Software Engineering
Through the eyes of a hacker, academic, employee, and CEO

Chad Spensky
chad@allthenticate.net
Founder and CEO of Allthenticate
My Journey

1990s: Internet pirate, hacker, and master tinkerer

2004-2008: College student at Pitt

2008-2011: PhD student (and dropout) at UNC - Chapel Hill

2012-2015: Research Staff at MIT Lincoln Laboratory (DoD work)

2015 - Present: PhD student at UCSB in the SecLab

2015 - Present: Member of Shellphish (CTF team)

2019 - Present: CEO and Founder of Allthenticate
Software Engineering

My definition:

Building software that is built to last, easy to share, amenable to collaboration, and has long-term maintenance in mind.
Alias: Shortman

Skills: Site Tech, Eggdrop programmer, Founder of the best "precheck" in "the scene"

Programming Languages: SQL, HTML, TCL, some C, mIRC scripting
Software Engineering for Hackers

**Version Control:** Hard No

**Budget:** Unlimited games and movies

**Hosting provider:** My basement server

**Developers:** Me

**Release structure:** as needed

**Format:** tarball with l33t README file, ASCII art was more important than content
Other high-school projects

**Porganizer**: Visual Basic .NET program that prints your weekly or daily schedule in the morning

**Porganizer on the Go**: An online organizer that interfaced over SMS to keep track of calendar events (pre-G Calendar)

**Carputer**: An in-car-computer that would automatically sync my downloaded mp3 files with my computer (pre iPhone)

**r0x0rs.us**: An online video upload site, targeted at funny videos (pre YouTube and CollegeHumor)

**Music Anywhere**: An in-home networked media player to play music in every room (pre Sonos)

**XBOX Modding**: A fun side business
Tooling: Highschool Hacker

- mIRC (Polaris plugin)
- Writing websites in Notepad.exe
- Scripting in pico and vim
- Hard drives and partitions everywhere
- Soldering Iron
Time Allocation (High School)

- Social life: 19%
- Building relationships on IRC: 30%
- Building software: 40%
- Hardware hacking/building: 10%
- School work: 1%
Undergraduate

**Degrees:** B.S. in CS (Honors), B.S. in Mathematics, Minor in Economics

**My take:** Universities can kill creativity

- Learned a lot of “science”
- Learned how to pronounce computer terms (e.g., “my-SEE-QUAL” and “TICKLE”)
- Stopped “engineering” things, and had effectively zero side projects
- Attended some epid parties, and made some great friends

**Software Engineering Required:** No
Tooling: Pitt

Eclipse (Pitt was a Java school)

Still doing stuff in Notepad

Because it thought it was l33t

Books and pencils...
Time Allocation (Pitt)

- Social life 50%
- School work 49%
- Side projects 1%
Graduate School (round 1)

Degrees: M.S. in Computer Science (Security), Ph.D. Dropout

My take: Graduate school can be amazing if you like the project that you are working on

You actually have time to build something great

Too much emphasis on “science” and “research,” which are very poorly defined

Tried to organize a class to teach git after my internship; it didn’t happen

Software Engineering Required: Yes! (but no one seems to think so)
Tooling: UNC-CH

Dropbox to sync files with home computer

Subversion for version control

No shared repositories in our group

Definitely no test scripts

Bugs galore
Time Allocation (UNC-CH)

- Social life (Football) 30%
- "Research" 69%
- Side projects 1%
Title: Associate Staff in the Cyber System Assessments (Offensive) Group

My take: The best environment to be in as a software developer

- Very interesting projects
- Smallish teams (2-20)
- Prototypes do not have to be “production” quality

Software Engineering Required: Definitely! (I felt very ill-prepared)
Holy resources! I got my own 7 server cluster (~24 cores each) with a single email

Introduced to Github Enterprise

Tiled window managers! A must!

Equipment makes a huge difference

2 OS > 1: One pretty, one useful
Title: [REDACTED]

My take: The internet is amazing!

   Things come in, but never come out

   No internet

   Every tool needs to be approved (and takes forever to approve)

   What the heck is git archive?

Software Engineering Required: You betcha
Tooling: TS

DVD Burners

Programming books! (They actually exist...)

