GOTO
Guide

LET US HELP YOU

Ask questions through **the app**

★★★★★

also remember to rate session

Download on the App Store

GET IT ON Google Play

THANK YOU!

#GOTOams

Security Styles

GOTO Amsterdam 2022

Eleanor Saitta / Systems Structure Ltd.

Two styles, both alike in dignity
In fair Internet, where we lay our scene

How do you stop phishing?

Type one: Run phishing tests, with a leaderboard so everyone knows who gets caught most often. Require mandatory training if you click on a link.

Type two: Roll out U2F tokens, eliminate native office clients, and build out-of-band confirmation loops for business processes. Mention phishing briefly in the annual training.

How do you work with other teams?

Type one: Use policy to build hard checkpoints into everyone's processes. Block functionality at deploy and allow developers to discover what won't work in prod.

Type two: Give devs candy and make friends with them. Sit in on team meetings so you hear about plans early. Document everything.

How do you defend a service?

Type one: Put a WAF in front of it, or better yet a firewall appliance, and install an endpoint security tool on the VM.

Type two: Have a human and some robots test it. Fix stuff. Instrument what it does so you can see if it acts weird. Make it immutable and ephemeral. Eliminate egress. Add rate limiting.

How do you handle compliance?

Type one: Do what the standard says. Hire someone to do the paperwork for the auditor. Use policy to force teams to do their part.

Type two: Figure out what's technically sensible for the spirit of the regs and automate that. Integrate the automation into team workflows. Sell it to your auditor and work to help them understand it.

How do you fix vulnerabilities?

Type one, small team: Write some code to fix each instance, one at a time.

Type one, big team: Write some code that creates a statistical mitigation that reduces the likelihood of the attack succeeding.

Type two: Fix the underlying structural problem at the framework level, with testing to confirm you've covered all paths.

How do you handle mistakes?

Type one: If a breach is (can be) attributed to human error, fire the person responsible, as per policy.

Type two: Have the person help you understand the drivers that led them to make the choices they did, and fix those structures/systems. Celebrate their contribution.

How do you make decisions?

Type one: Look at industry best practices and compliance requirements. Buy what you have budget for, and put anything you can't buy a solution for in the risk register and call it mitigated.

Type two: Reason about your technical, business, and human systems, from first principles. Design a solution that meets both business and security needs. Implement it collaboratively.

Quick tips for starting from zero

Ditch all your Windows boxes — use Macs/Chromebooks and ditch Office for Workspace

Yubikeys for everyone, everything tied to SSO

Get some Thinkst Canaries in prod/if you have an office

Backup everything and make sure it's tested

Track where your data is and be careful where it goes

Treat code as an expense, not an asset

Include maintenance when costing new SaaS tools

What is a System?

Systems exist to do things in the world

To be useful, they need to have certain emergent properties

Whole-system properties which occur in a specific context

Require unified effort to deliver

Properties you care about:

- Correctness
- Performance
- Efficiency
- Reliability
- Observability
- Security
- Resilience

What is Security?

A secure system is one that:

- Enables a chosen set of people to predictably accomplish specific goals
- Does so in the face of actions by a chosen set of adversaries
- Predictably prevents that chosen set of adversaries from

What is Security?

Alternately:

- Reliability and correctness of outcomes in the presence of an adversary
- Close-loop defense of outcomes

What is Resilience?

The ability of a system to deal with unforeseen modes of failure without complete failure

Resilience is a property of humans, not code

Designing for Resilient Security

Designing both processes and technical systems in accordance with specific principles leads to desired emergent properties

Properties of technical artifacts vs. properties of human processes

You are responsible for the impact of your work on people's lives.

Personas to Examine

A domestic violence victim seeking an abortion

A queer teen

A union organizer

Component Principles

A selection of interesting system design principles:

- Statelessness/Logiclessness
- Immutability and Ephemerality
- Canonical Stores
- Unlinkability

State and Logic

Services should either do computation or hold state, not both

Complex components are unpredictable

Immutability and Ephemerality

Data, configuration, and memory are all state

Immutable systems eliminate unnecessary state

Respinning a cluster resets state

Minimal, Canonical State

Every piece of state should exist canonically in exactly one place

As few systems as possible should be stores of state

Any duplicated state must be validated

Unlinkability

Privacy and anonymity are ill-defined

X piece of data is unlinkable to Y piece of data under these assumptions

Process Principles

And a few for the human side of the org:

- Declare and Generate
- Design for Failure
- Decide at the Edge
- Slack

Declare, don't Program

Declarative configurations are easier for both humans and computers to create, compose, and validate

Use parser generators, strongly typed languages, and state machine generators

Design for Failure

Failure and compromise are inevitable

Design components and systems to handle both predictable and unpredictable types of failures

Think about security controls as a whole, assuming that some layers will always fail

Build the system you'd like to have during a compromise or outage

Decentralize Decisionmaking

Empower teams and engineers to work autonomously, so decisions can happen where people have full context

Focus on coordination and communication over control

Ensure teams have thick horizontal relationships outside of formal processes

Slack

Resilience requires teams to have downtime

Improving any emergent property takes more time than the bare minimum

Apply hard caps to feature velocity, ensure people take vacations, have large on-call rotations, and track out of hours work

Startup looking to get serious about security?

Let's talk.

ella@structures.systems

Eleanor Saitta
Systems Structure Ltd.

GOTO
Guide

Remember to **rate this session**

THANK YOU!

#GOTOams