Secure Software Design

Andey Robins (They/Them)

Spring 23 - Week 1

Cybersecurity: Secure Software Design

Course Aims

The aim of this course: provide students with an understanding of the software design lifecycle, cybersecurity, and most importantly, the overlap between the two. Students will get hands on experience designing, writing, and maintaining applications with security included as an integral component.

Outline

- Syllabus & Housekeeping
- Software Design History
- Security Design Motivation

Disclaimer

In this course, you will learn the techniques and processes employed by skillful attackers. We learn this for the explicit purpose of developing more secure systems and subverting potential attacks. We discuss the situations and boundaries of ethical security practice in this course and you are held to this high standard. **Do not attack any system or information without explicit prior written permission.** Not only is it a really bad idea, it's probably illegal and a violation of UWYO network rules (UW Regulation 8-1) and the CEPS technology policy. I am not responsible for any actions you perform.

Meeting Times

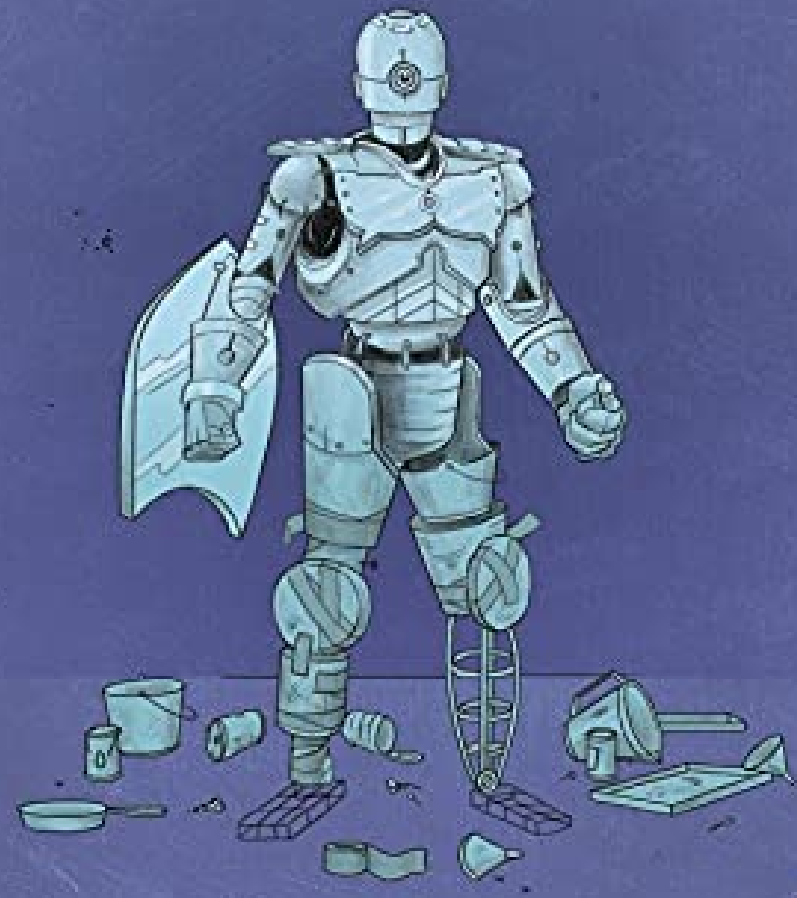
Class: MWF 8-8:50 am

Office Hours: M 11-12 & MR 1-2 & By Appointment

Office: EERB 228 (SSC Lab)

Designing Secure Software

A Guide for Developers



Loren Kohnfelder

no starch press

O'REILLY®

10th Anniversary Updated for Java 8

Head First Design Patterns

A Brain-Friendly Guide

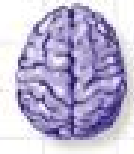
Avoid those embarrassing coupling mistakes

Discover the secrets of the Patterns Guru

Find out how Starbuzz Coffee doubled their stock price with the Decorator pattern

Learn why everything your friends know about Factory pattern is probably wrong

Load the patterns that matter straight into your brain



See why Jim's love life improved when he cut down his inheritance

Eric Freeman & Elisabeth Robson
with Kathy Sierra & Bert Bates

Assignments

- Written Homework
- Programming Homework
- Midterm
- Final Project

Structure

- Lecture presents concepts and gives examples
- Written homework assesses application of ideas
- Programming homework assesses ability to act on ideas

Grading

Category	4010 Grading	5010 Grading
Programming Homeworks	50 pts.	55 pts.
Written Homeworks	20 pts.	30 pts.
Midterm	10 pts.	15 pts.
Final Project	20 pts.	25 pts.
Total Points	100 pts.	125 pts.

Figure 3: Grade Breakdowns per Section

Late Work

- 75% credit up until the assignment is discussed in class

5010 Additional Topics

- Provable programs
- Zero-trust architecture
- Decentralized development
- Type theoretic security
- More/different as demanded (let me know if there are topics that sound interesting to you!)

Languages

- Javascript/Typescript
- Go
- Rust

Frameworks

- SvelteKit (TypeScript)
- Gin (Go)
- Rocket (Rust)

Questions?

The History of Software Engineering

What follows is a very abridged, loosly structured, highly opinionated view of the history of software engineering. I ask you to pay attention to trends and common themes more than any individual contribution. History is a series of movements, not individual, revolutionary steps. Here be dragons.

