Kaleb Smart

Senior Software Engineer

(256) 506-8537 | kalebsma@gmail.com | yeyande | kaleb-smart

Education

University of Alabama in Huntsville *Huntsville, AL*

Bachelor of Science in Computer Science, Magna Cum Laude *May 2016*

- Minor in Mathematics
- Minor in Russian

Skills

Technologies	Git, Linux, OpenStack, Jenkins, Packer, Artifactory
Languages	Python, Groovy, JavaScript
Frameworks	Pytest, Behave, Angular, Flask, Docker, Laravel, Ansible, Jinja, Celery

Work Responsibilities

ADTRAN - Senior Software Engineer *Huntsville, AL*

Telecommunications Vendor *June 2014 - May 2020*

- Designed and architected tooling for developers to work on customer-oriented deployments.
- Served on DevOps governance team to architect a company-wide vision of Continuous Integration/Continuous Deployment pipeline.
- Spearheaded the development of architectural runway and Developer Experience tooling as Scrum Master for system teams.
- Collaborated with developers in Canada, Germany, and India to coordinate new product development efforts.
- Architected and Maintained test aggregation and hardware resource management software for Continuous Integration pipelines.

Projects

Hackathons

- Reduced build time of SDX-6210 software from 2.5 hours to 45 minutes by designing and implementing binary packages for infrequently changing code, then implementing a process to consume those in the product build.
- Containerized services running on SDX-6210 hardware to lay groundwork towards network feature virtualization for the product.
- Created proof of concept for using Alpine packages to greatly simplify internal package generation and library linking resolutions.
- Automated the population of Common Vulnerabilities and Exposures in generated product security document, greatly reducing the time required to release new product security documents.
- Deployed a Github code review tool written in Python to keep track of requested changes to largely replace dated third party code review software.
- Worked on proof of concept Python Github bot to automatically create pull requests to integrate package dependencies to provide pre-commit integration on dependency chains.

SDX Aggregation Switches

- Architected Jenkins pipeline modifications to test common Behave feature sets on virtualized aggregation switch from on a common software image and the fan out procedure for product-specific acceptance tests on feature sets. This improved test stability, and when failing production code was entered, it failed faster.

Total Access 5000

- Architected layout and methodologies of testing framework using Pytest for TA-5000 GPON OLT line cards to be used in the Jenkins CI pipeline.
- Developed automated provisioning of test assets for a given testbed using Jinja and Ansible, which included SIP and TFTP servers with configuration options provided from the testbed definition.
- Redesigned lab network to allow for strict layer 3 isolation between testbeds in order to prevent network outages from misbehaving software. This network layout was adopted in 4 separate labs.
- Designed testbed configuration restoration process in Ansible to ensure testbeds were in a known default state prior to beginning tests. This made tests more reliable and gave confidence that failures were due to the product itself.

Release Notes and Security Documentation

- Designed generic templating Python software for automatically generating release notes and product security documentation for any product, which was adopted by all the Software Defined Networking products, reducing the time required to create and approve release notes from 2 weeks to 1 day.
- Augmented release note generation to populate new, longstanding, and fixed software issues from ticketing system which reduced the time required to create and approve release notes from 2 weeks to 1 day.
- Developed software to process internal security scans and tests to populate product security document with known security vulnerabilities which led to security investigations and the refreshing of automated security tests.

SDX-6210

- Containerized cloud-based network element controller software to decrease software upgrade time by a factor of 6.
- Developed system level verification pipeline and test framework in Python and Robot Framework to validate the SDX-6210 and the cloud-based controller software against customer requirements prior to software release. Sales engineers used the test results as a source of truth for what features could be demonstrated to a customer.
- Developed a CI pipeline information radiator and value stream map using Python's Celery and Flask libraries to visualize how software was flowing through our pipelines which resulted in additional efforts that both reduced the amount of time for software to be ready for release by 2 hours and increased the reliability of the CI pipeline by at least 20%.
- Spearheaded task force to inspire collective ownership of product's CI pipeline, reducing the amount of failures caused by infrastructure to under 10%.
- Developed deterministic automated build procedure and integration for acquired and newly developed software.
- Created test asset diagnostics Python tooling which would dump application state and logs to quickly identify the cause of test failures.
- Designed Python tooling replicatable hardware configurations based around customer deployments of our products using Jinja and Ansible, allowing our entire acceptance test suite to run on any of our testbeds, and reducing the testbed bringup time by a factor of 8.
- Integrated the SDX-6210 product into existing Continuous Integration (CI) pipeline to include tests already written for existing product features.
- Designed automated JIRA ticket creation and triage process for CI pipeline failures to collect granular metrics and direct the goal of the CI pipeline task force each iteration, which was adopted by product management to drive release readiness meetings.
- Designed high level Python library to interface with traffic generator hardware for use in CI testing.
- Created automated Jenkins deployments for DNS docker containers triggered and populated off any changes of the testbed definitions, which was regularly used by about 1/3 of the company.
- Deployed a network of ONOS controlled whitebox switches to validate solution interoperability with an ONOS managed network.
- Deployed Nagios monitoring software to collect health metrics on testbeds and their infrastructure.

MOSAIC OS

- Developed Jenkins CI pipeline to allow for feature tests to be consumed via a product's capabilities along with generic high level Python libraries to enable developers to write product-agnostic feature level tests, which was the standard CI pipeline for the company for about 4 years.

Skynet and Hydra

- Developed testbed inventory and reservation service. It was adopted universally in the company by Continuous Integration pipelines, developers, and product stakeholders that had physical hardware requirements to visualize and control the priority, allocation, and utilization of testbed resources.
- Designed test results aggregation software and trend visualization written in AngularJS and Laravel's PHP REST Framework that developers and product stakeholders used to ascertain the reliability of a software build or test suite.
- Created high level groovy and python APIs to allow developers to interact with these services in the same way as the CI pipelines.