# Danny Chen

+1 512 629 0747
dannychen579@utexas.edu

## EDUCATION

*Bachelor of Science in Computer Science* *2012-08/2016-05*
University of Texas at Austin
Coursework: Operating Systems, Cryptography, Algorithms, Network Security, Artificial Intelligence, Automata Theory, Theory of Computation

## TECHNICAL SKILLS AND KNOWLEDGE

*Languages:* C, Bash, Python, x86 Assembly, Java, Javascript, Golang, Ruby
*Technologies/Tools:* Amazon EC2, Amazon VPC, Git, Travis CI
*Operating Systems:* Linux

## EXPERIENCE

*Senior Software Engineer* *2021-08-02/Present*
Speech Runtime, Rev.ai

- Helped build a high performance automatic speech recognition (ASR) application
- Implemented a feature that allows for customized language models to our speech-to-text service

*Software Development Engineer II* *2016-07-05/2021-06-28*
EC2 Networking, Amazon Web Services

- Worked as part of a team that performs routing and packet translation for all network traffic going in and out of EC2 worldwide.
- Worked on and maintained a low-level packet processing application written in C.
- Implemented the network security layer for Inter-Region VPC Peering.
- Worked on a 24/7 oncall shift to troubleshoot hardware and software failures in production.
- Designed and oversaw projects to improve operational overhead and alleviate technical debt, often in conjunction with other teams in AWS.

*Software Development Engineer Intern* *2015-07-02/2015-08-28*
Platform Excellence, Amazon.com

- Worked as part of the team that monitors latency on the Amazon.com website.
- Created a data visualization tool for a terabyte-scale Redshift cluster.

*Software Developer Intern* *2014-07-02/2014-08-27*
Software Verification Team for WebSphere Application Server, IBM

- QA and tested IBM WebSphere.
- Wrote scripts to do automated testing of new releases of Websphere.

## PERSONAL PROJECTS

*HeapChecker* - https://github.com/SrsBusiness/HeapChecker

- Proof of concept for a strace-style tool that traces a target program's heap allocations
- Handles ELF binaries for x86_64 System V that use a limited set of dynamic linking types

*Minecraft Client* - https://github.com/NosotrosNueces/mcc

- Collaborated with friends to implement the Minecraft Client protocol.
- Provides an API for sending commands to the server like digging and moving.
- Allows for asynchronous handling of messages from the server through user-specified callbacks. Backed by libuv, the same library that handles asynchronous I/O for Node.js.
- Implemented simple bots that can break blocks, attack enemy entities, and build structures.

*Gameboy Emulator* - https://github.com/mukkid/GoBoy

- An active collaborative project with a friend to implement software emulation of the Gameboy console.
- Completed full emulation of the Z80 CPU
- Adopted and enforced test-driven development and CI practices.

*Chess AI* - http://github.com/SrsBusiness/Party

- Game state is represented using bitboards and transitions between states are implemented efficiently with bitwise operations and table lookups.
- Uses AI techniques such as alpha-beta pruning, transpositional hash tables, and iterative deepening.