Outline

1. The Early Days (200 BCE - 1980s)
2. The Era of Personal Computing (1983 - 1990)
3. Web 1.0 (1991 - 1999)
4. The Dot Com Crash (2000 - 2005)
5. The Agile Age (2006 - 2009)
6. The Microservice Era (2010 - 2014)
7. The Emerging Serverless Era (2015 - Present)

The Early Days

The Antikythera Mechanism

- The oldest analogue computer
- Used to calculate astornomical positions and eclipses

Figure 4: The Antikythera Mechanism

Babbage's Analytical Engine

- Credited as the first computer
- Has an ALU, control flow, loops, and memory
- First turing complete computer

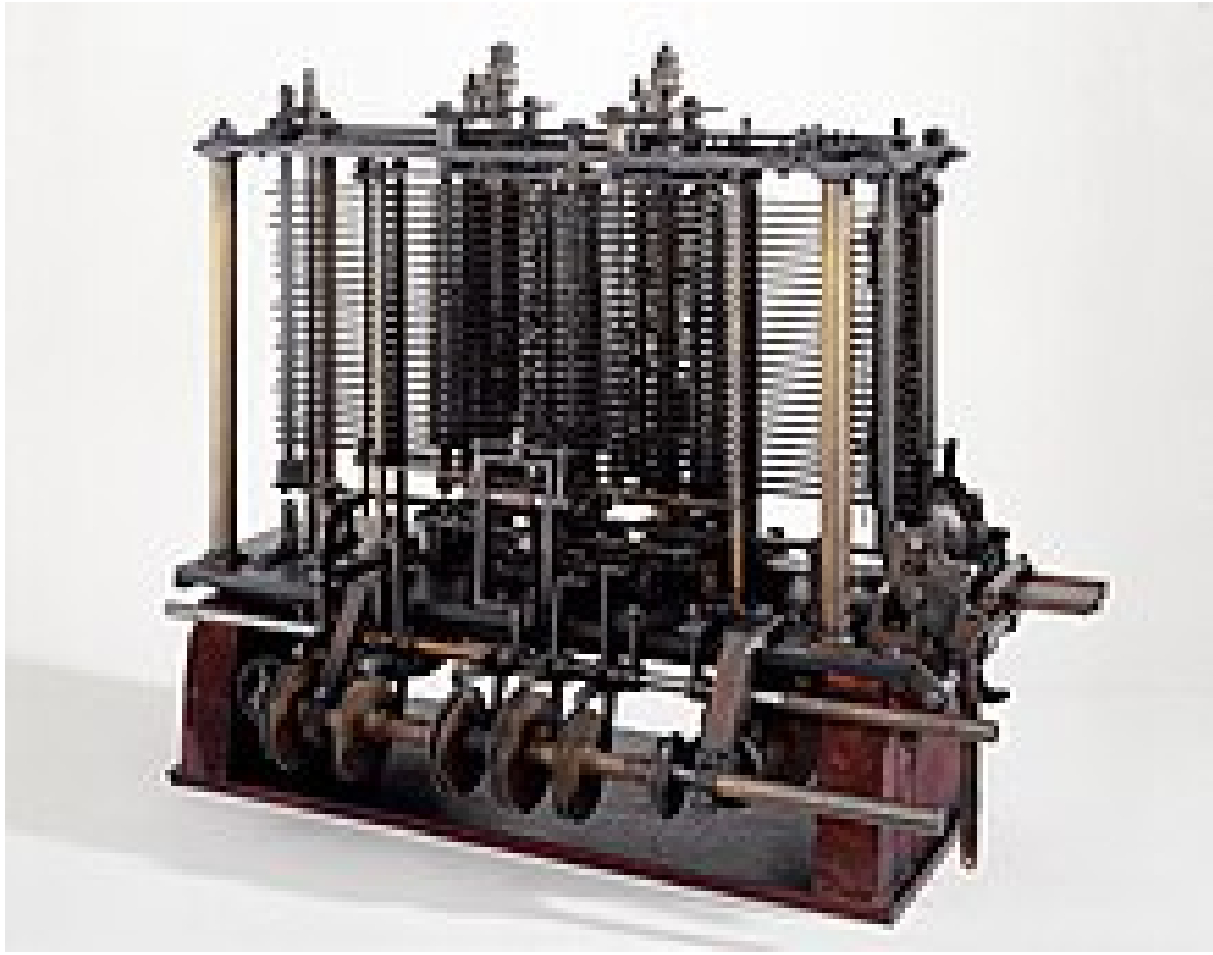


Figure 5: The Analytic Engine

Ada Lovelace

- Augusta King (1815 - 1852)
  - Countess of Lovelace, Augusta Ada King
  - AKA Ada Lovelace
- Published the first algorithm to be run by a computer
- Made to run on Babbage's Analytical Engine

"[The Analytical Engine] might act upon other things besides number, were objects found whose mutual fundamental relations could be expressed by those of the abstract science of operations, and which should be also susceptible of adaptations to the action of the operating notation and mechanism of the engine. . . Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent." - Ada of Lovelace

