Crypto(graphy) in the Dev Stack

Be A More Secure Coder

Resume.md

Director of Engineering at the Electronic Frontier Foundation.

10 Years in technology

- 💻Web Applications
- ☁️ Cloud & Network Security
- Certifications: *Security+*, *GFACT*, *GCLD*, *GPCS*, *GCSA*

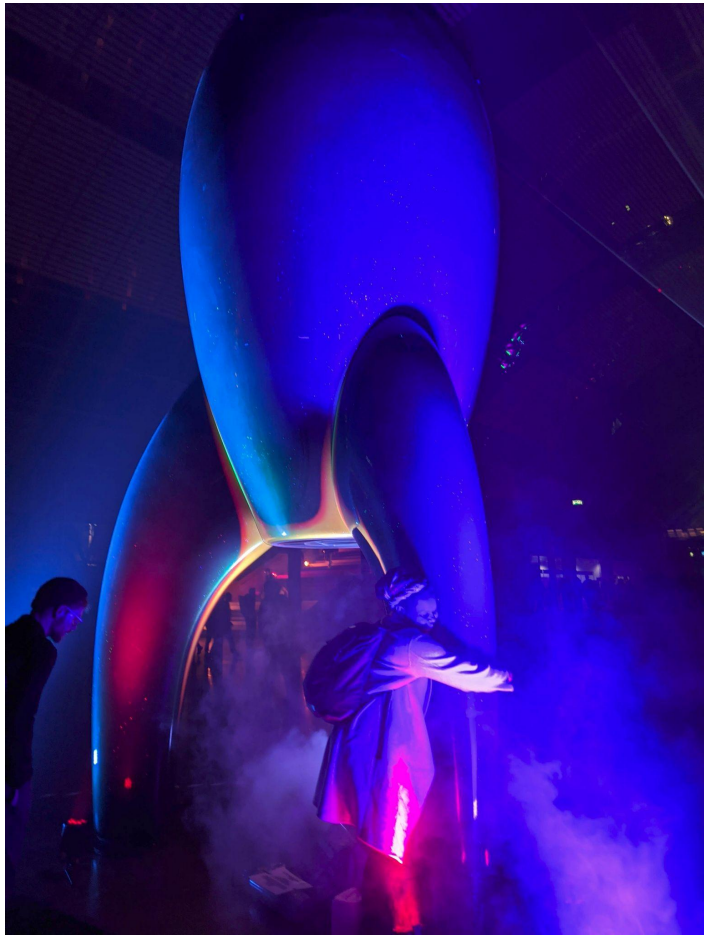
Cloud certs

Things I do at work

- 🔑Encrypt the Web
- Research how web PKI is impacted by government policy
- Research privacy implications of digital identity frameworks

Resu(me).md

- Mama/Wife/Aunt/Sister/God Mama
- Video Games + Crochet
- Home networking & hardware hobbyist
  - 💻PC builds
  - 📱Smartphones from around the world
  - Raspberry Pi
  - Arduino, etc.
- Generally breaking things and sometimes making things work



Being a Safer Developer

- **Authentication of developers**
- Integrity of code
- Secrets Management & Dealing with Github as a Public Record

Authentication Protocols

- LDAP
- Kerberos
- RADIUS
- Oauth2
- Open ID | OpenID Connect (OIDC)
- SAML | SAML2.0
- WebAuthn API / FIDO2

