StackStorm

CASE STUDY

Large SaaS Provider Uses StackStorm for End-to-End CI/CD

Enables 12x growth in deploys per developer per day

SUMMARY

In September 2014 StackStorm software was trialed by a SaaS operator for automating common responses to planned and unplanned changes to their operations environment, including changes at the infrastructure, virtual and application layers.

StackStorm entered production in December 2014 in the area of highest immediate need: *the orchestration of an end-to-end continuous integration and continuous deployment pipeline of a mission critical platform*. Thanks in part to StackStorm, this SaaS provider achieved a distinct competitive advantage in boosting their agility by approximately 12x, as measured by the number of developer deploys into production per day.

SaaS PROVIDER BACKGROUND

The SaaS provider has been a leader in first generation SaaS solutions, however, is continually threatened by more agile competitors that can address customer requirements more quickly. Additionally, the SaaS provider's technical operations team had identified considerable technical debt that was hampering its ability to innovate.

The SaaS provider set itself a goal of achieving a 10x, or greater, boost in agility by starting over with a clean slate and creating a separate group running as a virtual startup.

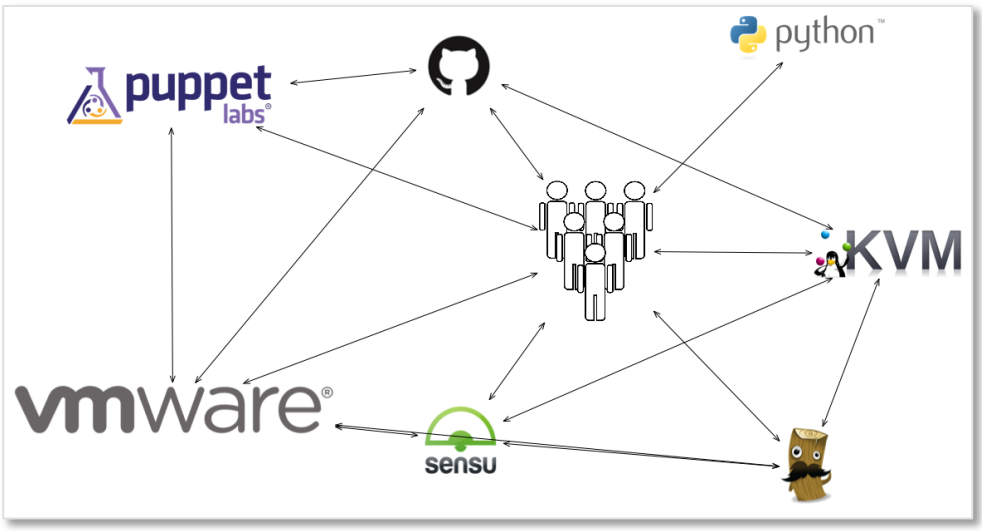
This group was comprised of eight operators with development experience, led by an experienced enterprise operations architect. As a comparison, the existing application was overseen by approximately 250 operators with a diverse set of skills and experience.

The application run by the SaaS provider was more sensitive to operations than many applications as it included real time collaboration components, such as video conferencing. The application consisted of 28 components and grew ranging from

authentication to media servers to the user interfaces and APIs; some of the components ran on bare metal, some on KVM, some on vSphere, and some on Docker. This level of complexity was even greater than that of the existing application for which the SaaS operator is best known.

SaaS OPERATIONS ENVIRONMENT

The SaaS operator had already written wiring, tying together many aspects of its operations, including its continuous integration and deployment. This wiring was primarily comprised of Python scripts that had evolved as needed. The operator estimated that it had invested seven years of development into this operational wiring.