The First Computed Algorithm

- Tom Kilburn
- First implemented program calculates the greatest divisor of 2^18 (262,144)
- Took 58 minutes to run

```python
print("Calculating the greatest divisor of 2^18")
n = 2 ** 18

for i in range(2, int(n ** 0.5)+1):
  if n % i == 0:
    print(f'The greatest divisor is {int(n / i)}')
    print(f'{n} / {int(n/i)} = {i}')
    return
```

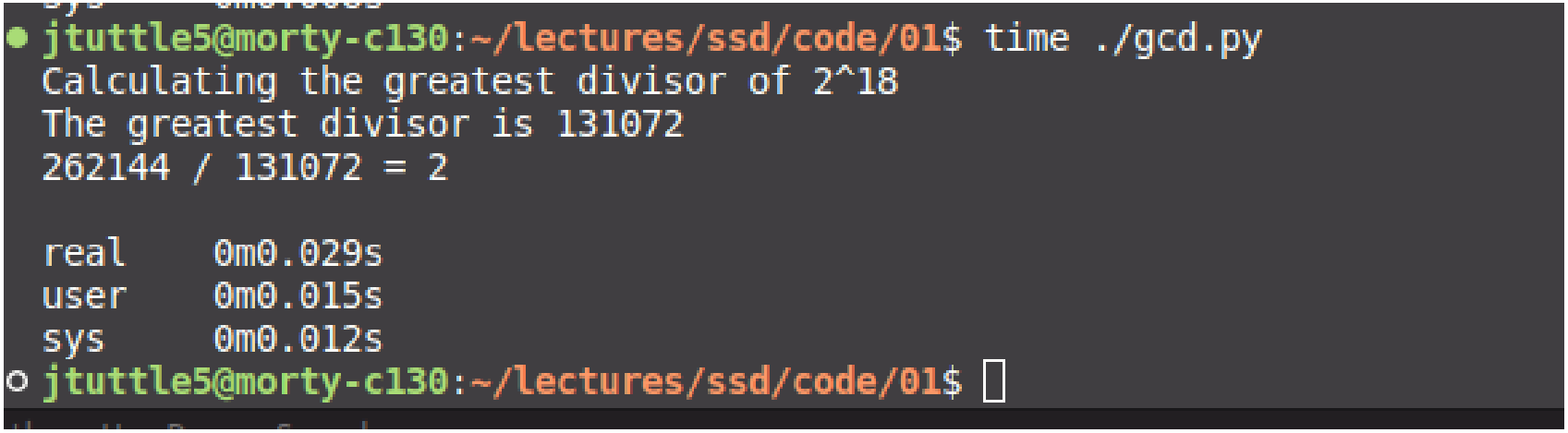


Figure 6: Timing of greatest divisor on a desktop computer

The Transistor

- Came to replace vacuum tubes
- Is the reason computers are reasonable sizes
- Led to the computer being a consumer product

First Compilers

- Plugging wires around and moving switches is hard, wouldn't it be nice if we had a program that could handle putting the program into the computer on its own?

COBOL

- There's a whole bunch of ways to program computers, wouldn't it be nice if there was some sort of coordinated logic for doing that?

Waterfall Design (1970)

- Formalizes the code writing process
- Creates a step by step process to create software
- One of the first industry standard processes

