App Privacy

Your customers’ privacy IS job 1

Sam I Am Not

- James O’Keefe
- Captain, Massachusetts Pirate Party
- … won’t make you walk the plank
- Testing computer software since 1990
- Teaching computer privacy and security since 2012

Why you should care about your customers privacy

- The right thing to do
- If you don’t, a competitor will
- If your competitor doesn’t, differentiate what you offer by caring about privacy
- … because a leak could tank your startup

Threat Modeling

- What data do you have to protect?
- From whom do you have to protect it?
- What harm would result if your data leaked or went away?
- What steps will you take to protect it and recover faster if your site is compromised?

Secure yourself

- Apply security patches to your laptop/development machine
- Update the software you use with security patches
- Get a virus checker and keep it up to date
- Back up early and often

Secure your hosting and domain Services

- Be sure to secure your master hosting provider and/or domain name service credentials with a long random password that you store in an encrypted password manager and back it up securely
- Those services require email addresses so get one that isn’t your personal email address
- Turn on Multi Factor Authentication (2FA/MFA) and use an app like Google Authenticator or Authy instead of Txts
- Can a tech at your hosting service confirm your password? Get a new hosting service

Beware of being Phished

- No, not the band, phishing attacks
- Be careful of all emails from your hosting or domain name provider
- If they ask you to login by clicking on a link in the email, don’t
- Check the link by hovering over it with your mouse pointer
- Always login in a separate window using the url you stored in your password manager

Secure your F&^%ing servers

- Use SSH to login to your servers via a terminal, ideally with a private/public key
- Change the server’s root or other login passwords to something long and random. Store in an encrypted password manager and back it up securely
- Do you control your own servers? Plan to patch the OS regularly and frequently

Secure your F&^%ing database

- Control your own database servers? Plan to patch frequently
- Don't make it easy for criminals, change the default DB and admin credentials
- Passwords: Salt liberally
- User password recovery questions: Never use plain text
- Other data: encrypt as much as you can

Secure your F&^%ing application software

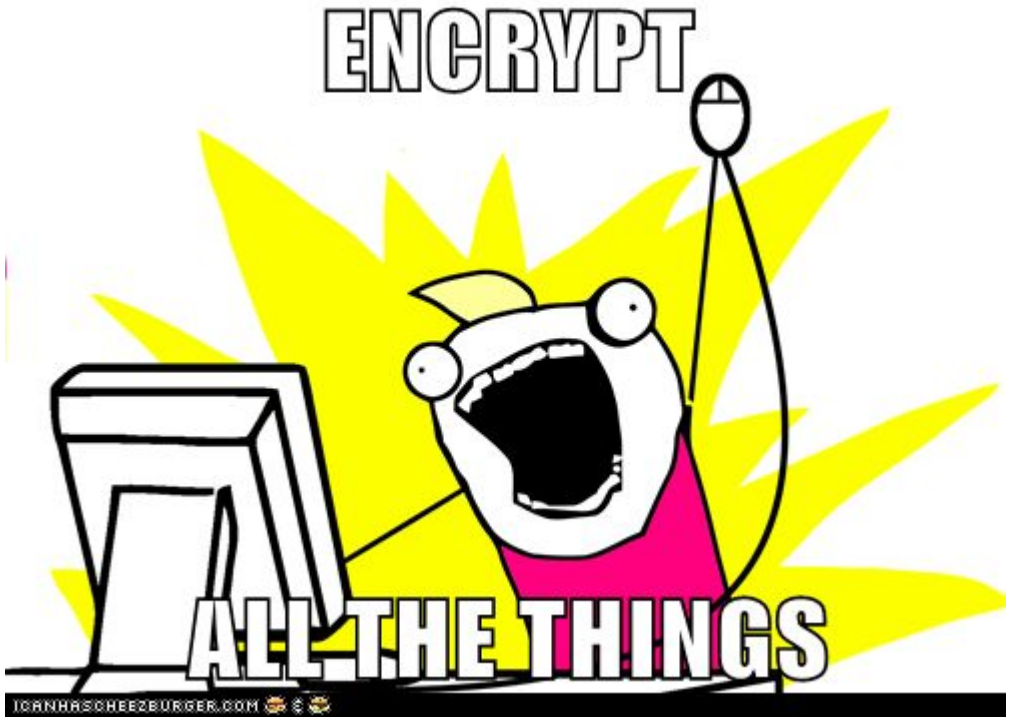
- Plan to patch your application software regularly and frequently
- Setup an RSS watch for updates on the application software you use
- … even for random third party library you get from Github
- Setup a scheduled search of the CVE and NIST Vulnerability Databases for the application software you use

... AND KNOW YOUR DEPENDENCIES

- Check your software's dependencies with one of the tools mentioned in techbeacon.com/app-dev-testing/13-tools-checking-security-risk-open-source-dependencies
- Do you really need some third party library that does one little thing or can you write it yourself so you have fewer dependencies?

