SPAWAR
Systems Center
PACIFIC

Using Automation to Speed up Software Delivery to the Fleet
ICCRTS, 2018

Samia Ali
SSC Pacific
6th November, 2018

SPAWAR Systems Center PACIFIC

Outline

- Background
- Objectives
- Navy relevancy
- Use-cases
- Technical highlights
- Technical challenges

SPAWAR Systems Center PACIFIC

Background

- In the Navy's C2 Domain, there are hundreds of different environments – from large, well-connected running on shore to small networks operating aboard a single ship - where Navy software needs to run. The rising cyber security threats adds onto the challenge.
- Need for regular software updates, installed with minimal delays, and relying on trained engineers makes it very difficult because of cost concerns and the sheer number of environments.
- Seamless code flow from development, through deployment, to testing and production is essential for effective software development and the deployment of updated bug free software. A bottleneck in any of these phases leads to delayed schedules, increased cost, and poor performance.

Objectives

SUDOE was designed with the goal of delivering software capabilities in a few hours to the warfighter. The tool aims at achieving this by automating the process of orchestration, configuration and installation of software services.

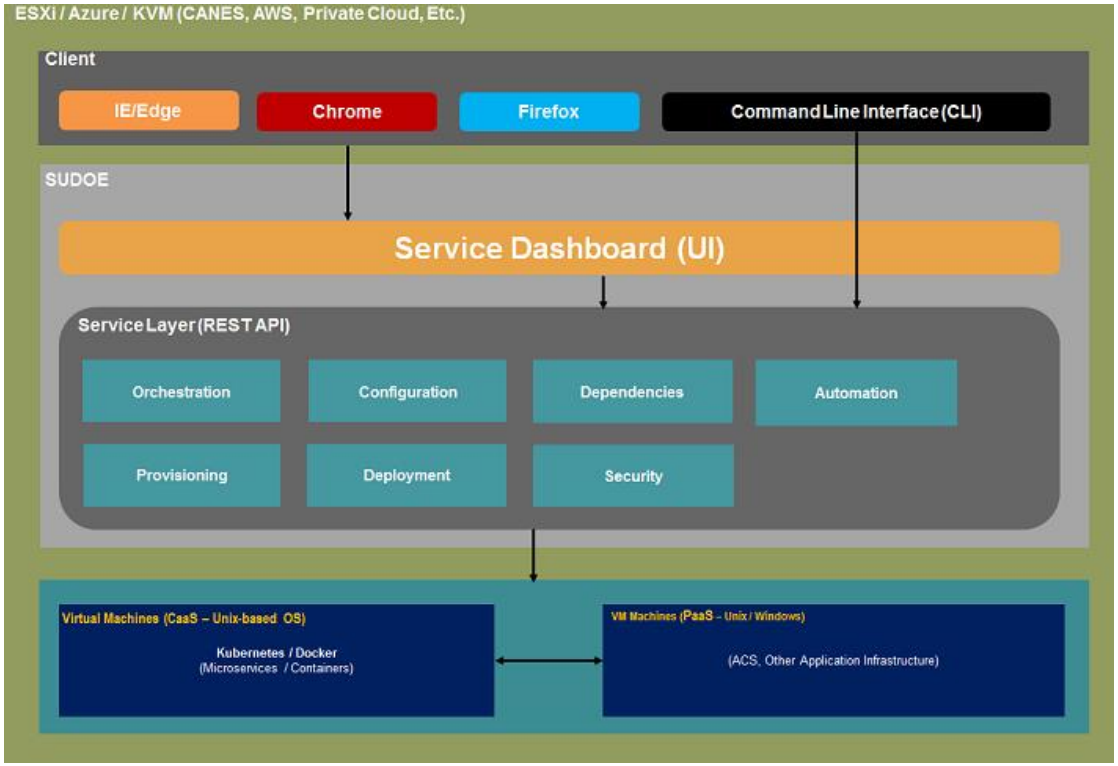
SHIP

Ship: Hey SUDOE, I need the latest patch of Software-x
SUDOE: Roger that.
… few minutes later …
SUDOE: The latest version of Software-x is available for you now.
Ship: Awesome! Thanks.