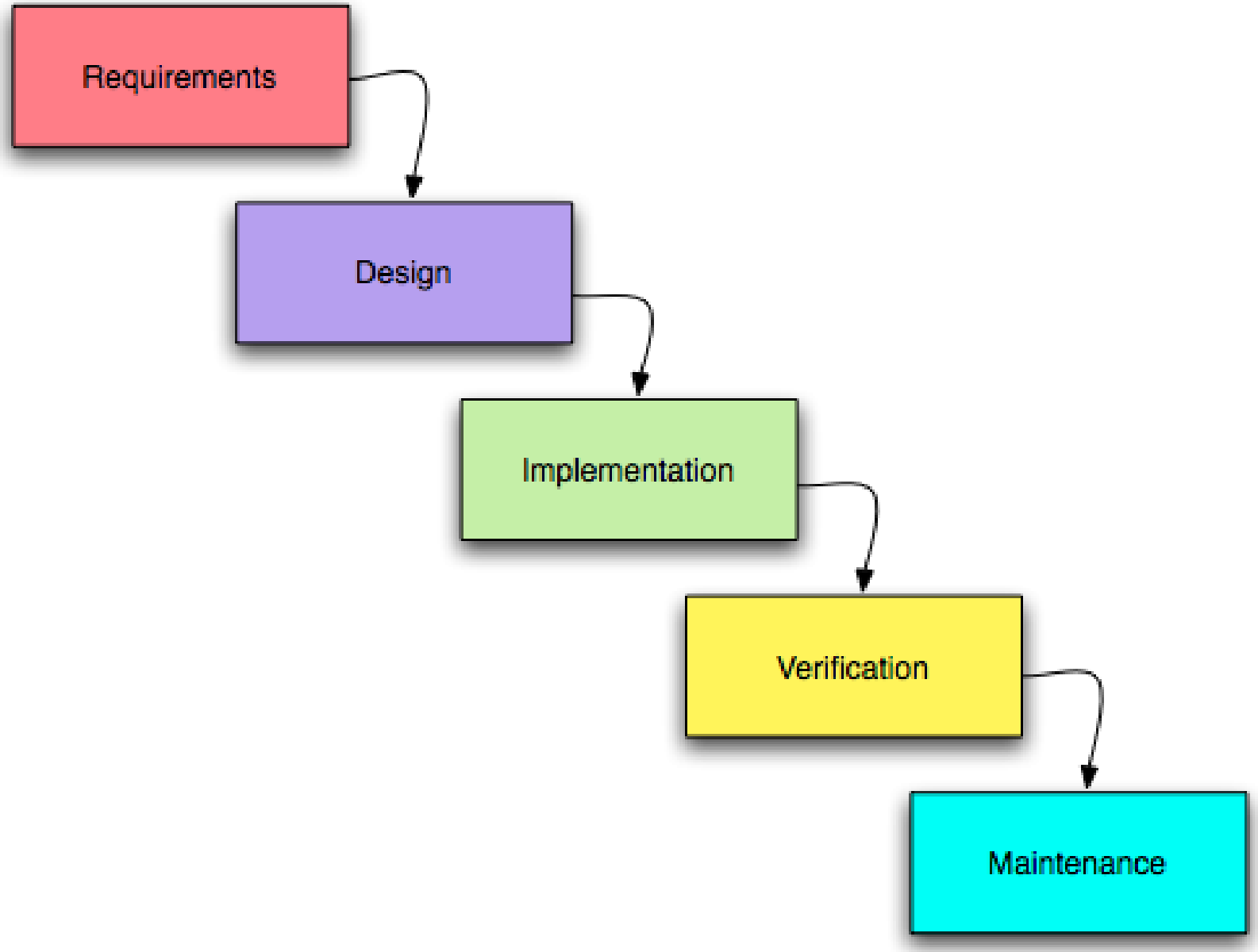


Figure 7: Waterfall design pattern

C (1972)

- Major standard
- Still around today
- Made interfacing with hardware more general and straightforward

Aside: C vs Rust

- "The last fifty years were written in C. The next fifty will be written in Rust"
- `NULL` was a "billion dollar mistake"
- NSA recommends businesses stop writing new software in C and instead use a memory safe language

The Apple II (1977)

- Began the wave of personal computing
- One of the first mass produced computers
- Affordable enough for home usage
- Led to the need for rigorous, deployable, software engineering

The Era of Personal Computing

DRM introduced (1983)

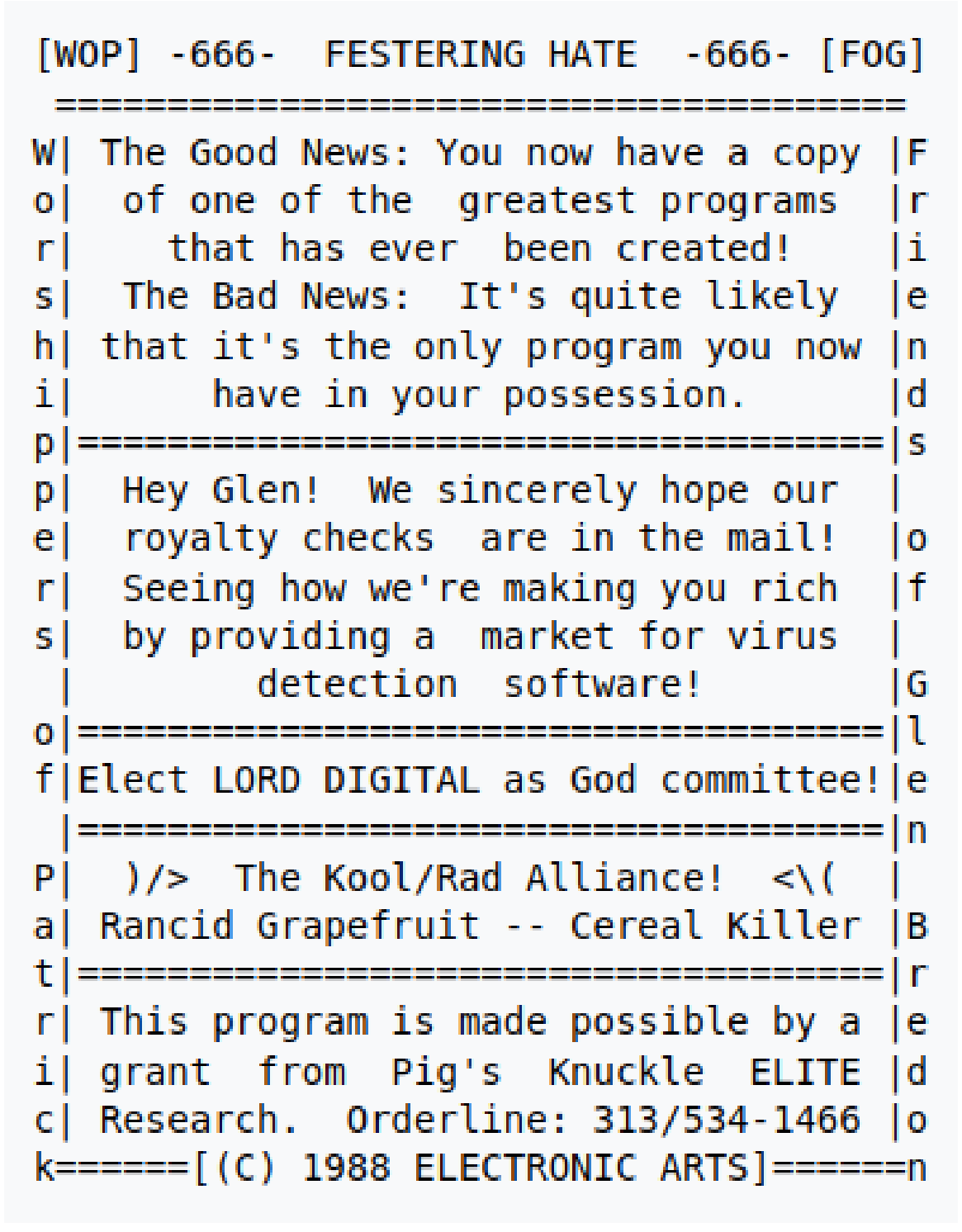
- Digital Rights Management
- The idea that intellectual property distributed digitally should be protected
- Gave rise to the cracking and piracy communities
- Valid and Questionable applications even to this day
  - Further complicated by the lack of “ownership” of digital goods