Protect your connections from prying eyes

- Https protects you and your customers
- You can pay for an SSL certificate or use free ones from Let’s Encrypt (letsencrypt.org)
- Not just for communicating with customers, use between your servers



You have keys, so keep them secure

- You have login credentials
- You have database credentials
- You have private keys
- You need to keep them secure and backed up
- … on your local computer
- … ideally in an encrypted password manager
- … but not in your source repository
- … especially a public one like Github

Back it up

- Back up your servers, database and code regularly
- Store your backups in a separate location
- Encrypt your backups
- Test your disaster recovery plan
- … Having reliable (and offsite) backups helps foil ran$omware attacks
- … and when terrorists take down your datacenter

Customer logins

- Always provide multiple third party sign on options like OAuth, Google, Facebook or Twitter
- Never store a customer’s password in plain text
- … Always encrypt and salt passwords
- Do support Multi Factor Authentication (MFA/2FA) like Authy and Google Authenticator. 2FA via Txt message isn’t that secure since your phone number can be spoofed
- Keep an eye on WebAuthn

Don't encourage bad password practices

- Short passwords are trivial to crack even with upper and lowercase letters, numbers and special characters
- Longer passwords, especially when random, are best
- … but humans won't remember those without a password manager, which not enough people have
- So don't require upper and lowercase letters, numbers and special characters and just give them positive feedback for longer passwords
- Never require them to update their password as they'll just choose <previous password>1

Authorization

- The customer can log on
- … now you have to make sure they only see their data or data for others they are allowed to see
- Use an automated test suite to test your authorization model to be sure data access is limited
- Don’t be AT&T, obfuscate your data ids:
  - Don’t use a numeric identifier to identify customers or their data since since if your authorization model fails and you will leaks data
  - Use unique random identifiers so someone cannot iterate over the data

Do you really need to collect that data?

- Really?
- Really? Really?!
- Really? Really? Really?!!!

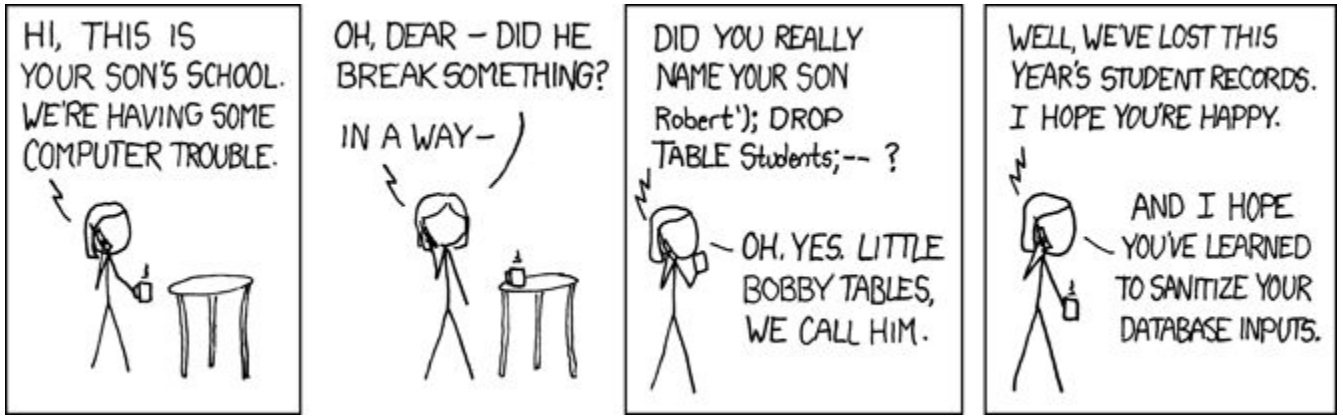
- Seriously, the less data you have, the less vulnerable you and your customers are if your database is compromised
- If you don’t need it, don’t collect it

Secure the data you do collect

- You know the minimum set of data you need to collect
- Analyze your dataset for what is the worst that could happen if it was published
- Anything private to your customers (name, address, social security number [do you really need that?], etc.) has to be secured and encrypted.
- Think hard about how you can secure personally identifiable information that isn’t private
- If nothing bad will happen if a piece of data gets out, then you probably don’t need to worry about it

Designing secure code

- Never forget Little Bobby Tables (SQL Injection attacks)



- Don't leave yourself open to XSS attacks either
- Keep up with the other OWASP Top 10 entries
- … and be sure to keep up with software updates

All along the watch tower

- Put some fake data, including email addresses, in your database that only you know about
- Register the fake emails with https://haveibeenpwned.com/
- ... so you get (sorta) advanced warning if your data leaked

You got hacked? Come clean early

- Honesty is the best policy
- Tell them what you:
  - did to minimize the breach
  - are going to do to solve this problem now
  - are going to do to make sure it doesn’t happen again
  - are going to do to compensate your customers for their hassle (free 3 months added to membership?)
- Don't be a meme: Avoid the Streisand effect

Resources

OAuth 2.0: oauth.net/2/

OWASP Resources:

- owasp.org/index.php/SQL_Injection
- owasp.org/index.php/Cross-site_Scripting_(XSS)
- owasp.org/index.php/Category:OWASP_Top_Ten_Project

How to reach me

- Email: jokeefe@jamesokeefe.org
- My public key: https://www.jamesokeefe.org/public-key/