KEY COMPONENTS

- Puppet
  - Masterless Puppet was used in conjunction with Hiera
- ELK stack
- Sensu and other monitoring
- Application components
- YAML definitions of application requirements
- GitHub Enterprise

As the environment was put into product to deliver services to customers, several challenges emerged, including:

1. Keeping the automation updated with a rapidly changing application
2. Managing the automations themselves, including troubleshooting

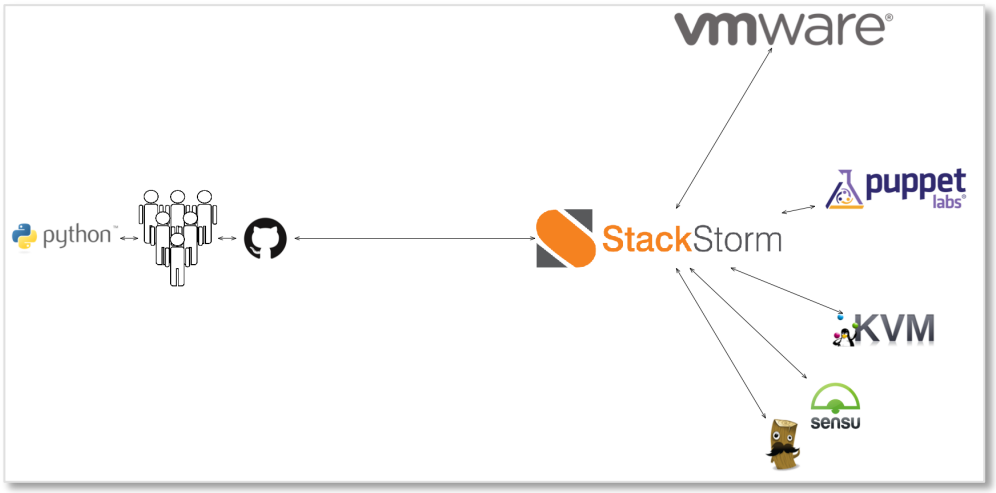
3. Scaling the automation layer globally across multiple data centers
4. Achieving requirements of security conscious partners, including large banks and telecommunications firms

The SaaS operator decided to focus its scarce resources on other matters and leverage StackStorm for operational wiring.

STACKSTORM DEPLOYMENT

The first task for the SaaS operator was to have StackStorm ingest the existing Python scripts into StackStorm. The Voodoo utility from StackStorm automatically added all the scripts to StackStorm's automation library and added relevant metadata to them. Once ingested, the scripts were available via the StackStorm CLI, which made them easier to manage and to operate thanks to help, parameter checking, logging and more. The flexibility of adding the actions programmatically (via the Voodoo utility), also allowed the SaaS operator to update their StackStorm action library automatically if any changes were made to the existing Python scripts. This minimized divergence between the currently deployed code base and StackStorm. Additionally, these scripts, now called actions, could be combined into pipelines via StackStorm's workflow engines and could be triggered either directly by operators, or via rules triggered by StackStorm sensors, or as a part of a broader workflow. The typically somewhat idiosyncratic scripts found new life as reusable building blocks.

StackStorm was deployed to run within the environment in a way that enabled StackStorm to both automate the CI/CD pipeline and itself fit into that pipeline.



Deployment Details and Considerations

As background, StackStorm itself leverages source code repositories, such as GitHub, to store all integrations and automations. This means that StackStorm complies with an approach that treats infrastructure as code, and can be rebuilt and restarted in an automated fashion.

The SaaS operator was using StackStorm to orchestrate an entire CI/CD process that included the addition of new application components, as well as enhancements to the existing components.

As new components were added, their specific deployment dependencies were recorded in a database that acted as the SaaS operator's "source of truth" for the environment. StackStorm accessed this source of truth for each deployment; associated fields were then passed as parameters to the automations executed by StackStorm. Certain automations were associated with each component by being stored in their repositories.

StackStorm is built to scale horizontally, via the use of a message bus, so that each component can be split into multiple workers run across different servers. As the operator's SaaS service scaled, this capability was critical in reducing downtime while also increasing throughput of the automations.

Conclusion

StackStorm software helped this SaaS operator achieve significant boosts in agility, while reducing the maintenance costs of their highly automated environment – all thanks to a unique ability to model and automate entire pipelines while treating each automation and integration as code that can be changed controlled.