Word & Excel (mid 80s)

- Some of the earliest “killer apps” for computers
- Proved the usefulness of computers within business contexts
- Proprietary and licensed

Ping-Pong Virus & Cyberaids (1988)

- One of the first boot sector viruses
  - Caused a block to ping pong across the screen instead of booting
- CyberAIDS was one of the first major pieces of malware to escape into the world


[WOP] -666- FESTERING HATE -666- [FOG]
The Good News: You now have a copy of one of the greatest programs that has ever been created!
The Bad News: It's quite likely that it's the only program you now have in your possession.
Hey Glen! We sincerely hope our royalty checks are in the mail! Seeing how we're making you rich by providing a market for virus detection software!
Elect LORD DIGITAL as God committee!
)/> The Kool/Rad Alliance! <\(
Rancid Grapefruit -- Cereal Killer
This program is made possible by a grant from Pig's Knuckle ELITE Research. Orderline: 313/534-1466
[(C) 1988 ELECTRONIC ARTS]

Therac-25

- Likely the first time a human died because of code issues
- Radiation therapy machine
- Delivered lethal doses of radiation to patients due to race conditions in the code

Figure 9: Therac 25 Device

Web 1.0

The Internet (1991)

- much wow
- nifty! [citation needed]
- home of XKCD

Linux Kernel (1991)

- One of the first major OSS projects to gain traction
- Led to the major offshoots of operating systems we see today
- Established a lot of the precedent for OSS work

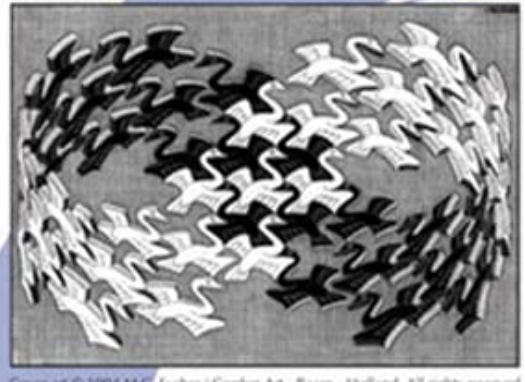
Design Patterns (GoF) (1994)

- An attempt to structure the practice of software engineering like other engineering disciplines
- Codified common practices of the day
- Seen by many as the foundation of modern software development

Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides



Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Citibank Hack (1995)

- Russian hackers extracted ~$10 million dollars
- First big-money/big-industry hit of the hacking community
- Proved the vulnerability of major systems to the public

Php (1995)

- Greatly simplified and abstracted the process of developing web applications
- Allowed for software development practices to be translated into the web
  - No longer requiring stringing together HTML and CSS and other small pieces

Java (1996)

- Why was java revolutionary?
- Runs on the JVM
- Widespread adoption of OO

Mobile Devices

- First PDAs emerge in 1996
- Similar to the Apple II, led to the need for consumer software
- Began a process which is slowly replacing traditional computers

The Dot Com Crash

SaaS (1999)

- Salesforce launches their customer relationship management platform
- First instance of Software as a Service
- Changed the profit model for software businesses to this day

ILOVEYOU (2000)

- Computer worm infection 10 million+ machines at the turn of the centur6y
- Overwrites, moves, and modifies random files before sending itself to as many people in the address book as possible
- One of the earliest email viruses

Agile (2001)

While the waterfall design methodology had its place, the needs of SaaS businesses require being able to adapt software to business needs faster than the methodology allows. Enter "Agile" methodologies (Scrum, Kanban, XP, etc.) which completely change the way software is designed, written, produced, and maintained.

Gary McKinnon and Military hacking (2001-2002)

- Gary McKinnon was a foreign civilian who broke into US military networks
- Ran through them for multiple years
- Only looking for proof of UFOs

AWS (2002)

- Web service provider
- Currently runs one third of the internet [Forbes]
- Expanding to provide any IaaS service you could need
- Lambda, EC2, S3, Dynamo, etc.