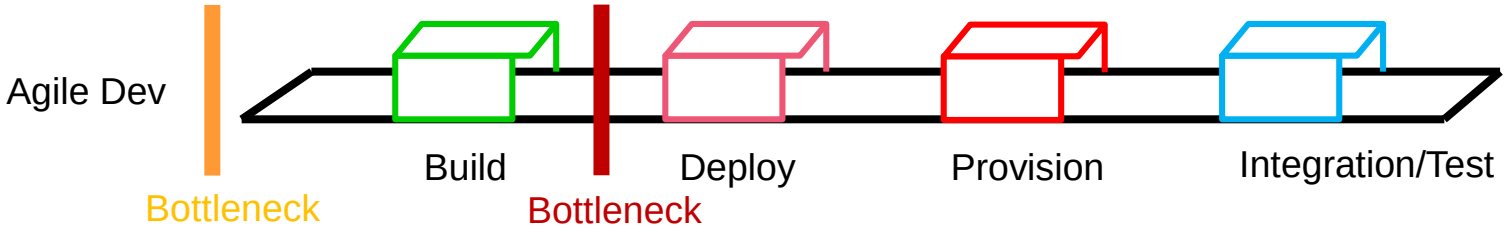
SUDOE

What is SUDOE?

SUDOE is a government-developed software suite of tools for fast, reliable, secure and automated deployment of software and infrastructure. It leverages technologies such as virtual machines, containers and micro services.



Navy Relevancy



The Navy faces many challenges in the area of software deployment

- Lack of Automation.
- Lack of Advanced Infrastructure Technologies.
- Navy Software systems are very complex and designed around monolithic architecture.
- Manual process to comply with numerous cyber security requirements and support different enclaves.

SPAWAR Systems Center PACIFIC

Underlying Methodologies

DevOps

A cultural and professional movement that stresses on communication, collaboration and integration between software developers and IT operations professionals while automating the process of software delivery and infrastructure changes

Microservices

Microservices architecture is a way of designing software applications as suites of independently deployable services.

- Applications are built as suites of services.
- Services are independently deployable and scalable.
- Fast and easy to deploy.
- Easy to replace.

SPAWAR Systems Center PACIFIC

Underlying Architectures, Frameworks & Technologies

- Kubernetes
- Microservices
- Docker
- Jenkins
- Git
- Ansible
- PKI
- Keycloak
- LDAP
- Restful
- PostgreSQL
- Typescript
- Node
- Sails
- Express
- Angular6
- Bootstrap4

SPAWAR Systems Center PACIFIC

Technologies – deep-dive

Docker Containers

- Eliminate messy dependency resolution with all-in-one package
- Share kernel with neighbors, less virtualization overhead than VMs, more efficient memory and storage packing

Kubernetes

- Orchestrate container stack deployments over multiple nodes with service discovery, health-checking, and auto-healing
- Additional rules for ingress traffic from cluster-external clients and egress to downstream VM-based stateful services

SPAWAR Systems Center PACIFIC

VM Provisioning Stack

Ansible

- Automate VM installation with declarative playbooks
- Place infrastructure and dependency installation under version control as well

Jenkins

- Automation server popular for building artifacts in CI/CD pipeline
- Extend familiar environment to support Ansible deploys as next logical step
- Inherit existing server config through existing plugins

Git

- Place stable Ansible playbooks under source control
- Allows for versioning and rollbacks of VM cluster provisioning

SPAWAR Systems Center PACIFIC

Technologies – deep-dive

Node

- Need for unifying API to prioritize individual connections and transactions rather than processing power
- Plethora of additional modules to support multi-transactions across additional APIs

Angular

- Modular framework enables independent feature development with less integration hassle
- Allows data to be collected across orchestration APIs and collated into a single view of the hybrid deployment

SUDOE Use Cases

Scenario 1: New Team Member

Not Using SUDOE

New member joins the team

Is given a set of software to install and instructions to set up Dev/Test environment

Takes a day or two to set up the Dev/Test environment

All set to make ground-breaking changes

Using SUDOE

New member joins the team

Uses SUDOE to set up Virtual machines or containers needed for Dev/Test environment
... Drinks coffee ...
All set up

All set to make ground-breaking changes

SUDOE Use Cases

Scenario 2: New Features to be Tested

Not Using SUDOE

A new feature needs to be tested

Testing environment is prepped for staging, patch is deployed, takes about a day.

Integration Team: Environment ready for testing
Testing Team: About time!
Testing begins ....

Using SUDOE

A new feature needs to be tested

Tester logs into the testing namespace in SUDOE, deploys container or virtual machine with latest change .... drinks coffee and testing begins.

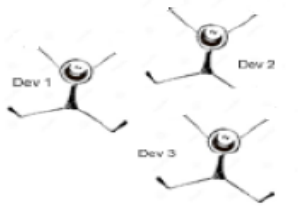
SUDOE Use Cases

Scenario 3: Software Development Lifecycle

Not Using SUDOE

Now that we have this awesome idea for our app and the software spec lets begin

Devloper 1: What database have we decided on?
Developer 2: Postgres
Developer 3: Roger that.
Lets get started.



Yay! it works in the development environment. Its time for integration.

After a week ....
Yay! we are all set in the Ops side! The customers love it.

