DevOps, DevSecOps, and vArmour

DevOps and DevSecOps
Supporting DevSecOps with New Tools
vArmour DSS Distributed Security System
vArmour in 4 Common DevOps Scenarios
  1. Rolling out a New Application
  2. Promoting Applications from Dev/QA to Prod
  3. Expansion of Original Application
  4. Quarantine of Workloads
DevOps and DevSecOps

DevOps has become a well established practice within many organizations. It aims to improve the efficiency of the software development and delivery process through increased collaboration between teams, as well as automating the many stages of the process.

The increased collaboration and use of automation helps the teams move faster in delivery and results in several changes to typical application development, testing, release, operations, and maintenance. The increased pace of operations results in much more frequent releases, using a diverse set of tools at each stage of the process such as source code management, platform builds, infrastructure monitoring, and orchestration engines.

As DevOps introduces several changes to the application development workflows, infrastructure, and culture at an organization, traditional security practices start becoming impediments to achieving the desired goals. A security program practices, culture, and tooling built around the traditional software development model of relatively staggered, “waterfall” product releases and relatively immutable internal infrastructure will not be able to keep pace with the accelerated pace of DevOps changes, be they new applications, new code releases, or elastic infrastructure.

Enter DevSecOps. Similar to the mindset change that resulted in DevOps, DevSecOps is a rethinking of the role that security plays in the IT life-cycle. Specifically, the key tenets for security in a DevSecOps culture include elements such as:

- “Everyone is responsible for security”. The cooperation between Dev and Ops teams now includes Security as well. This means, for example, that security controls are built being keenly aware of specific application requirements, and that the infrastructure where the application runs will be highly elastic.

- “Security as Code”. This means that the contributions from the security organization be they compensating controls, security reviews, tactical information about threats, or security response processes have to be offered and deployed “as code”, to be integrated into the DevOps pipeline and final product.
Supporting DevSecOps with New Tools

As DevSecOps introduces a new approach to deploying security to a multitude of environments, there are new requirements for the security tooling used. Poor support for any of these DevOps requirements translates directly into poor outcomes in terms of security coverage, agility, or adoption of adequate security controls.

The key requirements are:

**Be designed for a modern infrastructure**

The modern data center or cloud infrastructure is not static: workloads are added and deleted programmatically, new code updates are pushed out multiple times per week or even per day, workloads move between hypervisors, and so on. A security design based on the traditional perimeter model will not be able to keep up with this level of dynamics: as an example, as workloads move between environments, a traditional security solution built on a perimeter mindset will have difficulty maintaining the state of connection.

**Be a software only architecture, not relying on hardware components**

In order to support the agility and elasticity of the modern environment, security controls have to be deployable as software, able to be quickly scaled up or down, or be reconfigured to support changes in the environment. A security architecture design that relies on hardware components will fare poorly on all these categories.

**Support orchestration via a robust and open API**

Lastly, it is critical that the tooling support easy configuration and orchestration via a robust API. The new application development and operations pipelines include significant amount of automation, orchestration, and use of standard design patterns. Any security component that cannot be completely deployed, configured, and managed programmatically will introduce unacceptable levels of friction to the DevOps cycle.
vArmour DSS Distributed Security System

IT leaders are transforming their data center architectures at unprecedented rates to harness the value of speed, moving from a reactive, inflexible organization to one that is a more proactive, agile part of the business that can respond quickly to changing business requirements. Research indicates that 86% of workloads will be processed by cloud data centers by 2019. Cloud services are based on simplicity, scale and economics. These same tenets should hold true for security.

With that in mind, vArmour created the vArmour DSS Distributed Security System. vArmour DSS provides application aware the micro-segmentation, coupled with advanced security analytics of networks, applications, and users. With its patented software, vArmour DSS moves security controls down next to each asset, wrapping fine grained protection around each workload, regardless of where the workload resides.

vArmour DSS is an all software solution, independent from underlying infrastructure, with distinct components that work as a unified security system. These components are:

**vArmour Fabric, composed of two types of elements:**
- **Director.** The Director is vArmour’s central point of orchestration (REST API) and management (via WebUI or CLI). It offers a highly available and scalable controller for the rest of the vArmour Fabric.
- **Enforcement Points.** vArmour Enforcement Points (“EP”s and “EPI”s) are distributed security processing nodes, offering L2-L7 visibility and access control to network traffic to/from the workloads. The Enforcement Points offer attractive performance, scalability, and availability characteristics that make them suitable for the elastic nature of virtualized and cloud workloads.