Offline versions of online docs

Thinking on your feet is critical

You better “really” know your programming languages
Time Allocation (MIT LL)

- Meetings/Administrivia: 30%
- Software Development: 50%
- Powerpoint Engineering: 20%
Graduate School (round 2)

**Degrees:** Ph.D. in Computer Science (Securing and Analyzing Embedded Systems)

**My take:** Got to work on some really awesome, complicated problems

- Repeatability is really important
- Experiments and continuous integration (CI) aren’t very different
- Open-sourcing code makes you a better programmer (others will see it)
- Submodules are a must!

**Software Engineering Required:** Yes! (but no one seems to think so)
Tooling: UCSB

Time to “pro up”

I3 + Terminator

Pycharm, Clion, ... (IntelliJ)

TexShop

Internal Gitlab

direnv + virtualenv  a must have
CTF Player

Title: N00b hacker

My take: An incredible experience to a lot about alot in very little time

Like drinking computer science from a fire hose

Much more than just “hacking”

Stresses your knowledge about how computers work (like... that the even turn on)

Software Engineering Required: Maybe?
Software Engineering in a CTF

Speed over correctness

Correctness is extremely important

Speed is also important

Extensibility isn’t important, but it also might be

from pwntools import *
Tooling: Shellphish

IDA Pro, Ghidra, Binary Ninja, Radare: Collaboration is a mess!

Git with some special sauce to “throw” exploits to “grill” the other teams

Slack or Discord with a different channel for every challenge

Physical separation of teams for each challenge

Complicated networks for sharing “floor” data with people in the suite
Time Allocation (UCSB)

- Reading/Writing  40%
- Software Development  30%
- Meetings  10%
- Side projects  20%
**Title:** PhD Research Intern

**My take:** Big things move slowly and have a lot of moving parts

- The resources were incredible! More cores than you could ever want
- Lots of amazing coworkers and internal knowledge
- Took 3 months to acquire the hardware required for my research

**Software Engineering Required:** Yes. This has to work on my computer back at UCSB
Tooling: IBM Research

Apparently you can do software development on a Mac, although I wouldn’t recommend it

SizeUp (kind of allows for tiled windows)

Starting to doing VIM practice to pro up

VS Code! Love it. (but not for the Python yet)

Parallelizing Python is way to hard, still

Spent my evenings re-organizing git repositories for my real passion...
Time Allocation (IBM)

- Reading/writing: 19%
- Software Development: 80%
- Meetings: 1%
What am I doing with all of this?

(Insert impressive company pitch here)

A smartphone-based solution.

Chad Spensky | Allthenticate.net | chad@allthenticate.net
Authentication is making us miserable.
It’s time for a revolution.

76% of businesses were victims of phishing last year

Avg. cost of data breaches is nearly $4 million per business

80% of hacking-related breaches tied to passwords

Existing readers cost over $2,500 per door

Upgrades require replacing the reader and issued cards

Proximity cards are easily forgotten, lost, or stolen
Chad Spensky, CEO
Ph.D., Computer Science (Security)
MIT Lincoln Laboratory
IBM Research
IBM PhD Fellowship recipient
15+ academic publications

Rita Mounir, COO
B.S., Financial Mathematics and Statistics
Carpe Data
Center of Academic Achievement
Startup Weekend organizer
1st place Port Hueneme Startup Weekend

Evan Blasband, CTO
M.S., Electrical And Computer Engineering
Lockheed Martin
Best UCSB EE project
1st place SpaceX Hyperloop Competition
1st place UCSB Startup Weekend

We have been developing this patented technology for 8+ years
Single Device Authentication

One credential for all – digital & physical
Supports any interface
Resistant to software-based attacks
How it works

Unlock Doors

Secure Interaction

Remote Services

Computer Logins

Unsecure

Secure

Patented
A Secure Foundation

Untrusted Software
- Apps
- Programs
- Software
- Services
- Phone OS
- Reader OS
- Computer OS
- Server OS

Trusted Hardware
- Trusted IO
  - Secure Interactions
- Trusted Execution Environment
  - Secure Processing
- Secure Element
  - Secure Credentials

Our Code

Patented
How we do it

Cloud-based Management
One-stop authentication stop

Comprehensive Logging
Per employee entries
Digital and physical
Unforgeable

Device Management
Laptops
Phones
Doors
(Anything)

Flexible Policy
Create user groups
Time of day
Location etc.
Interface

SDA

Althenticate
A smart phone-based solution.