Developer 1: Oh no! I messed up, we need to revert to the previous version, the current version is a no go.

Integrator: Spends a couple of days to get the previous version of the software and reverts to it.

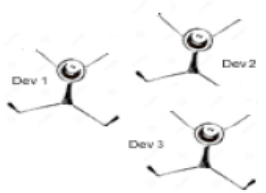
SUDOE Use Cases

Scenario 3: Software Development Lifecycle

Using SUDOE

Now that we have this awesome idea for our app and the software spec lets begin

Developer 1: what database are we using?
Developer2:just use the container x in SUDOE
Developer3:Gosh,I still get nightmares thinking about our days BS(Before SUDOE)



Yay! t works in the dev environment, its time for integration. Let give the integrators our config file to import into SUDOE

Developer 1: Oh no, I messed up. We need to revert to the previous version.

Integrator: Goes into SUDOE, chooses the previous image tag and hits enter.
A minute later …
Integrator: We are good to go!

SPAWAR Systems Center PACIFIC

Results

SNO	Description	Using SUDOE	Time taken	When not using SUDOE	Time taken
1	A developer makes a change to one of the services in the test application	The developer makes the necessary change in SUDOE's GIT and pushes the change. SUDOE's Jenkins build's a container image, the Jenkins uses SonarQube to test the code. The new container image is available through SUDOE's web interface. With once click the new change can be deployed and is seen immediately.	5 minutes	The developer makes the change in code, the code is deployed, then sent for testing. Once it passes testing the code is deployed in production.	2 days
2	The test application has a compromised service which needs immediate attention	When using the microservice architecture, each service is its own independent entity. The developer pulls down the affected service and fixes it without affecting the application. The other parts of the applications except the affected service are available to the users.	1-3 days There is no downtime for the customers	If the application is a monolith, the entire application may be compromised when one service or a part of the application is affected. The development team will need to stop service, pull down the application, fix the issue and re-deploy	1 week The customers don't have access to the application.
3	The user wants to deploy and test new software	The user provides SUDOE with the repo where their scripts to deploy the application live, they use the SUDOE front-end to deploy the application and test it immediately.	A few hours	The user will first need to set up an environment where they can deploy their application. Then they manually install it.	A couple of days.
4	The developer wants to revert to a previous version of the deployed application	The developer uses SUDOE's edit deployment screen and selects the previous deployment from the drop down and deploys and tests the application.	A few minutes	The developer will have to bring down the application and manually revert to the old state. If there were any configuration changes made, the developer will have to take care of them as well.	A day or so.

SPAWAR Systems Center PACIFIC

Technical Highlights

- Automate the process of orchestrating, configuring, and installing software services in a fast, consistent and reliable manner.
- Facilitate user management of virtual machines (VMs), containers, and micro-services.
- Compile, build, deploy application source code files (e.g. microservices, containers) at the target environment
- Cluster-wide deployments (Kubernetes)
- Deploy application binary files (e.g. microservices)
- PKI authentication to automatically create and enforce security models (virtual machine and container)
- Significantly reduce costs, and bring new capabilities/updates to the fleet much faster
- Customized SUDOE can be used in many target environments (Unclassified, Classified, etc.)

SPAWAR Systems Center PACIFIC

Technical Challenges

Integration with existing PKI infrastructure

- Session must be shared among multitude of downstream APIs
- Solution: Have upstream IDP issue short-lived identity token and run group/authz mappings against cluster-scoped LDAP

Extending container namespace restrictions to VMs

- Namespaces impose access, routing, and resource restrictions on containers, ideally should apply to VMs as well
- Solution: Maintain VM membership lists in SUDOE, enforce with combination of namespaced access credentials and hypervisor API

Creating a service to drive Ansible installations

- Developers familiar with monitoring individual Ansible play through terminal, infeasible when driving multitude of cluster deployments
- Solution: Augment Jenkins experience to support Ansible plays as well as artifact builds with additional configs from SUDOE API

SPAWAR Systems Center PACIFIC

Technical Challenges

Raising developer awareness of Docker and Ansible

- Efficiency of orchestration pipeline is directly proportional to the quality of deployment artifacts maintained by app developers
- Teams require more experience with Docker and/or Ansible development for rapid declarative deployments
- Uncertainty of STIGS required for container images

Solution 1: Assist projects with Docker/Ansible transitions

- Suggest local containerization prototypes leveraging docker-compose for development only
- Migrate to kubernetes after container stack is successfully deployed through compose

Solution 2: Present open sandbox SUDOE for experiments

- Offer to provision standalone SUDOE clusters or grant limited namespaced access to shared SUDOE sandbox
- Enables developers to familiarize themselves with the technologies and gauge appropriateness for their own projects as well

SPAWAR
Systems Center
PACIFIC

Questions?