Making Money as a Tech Business

- Large amounts of capital combined with a new, emerging technology led to unwise investments
- Company must balloon quickly and gain a lions-share of users/marketshare
- Convert this dominance to sales
- Without a plan to become profitable, investors lose faith
- The money dries up
- The company is either Facebook or Myspace

Dot Com Bubble

- The start of the internet led to unknowns surrounding businesses operating in the space
- Low interest rates at the end of the '80s and early '90s meant lots of VC funding
- Any old idea could get pitched as investors wanted to get into the space

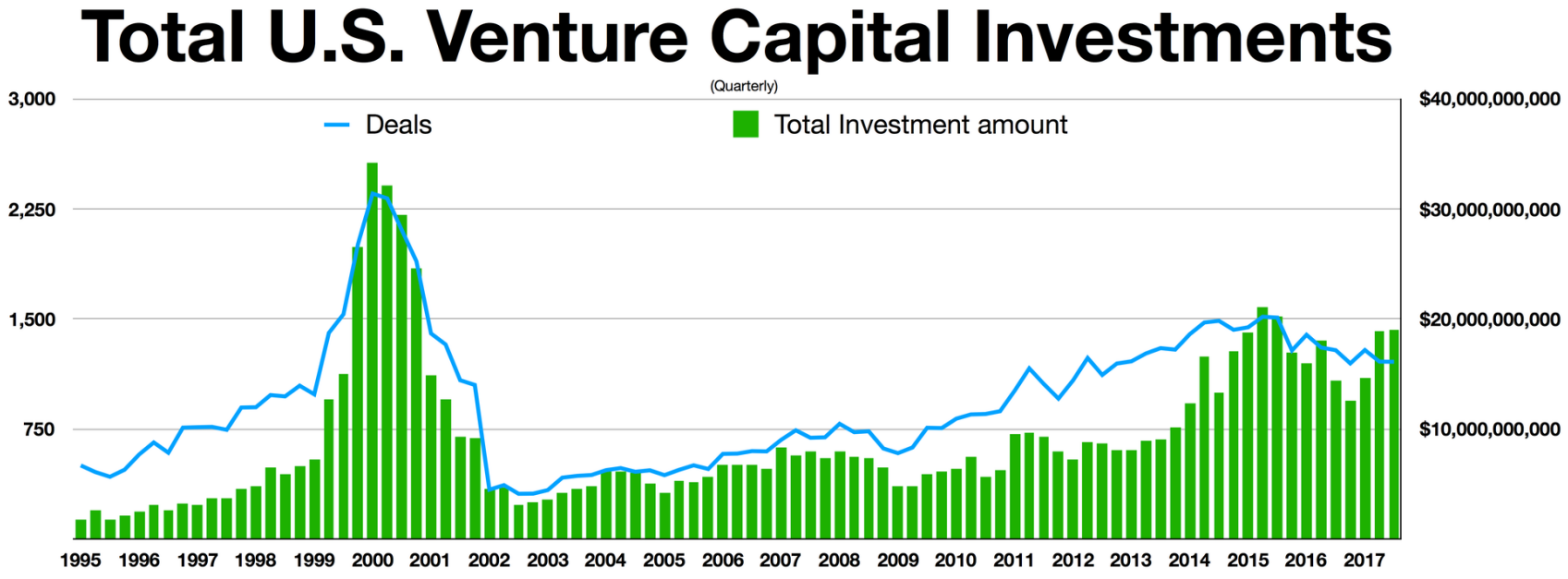


Figure 11: VC Funding in businesses around the dot-com bubble

Sandworm/Voodoo Bear is First Observed (2004)

- Later known for NotPetya
- Took down the Ukranian power grid in 2015
- Attacked Ukraine in 2017
- Responsible for the cyberattack on the 2018 Winter Olympic Games
- The earliest APT I could find

Ruby & Rails (2004)

- Built on the idea that software should be understood by humans first and computers second
- The first major “full-stack framework” which applied software engineering patterns and paradigms
- An evolution of PHP, further increasing accessibility for software development for the web

The Agile Age

jQuery (2006)

- It turns out, pure javascript doesn't integrate with the DOM very well
- Wouldn't it be nice if there was a tool which allows us to interact with the DOM?

First iPhone (2007)

- Did for mobile computing what the Apple][did for home computing
- Put a computer in everyone's pocket
- Increased "software engineering" to include mobile applications
- Drove the need for meshnets across the world

Anonymous is First Observed (2008)

- First association with hacktivism
  - Previously known basically for being a club on 4chan
- Began as attacks against the Church of Scientology

"Anonymous is tired of corporate interests controlling the internet and silencing the people's rights to spread information, but more importantly, the right to SHARE with one another. The RIAA and the MPAA feign to aid the artists and their cause; yet they do no such thing. In their eyes is not hope, only dollar signs. Anonymous will not stand this any longer." - Anonymous

Node (2009)

- Lots of people know Javascript, what if we wrote our servers with it?
- Led to the disaster that is NPM
- Can be credited with popularizing “full stack” development

Putter Panda is First Observed (2009)

- Hacking group out of China
- Focus on satellite and communications technology
- Steals trade and military secrets
- Primarily make use of spear-phishing

The Lazarus Group is first observed (2009)

- The North Korean state sponsored hacker group
- Responsible for breaches of corporate networks
- Tried to delete the movie *The Interview (2014)*

The MicroService Era

Angular (2010)