Password + ? = MFA

(Multi - factor Authentication) * Not Authorization

? = Something you have

? = Something you know

? = Something you are

Different ways to Authenticate

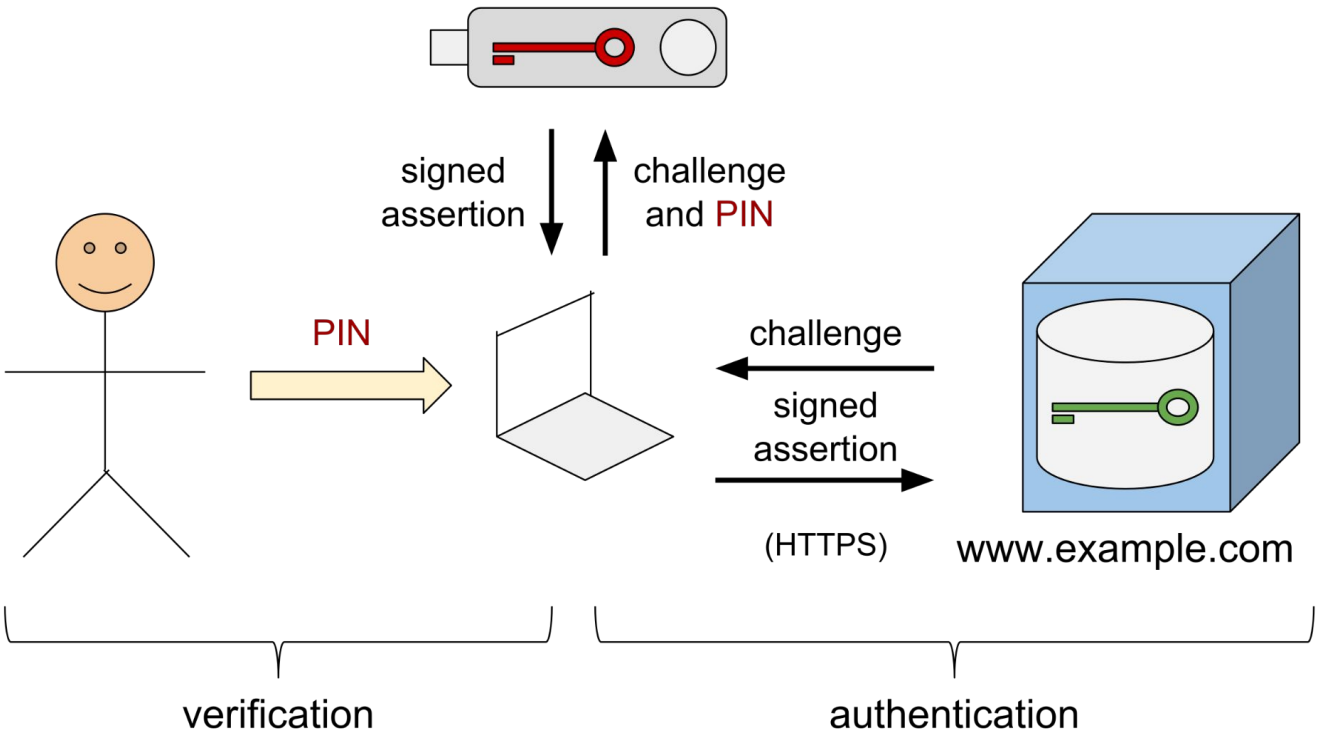
- **Security keys**
- One Time Token
- Biometrics
- Pin code
- ...SMS :(
  - Vulnerable SS7
  - Sim Jacking

Why Smartcards?

- Easily replaceable*
- Private keys never leave device
- Low resource usage, lasts a long time

DEMO: 2FA with Yubikey

WebAuthn API + FIDO2



“What about the CLI though?”

Creating a personal access token

You should create a personal access token to use in place of a password with the command line or with the API.

Note: If you use GitHub CLI to authenticate to GitHub on the command line, you can skip generating a personal access token and authenticate via the web browser instead. For more information about authenticating with GitHub CLI, see `gh auth login`.

Personal access tokens (PATs) are an alternative to using passwords for authentication to GitHub when using the GitHub API or the command line.

Being a Safer Developer

- Authentication of developers
- **Integrity of code**
- Secrets Management & Dealing with Github as a Public Record

Code Integrity in Github

- Signed commits
  - Why?
    - Ties single cryptographic identity (author) with a code commit
    - Ties specific code changes to the author
    - Significantly eliminates impersonation
  - How
    - Generating a GPG Key and linking to your Github profile
      - GPG Keys can be used to sign messages
      - Sign emails
      - Encrypt files
      - Management Tips: https://www.digitalocean.com/community/tutorials/how-to-use-gpg-to-encrypt-and-sign-messages

Public Key Systems

- Asymmetric cryptography
  - Public key
    - Shared with other systems
  - Private Key
    - Guard with your life
- Different systems of management and distri
- PKI and web security are the biggest examples of this
  - Certificate Authorities

Public Key Systems

Not all keys are made equal

- RSA and ECC
  - RSA has been standard for a while, but with vulnerabilities (3K-4K generated keys are generally safe)
    - Prime factorization principles
  - ECC is the recommended algo going forward due to lighter and more secure
    - Mathematical representation of elliptic curves

DEMO: GPG and Github

Generate GPG Key

alexis@alexis-ThinkPad-X1-Carbon:~

alexis@alexis-ThinkPad-X1-Carbon:~ | alexis@alexis-ThinkPad-X1-Carbon:~

```
~/Documents/techint > gpg --full-generate-key                    at 10:45:20 AM
```

