Jason Sun's Resume

jason.sun@uwaterloo.ca	sunapi386.ca	+1 (519) 500-2969	Compiled on 07/02/2015

About Me

- *University of Waterloo* Computer Science undergraduate in 4B term, previous exchange student at *Swiss Federal Institute of Technology (EPFL) at Lausanne*. Graduating 2015.
- Very driven and motivated with many hobbies (visit my blogs at *blog.sunapi386.ca*), especially into computer science, which I discovered from trying two previous majors: physics, business.
- Over eight years of experience using unix: Ubuntu, Arch Linux, OS X and unix utilities like grep, fdisk, etc.
- Experienced working in startup environments, with five terms at Velocity residence. Enjoys tech events, like *DEFCON 22*, *2600 Hope #9*, MHacks, HackMIT, PennApps, HackZurich 2014.
- Won hackathon prizes at:
  - *HackZurich 2014* at ETH Zürich, received Tamedia Digital Award (all-inclusive team trip to visit startups in Berlin, Germany) with *.GIFMeIt*: An iOS app that lets a user easily capture and share GIF images.
  - *PennApps 2013* at University of Pennsylvania, received Twilio's Communication Award ($500) with *Marmoset*: A chatbot to respond to your chosen Facebook friends without them knowing.

Notable and Interesting Projects

- Advanced Algorithms class (EPFL CS 450): A first graduate course in algorithms, learned theoritical techniques and their applications to solve problems. Interesting techniques such as network flow, randomization, dynamic programming.
- Intelligent Agent class (EPFL CS 430): Developed 5 agents, each using different strategies to pickup and deliver a parcel in a simulated environment: reactive, deliberative, centralized, decentralized, auctioning.
- Concurrency class (CS 343): Multithreaded quicksort, token ring network simulation in ***uC++***, a dialect developed at University of Waterloo supporting concurrency.
- Distributed Systems class (CS 454): Built a remote procedure call library in **C++** over TCP, for both the client and server side. Implemented the go-back-N reliable transmission protocol, over UDP using **Java**.
- Computer Security class (CS 458): Created exploits in **C** using vulnerabilities such as buffer overflow and format strings. Implemented an intrusion detection system in ruby that parses output from tcpdump to detect spoofed packets, malicious hosts, and worms.
- Compilers class (CS 251): MIPS compiler using context-free parsing to generate **MIPS assembly** code. Also designed a simple pipelined CPU written in **Verilog**, supporting 8 instructions for computer architecture class. Theoretically this is sufficient to run my machine code produced by my MIPS compiler.

- Personal project: One of the larger scale is Dotabuff-ripper. Written in **Ruby** to aid the counter-hero picking in 5v5 dota games. A scraper collects about hero winrates from Dotabuff and inserts into a ***Neo4j* graph based database**. The tool then suggests a list of potential counter-picks. Meant to have a web interface, but the web component was never finished.

Experiences

- Software Engineer Intern *Shutterfly Inc.* in Redwood City, California

Develop functional load tests for distributed services. Design and implemented a distributed key value storage service, using technology like **Jersey** RESTful Web Services framework and Apache **Cassandra**.

- Software Developer at *Encircle Inc.* in Kitchener, Ontario

A startup in the*Velocity Garage*, worked in **java**, **coffee**, and **python**. Worked on the web and android application stacks.

- Undergrad Research Assistant at University of Waterloo, in Waterloo, Ontario

Working with professor and experimented acquiring input from a NI myDAQ, a low-cost data acquisition device, and some data analysis using **Matlab**.

- Software Tools Developer Intern at RIM (BlackBerry) in Ottawa, Ontario

Built features to the GitLab open source project (**Ruby on Rails**). Developed a testing framework for testing website user interfaces, using the **Selenium** Java framework.

- Physics Teaching Assistant at *Wilfrid Laurier University* in Waterloo, Ontario

Developed a spectrometer reading program in python, using the **pySerial** library, and automate queries over serial port - previously you had to punch numbers on the machine.