- Webpages are hard, so what if there was a way to act like they used older ideas?
- Angular was the first popular framework to emerge for building web applcations
- Combine individual pieces componentwise

Stuxnet is First Observed (2010)

- Malicious cyber worm developed by the USA
- Hyper-targeted malware which could destroy nuclear centrifuge's
- Used to cripple the nuclear program of Iran

Microservices (2012)

- Having everything in one massive monolith makes things a little difficult, wouldn't it be cool if we could write small, individual parts and plug them all together?
- When they work, it's almost like magic
- When they don't, it's often nearly impossible to figure out why
- Work well for massive organizations where engineering teams aren't able to directly communicate all the time
- Rely on rigorous design documents

Electron (2013)

- Writing native applications is hard
- Wouldn't it be nice to just release your webpage as an app?
- Electron is chromium bound to a single application
- Now everything can be web development :)

Docker (2013)

- Put work into a “container”
- Containers “run the same anywhere”
- Goes hand-in-hand and enabled with microservices

MAN, DOCKER IS BEING USED FOR *EVERYTHING*.

I DON'T KNOW HOW I FEEL ABOUT IT.

STORY TIME!

ONCE, LONG AGO, I WANTED TO USE AN OLD TABLET AS A WALL DISPLAY.

I HAD AN APP AND A CALENDAR WEBPAGE THAT I WANTED TO SHOW SIDE BY SIDE, BUT THE OS DIDN'T HAVE SPLIT-SCREEN SUPPORT.

SO I DECIDED TO BUILD MY OWN APP.

I DOWNLOADED THE SDK AND THE IDE, REGISTERED AS A DEVELOPER, AND STARTED READING THE LANGUAGE'S DOCS.

...THEN I REALIZED IT WOULD BE *WAY* EASIER TO GET TWO SMALLER PHONES ON EBAY AND GLUE THEM TOGETHER.

ON THAT DAY, I ACHIEVED SOFTWARE ENLIGHTENMENT.

BUT YOU NEVER LEARNED TO WRITE SOFTWARE.

NO, I JUST LEARNED HOW TO GLUE TOGETHER STUFF THAT I DON'T UNDERSTAND.

I... OK, FAIR.

React (2013)

- Currently the largest Component Library
- Instead of writing individual components, you can build them up like a model kit
- Streamlines frontend building

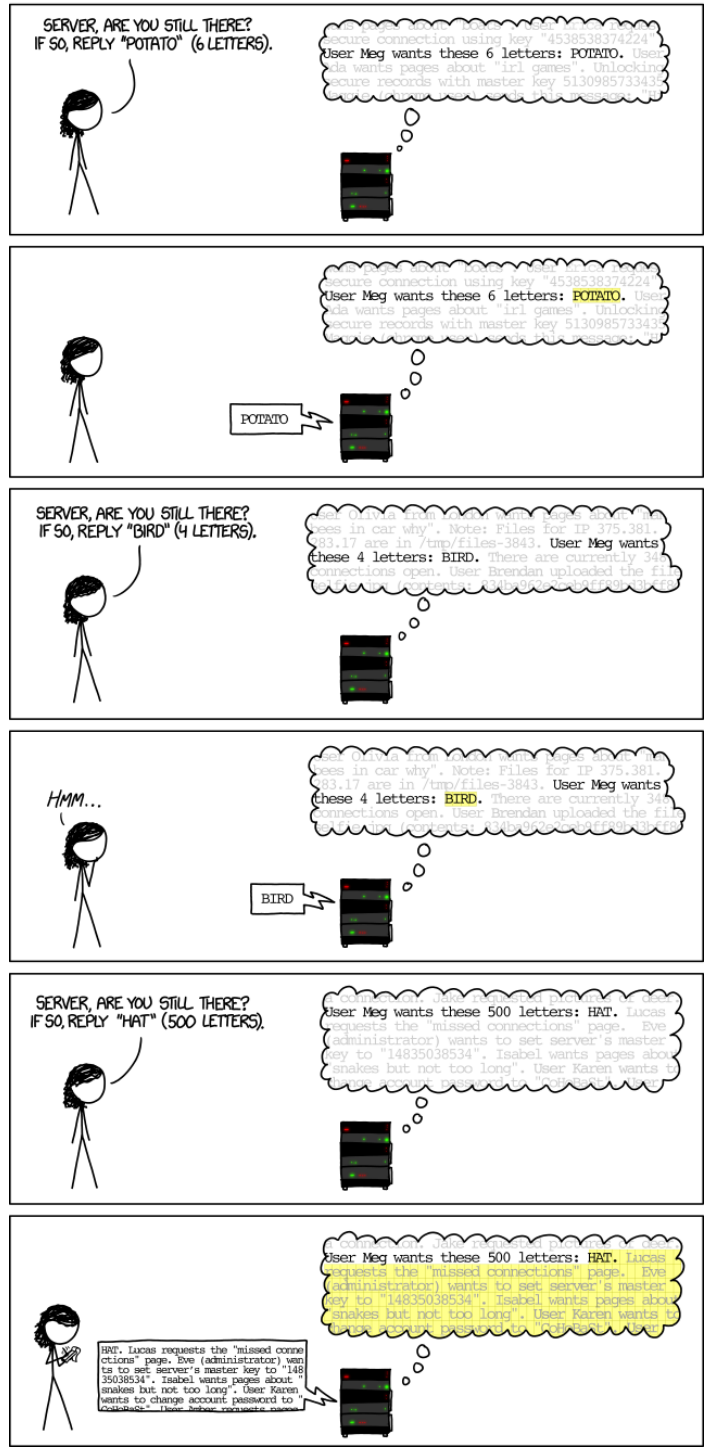
Footnote: React-native (2015)