Export Public Key

alexis@alexis-ThinkPad-X1-Carbon:~

alexis@alexis-ThinkPad-X1-Carbon:~ alexis@alexis-ThinkPad-X1-Carbon:~

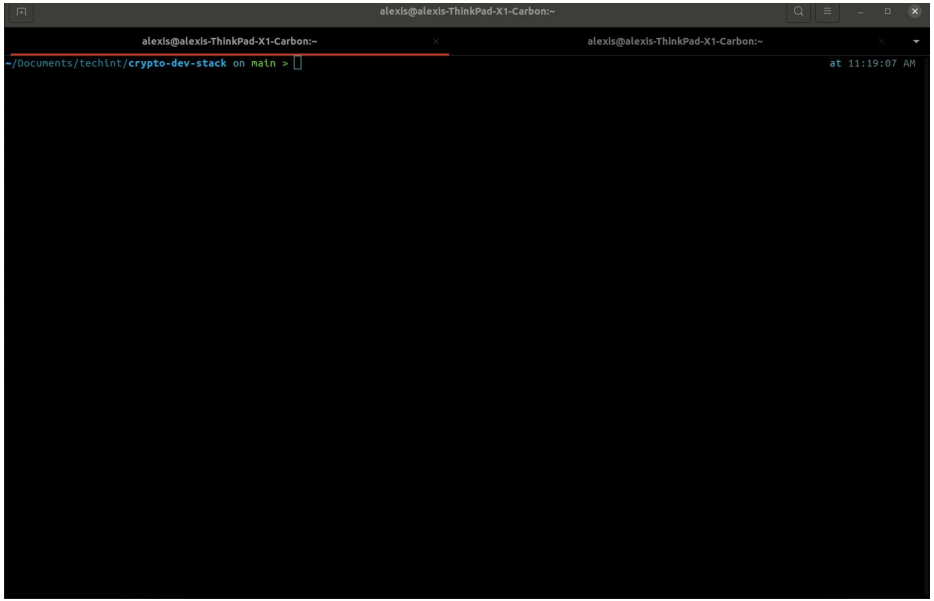
~/Documents/techint > at 10:49:39 AM

DEMO Bloopers

Commits on Mar 25, 2022

Add new GIF for GPG tutorial zoracon committed 7 minutes ago	Unverified	969b381
Add GIFS zoracon committed 14 minutes ago	Unverified	e58a465
Initial commit zoracon committed 1 hour ago	Verified	4c7fbdd

???



Wowwwwwww

EFF DEMO Bloopers

main

Commits on Mar 25, 2022

Edit README zoracon committed 29 seconds ago	Verified	d2aa1b8
Edit README zoracon committed 6 minutes ago		1630
Edit README zoracon committed 8 minutes ago		b82b
Edit README zoracon committed 10 minutes ago		5b79
Add new GIF for GPG tutorial zoracon committed 19 minutes ago	Unverified	969b381
Add GIFS zoracon committed 26 minutes ago	Unverified	e58a465
Initial commit zoracon committed 1 hour ago	Verified	4c7fbdd

This commit was signed with the committer's verified signature.

zoracon
Alexis

GPG key ID: EF9F92E86D8CF643
Learn about vigilant mode.

Where to store private key?

- Airgapped computer
- HSM (CAs)
- Offline (encrypted storage)
- Commercial key vaults

Quick Overview Hashing and Salting

Hashing

- One way cryptographic function
- SHA-256 (SHA-1 and MD5 have collision attacks)

Salt

- Adds random data to ensure unique hash

Integrity that file or data was not changed, and is unique

Being a Safer Developer

- Authentication of developers
- Integrity of code
- **Secrets Management & Dealing with Github as a Public Record**

Keeping Secrets

- API Keys
- Tokens
- Passwords
- Network information (ips and internal hosts)
- Yes, even in private repo, secrets management goes a long way

Git Secrets Scanning

alexis@alexis-ThinkPad-X1-Carbon:~

alexis@alexis-ThinkPad-X1-Carbon:~ alexis@alexis-ThinkPad-X1-Carbon:~

~/Documents/techint/crypto-dev-stack on main > at 03:11:26 P

Git Secrets in Github Actions

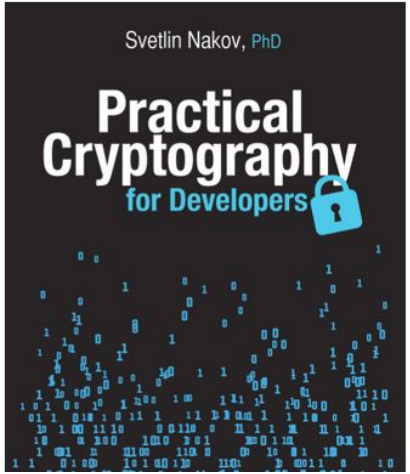
- **GITHUB_TOKEN** is an auto-generated personal access token, encrypted
- Scope is limited, note for open source projects
- You can use repo generated secrets in Github
- Not all Github Actions are to be automatically trusted.

https://cryptobook.nakov.com/

P Practical Cryptography for ...