Verify Your Email

Jane Doe

Front Door
Conference Room A
Work Desktop
Company Car
Personal Laptop
Any Device
Customizable Security

FLEXIBLE POLICIES

Time or Day
Location
Delegate Resource
Temporary Access

FLEXIBLE SECURITY INTERACTION LEVELS

Things magically open (lowest)
Intent to do something (intermediate)
Prove identity (highest)
# Feature Comparison

<table>
<thead>
<tr>
<th></th>
<th>PHYSICAL</th>
<th>DIGITAL</th>
<th>HAS IT ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Legacy Solutions</td>
<td>Smartphone-based MFA solutions</td>
<td>ALLTHENTICATE</td>
</tr>
<tr>
<td>Flexible Security</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Backward Compatible</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Simple Installation</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Over-the-Air Upgrades</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Eliminates Phishing</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Smartphone-Based Solution</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>No Internet Required</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Cross-domain Solution</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Price</td>
<td>$$$</td>
<td>$</td>
<td>$$</td>
</tr>
</tbody>
</table>

ALLTHENTICATE

**HAS IT ALL**

- Backward Compatible
- Simple Installation
- Over-the-Air Upgrades
- Eliminates Phishing
- Smartphone-Based Solution
- No Internet Required

**Price**

- $$$
- $$$
- $
- $$
- $$
- $$
Allthenticate Yourself — Future state

Let’s get you set up quickly.
(Transition back to ugly slides)
Allthenticate (a cybersecurity startup)

Title: Founder and CEO

My take: WWWHHHHHEEEEEEEEEEEEE!!!!

- Serious tradeoff between moving quickly and doing it “right”
- Sound software engineering feels more expensive than ever
- Managing a company is harder than managing a team
- People are harder to coordinate than software

Software Engineering Required: Your company will certainly fail if you do not.
Allthenticate Internals

5 distinct products

40 gitlab repositories

Java, C++, Objective C, Dart, Javascript, CSS, SQL, Python 2 & 3, Bash scripts

Cross-compiled native libraries for every iOS and Android architecture

Environments supported: OSX, Linux, Raspberry Pi, Windows, iOS, Android, Chrome
Allthenticate Manufacturing

We design our own hardware from scratch

PCB design and testing

  Outsourced fabrication and placement (something you only want to do once)

Mechanical design

  Designed in house, printed externally

Hardware debugging is much harder than software debugging (software developers have it easy)
Management Tools

JIRA, Asana, Trello, ...

Gitlab, GitHub, ...

Wikis, Issues, ...

Slack, Discord, ...

Meet, Zoom, ...
What we use

**Trello:** All technical issues, administrative issues, and hiring

**Gitlab:** Free runners mixed with custom runners (e.g., a Pi and Mac mini)

**Slack:** Sharing memes

**G Suite:** email, conferencing, and files

  Google drive and slides are life savers!

Hardware tests require real hardware (*Phones as Pis*)

**Payroll software, Quickbooks, and Zapier**
CI/CD Awesomeness

Deploy keys are amazing!

ssh integration with CD is next level

Have you tried Netlify, or a similar CMS?

Linting in CI

Submodules!

Branch-dependant stages
My typical day

Sleeping 8 hours (+/- 30 min)

Exercise 1-5 hours

Administrivia 1-35 hours

Engineering 0-10 hours

Eating 1-2 hours
Lessons Learned

Learn git, really learn it, and use it as properly as you can

Do CI early and often

Practice, take time to not program, but make yourself more efficient

Ergonomics is important

Spend the money, don’t compromise. You only get 1 body

Invest in good equipment.

You should never be held back by your equipment. It's too cheap to suffer.
Be nice to your colleagues and future you

Just because you “can” do something in a language, does not mean that you “should”

Pythonic code should only be used if it makes the code more readable, faster, or more extensible.

Function that returns the set of all subsets of its argument

\[ f = \lambda x: \{ \{ y \text{ for } j, y \text{ in enumerate(set(x)) if } (i >> j) & 1 \} \text{ for } i \text{ in range}(2**\text{len(set(x))}) \} \]

No!
Questions?

chad@allthenticate.net