Page: [Link]

Contents
diffed last unique snap (#6102)

- Added
- Removed
Your most important documents

• Data classification guide and policy
  What data you have, where it’s stored, who has access to it

• Access control checklists
  Access control is tremendously important, and you’ll spend ages cleaning it up later if you don’t get it right now.
  Bonus: you can use these for on- and off-boarding.

• Exception process
  There’s an exception to every rule (except this one). So document how they work, and document when you grant them.
BONUS POINTS
Formal risk programs

• Safe Harbor
  Is fairly easy, simple (mostly privacy-related), and lets you sell services to Europe.

• PCI-DSS 3.0 SAQ-A-EP
  Fairly complex (~100 controls), but well-written, actionable, and a good starting point. Many companies use PCI as a proxy for “has their shit together”

• Consensus Assessments Initiative Questionnaire
  https://cloudsecurityalliance.org/group/consensus-assessments/
  Consolidates most-commonly-asked questions into a single questionnaire, focused on *aaS. Comprehensive (~300 controls), maps to PCI, HIPAA, FedRAMP, etc.
Governance, Risk, and Compliance

• Document everything

• Write a few basic policies:
  Data classification, access control, exception process

• Bonuses: Safe Harbor, PCI, CAIQ
Day 5: BRAG about it
Congratulations:
you’re now better off than 90%* of your peers!

* completely made-up statistic
Tell the world

• Privacy policy
  yoursire.com/privacy

• Security information page
  yoursire.com/security

• Security FAQ/knowledgebase
  May be public or private, depending (more later)
Privacy policy

- Boilerplate, but necessary — don’t skip it
  Many companies won’t do business with you without one.

- If you have lawyers, ask them to write it.

- If not, there are several OK templates to start from:
  http://automattic.com/privacy/ (CC-By-SA)
  https://wordpress.org/about/privacy/ (CC-By-SA)
Security information page

• Summarize your security practices and policies
  This can (and should) be less formal; it’s where you can tell your security “story”

• If you have any formal attestations, list them here.

• Explain how to report vulnerabilities
  security@yoursite.com should be a thing
  Having a PGP key helps pass the “brown M&M test”
Security FAQ/KB

• Every time a customer asks you a question about security, write down the answer.

Over time, natural groupings will occur. Or, you could use the PCI or CAIQ topics as your groupings.

• Should you publish this publicly?

Transparency is good, but there are also good reasons to limit this to “available on request” or “only with an NDA”

My litmus test: does publishing it make customers safer? Then it should be public. If not, keep it private.
Day 5: Brag about it

- yoursit.com/privacy
  Boilerplate, but necessary

- yoursit.com/security
  Tell your story, and how to report vulnerabilities

- Security FAQ
  DRY for customer customers
1. Train your staff

2. Develop an SDL

3. Plan for incident response

4. Governance, Risk, and Compliance

5. Brag about it!
Congratulations!
minimum viable SECURITY

jacob@jacobian.org