Welcome

> **Warning**: this book is **not finished**! I am still working on some of the chapters. O completed, I will publish it as PDF and EPUB. Be patient.



https://cryptopals.com/

the cryptopals crypto challenges

- Set 1: Basics
- Set 2: Block crypto
- Set 3: Block & stream crypto
- Set 4: Stream crypto and randomness
- Set 5: Diffie-Hellman and friends
- Set 6: RSA and DSA
- Set 7: Hashes
- Set 8: Abstract Algebra

Welcome to the challenges

Work in progress.

This site will host all eight sets of our crypto challenges, with solutions in most mainstream languages.

But: it doesn't yet. If we waited to hit "publish" until everything was here, we might be writing this in 2015. So we're publishing as we go. In particular: give us a little time on the challenge solutions.

We can't introduce these any better than Maciej Ceglowski did, so read that blog post first.

We've built a collection of 48 exercises that demonstrate attacks on real-world crypto.

This is a different way to learn about crypto than taking a class or reading a book. We give you problems to solve. They're derived from weaknesses in real-world systems and modern cryptographic constructions. We give you enough info to learn about the underlying crypto concepts yourself. When you're finished, you'll not only have learned a good deal about how cryptosystems are built, but you'll also understand how they're attacked.

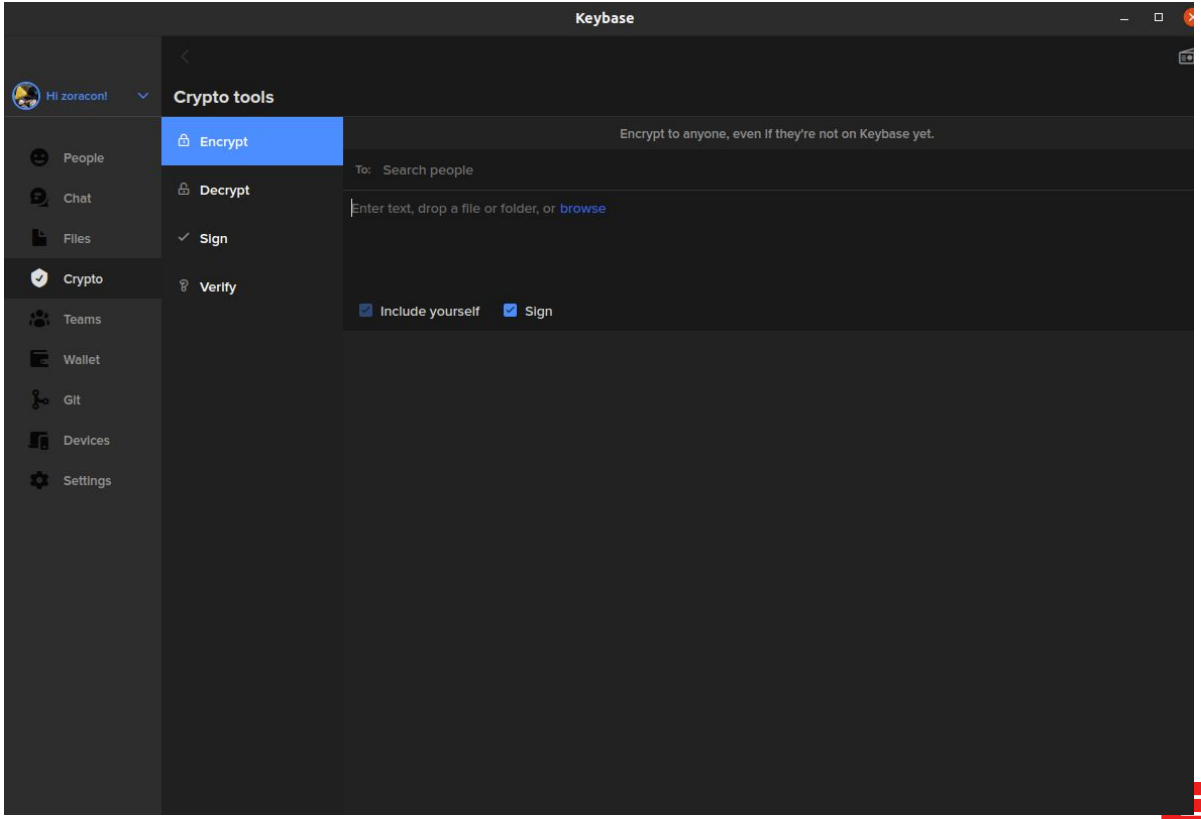
What Are The Rules?

There aren't any! For several years, we ran these challenges over email, and asked participants not to share their results. *The honor system worked beautifully!* But now we're ready to set aside the ceremony and just publish the challenges for everyone to work on.

How Much Math Do I Need To Know?

If you have any trouble with the math in these problems, you should be able to find a local 9th grader to help you out. It turns out that many modern crypto attacks don't involve much hard math.

Favorite Crypto in Action



https://keybase.io/

Thank you!

alexis@eff.org

https://keybase.io/zoracon