- What if we took electron and its premise but made it for mobile?
- Write React code that runs a mobile app instead of a website

Heartbleed (2014)

- One of the biggest vulnerabilities of the last decade
- Threatened virtually every server on the planet
- A common example in the textbook

HOW THE HEARTBLEED BUG WORKS:



Solidity (2014)

- Runs on the Ethereum Virtual Machine
- Currently v0.8.17
- “Smart Contracts”

The Emerging Serverless Era

Serverless (2015)

- The continuation of the move from:
  - Infrastructure aaS
  - Servers aaS
  - Functions aaS
- Began with the Serverless Framework

```
func Handler(request events.APIGatewayProxyRequest)
    (events.APIGatewayProxyResponse, error) {

    name := request.PathParameters["name"]
    message := fmt.Sprintf(
    " { \"Message\" : \"Hello %s \" } ", name
  )

    return events.APIGatewayProxyResponse{
    Body: message, StatusCode: 200
  }, nil
}
```

Equifax breach (2017)

- 147.8 million private identifying records were exfiltrated
  - Terrabytes of data
- Settlement worth $425 million paid out by Equifax

How did it happen?

1. Entry through a consumer complaint portal using a known, but unpatched vulnerability
2. Moved to “secure” servers due to lack of segmentation
3. Unrenewed certificate on Equifax’s end let them exfiltrated encrypted data for months
4. Executives started dumping shares and the breach wasn’t publicized for more than a month after discovery

PittyTiger is first observed (2017)

- PittyTiger, one of the most recent APTs is first seen
- Emerged 2017
- Targeted AIRBUS Defense & Space
- Corporate Espionage
- Assumed not State Sponsored

Crypto Bubble (2021)

- Despite interesting technology and good engineering, there was no real demand
- Crashed losing multiple billions of dollars for everyone involved
- Massive fraudsters at FTX at a level not seen since Enron

Lastpass Breach (2022)

- First breach in August of '22
  - "no customer data was accessed"
  - Later announced source code and technical information were stolen
- Later, customer "vaults" were extracted
  - "a proprietary binary format that contains both unencrypted data. . . as well as fully-encrypted sensitive fields"
- Lastpass recommends no updates at this time
- Security Professionals recommend changing all passwords in Lastpass, enabling 2FA anywhere its available, and migrating to a different password manager

SvelteKit (2022)

- Web framework
- Handles all the major things like auth, routing, distribution, compilation, bundling, etc.
- Major competitor is Next.js, time will tell which becomes the most common

Major Trends

Problems in Software Design

1. Create a product that people have a need for
2. Create a product that is able to live on
3. Create a product that protects the people involved with it

We can make a tool for that!

- When we encounter a problem, it's common for out solution to be, let's build a tool to do that
- This mirrors the abstraction that comes so naturally to software engineers
- Abstracting out lower layers leads to multiple assumptions

Assumptions:

1. Everything works as it's intended to
2. Security issues at lower levels are handled
3. We can operate without an explicit understanding of lower layers

Fact:

1. ~~Everything works as it's intended to~~
2. ~~Security issues at lower levels are handled~~
3. ~~We can operate without an explicit understanding of lower layers~~

Abstraction

Even the concept of a computer undergoes abstraction in our model.

Room sized computing devices -> mainframes -> computers -> servers -> IaaS -> SaaS -> FaaS

Why Design in Security

Major Threats

- APTs
- Crackers & Piracy
- Breaches
- Modern DDoS
  - Azure DDoS of 3.47 Tbps for 15 mins ~3 Petabytes of data ($63k alone just to store)

"The attacker used several networks to spoof 167 Mpps (millions of packets per second) to 180,000 exposed CLDAP, DNS, and SMTP servers, which would then send large responses to us. This demonstrates the volumes a well-resourced attacker can achieve: This was four times larger than the record-breaking 623 Gbps attack from the Mirai botnet a year earlier." - Google Press Release

"In November [2021], Microsoft mitigated a DDoS attack with a throughput of 3.47 Tbps and a packet rate of 340 million packets per second (pps), targeting an Azure customer in Asia. We believe this to be the largest attack ever reported in history.

"This was a distributed attack originating from approximately 10,000 sources and from multiple countries across the globe, including the United States, China, South Korea, Russia, Thailand, India, Vietnam, Iran, Indonesia, and Taiwan. Attack vectors were UDP reflection on port 80 using Simple Service Discovery Protocol (SSDP), Connection-less Lightweight Directory Access Protocol (CLDAP), Domain Name System (DNS), and Network Time Protocol (NTP) comprising one single peak, and the overall attack lasted approximately 15 minutes." - Azure Press Release

1. Attackers are getting more and more capable
2. Increasing numbers of people are conducting increasingly important tasks online
3. Malicious actors are incentivized to get in
4. It only takes one exploit to invalidate any one security

Questions?