**vArmour Analytics.** vArmour Analytics provides application-layer visibility into east-west-type traffic flows, which in some cases constitutes 80% or more of the total traffic in virtualized data centers and cloud deployments. It also provides analytics/insight capabilities related to common patterns associated with more sophisticated attacks against data center / cloud resources.
vArmour can help DevOps build security awareness into all stages of application design and deployment with its key features:

- Being a **software-only component**, designed from the ground up to support modern environments, vArmour DSS has unmatched performance, scalability, and sizing characteristics than traditional perimeter approaches.

- By being **application-aware**, it can perform more granular inspection of traffic, which results in stronger protection and better definition of policy that can be aligned with development or operations requirements.

- Finally, vArmour has a **robust, REST based API** that allows for full automation and orchestration of all aspects of operations and management, from policy creation, software configuration, and ongoing monitoring.

For more information about vArmour DSS, please refer to [https://www.varmour.com/product/how-it-works](https://www.varmour.com/product/how-it-works)
vArmour DSS in 4 Common DevOps Scenarios

One of the key characteristics of ‘DevOps’ type deployments is that even though they share some broad design and operations patterns, each one is actually quite unique. Each organization that has embraced DevOps does it in a slightly different manner: similar but different workflows, deliverables in each stage, tooling being used, and others. The security tooling has to be flexible enough to meet these demands.

This section illustrates how vArmour DSS can fit into these patterns, without specific requirements on tooling. Customers may use whichever tools they best see fit for the DevOps workflow, and rely on vArmour’s open API for integrating the security policies provided by vArmour DSS.

1. Rolling Out a New Application
2. Promoting Applications from Dev/QA to Prod
3. Expansion of Original Application
4. Quarantine of Workloads
1. Rolling Out a New Application

In this scenario, as a new application is being built, the Dev team works alongside Security and Ops to define what the network security requirements will be. These requirements are then coded into the ‘playbook’ for the deployment. Depending on the tooling used, this can be an Ansible Playbook, a Puppet/Chef Recipe, or a Docker Dockerfile. In any case, the customer can use vArmour’s API to orchestrate that new policy into the vArmour Director.

**In the example below, a typical workflow might be:**

1. The Development and Operations teams define the application requirements into the tooling used for orchestration, maybe a Playbook or a Recipe.
2. The workload management component, such as Puppet, Chef, or Ansible, uses the vArmour REST API to create the appropriate security policy.
3. vArmour Director creates the policy and makes it available to the Enforcement Points on the environment.
4. (Not shown in diagram) As the workload manager creates the actual workloads, security policies are already available to be applied.
2. Promoting Applications from Dev/QA to Prod

There are many ways for a DevOps-focused organization to organize its processes for promoting workloads from dev/QA to production. One common approach is called the “Blue-Green Deployment”: two identical workloads exist, Blue and Green, and traffic is directed to the one defined ‘Prod’. After all work for the new release is done on the other environment, the traffic is redirected to the new environment. If there is an issue in the new environment, the workload can be easily redirected back.

vArmour DSS can easily support this by automating the application of security policies to workloads given their status (‘dev/qa’ versus ‘prod’). As the release management process instructs the external router to switch between Blue/Green, that information can be passed to vArmour as well, and Production security policies (such as more stringent access control rules) can easily be applied.
Sample workflow is as follows:

- At $t = 0$, the external router is pointing to ‘blue’ as the production system.
- At $t = 1$, release management instructs both external router and vArmour to treat ‘green’ as production.
- At $t = 2$, vArmour updates the security policies, be they broader zone-based segmentation or micro-segmentation policies on specific workloads.
- At that point, ‘production-level’ security controls are already applied to the chosen environment.

There’s several ways vArmour can be notified of the change in dev/prod state, including:

- **Changing tags on workloads.** As the organization chooses to promote a subset of the workload from ‘QA’ to ‘Prod’, all that it needs to do is tag the workload as such on the System of Record (CMDB, IPAM, VM tags, ...). vArmour DSS detects the change in information related to the workloads and automatically applies the new security policies.

- **Directly changing address group and/or information on vArmour.** The REST API includes full support for managing address groups and policies.
3. Expansion of Original Application

DevOps environments are generally elastic: new workloads are created and removed based on near real-time demand. This combines the efficacy of elastic infrastructures such as private, public, or hybrid clouds, with the efficiency of controlling operational costs.

In this scenario, an existing workload needs to be expanded (new instances added). As the new workload is instantiated, vArmour Director is able to read workload metadata and automatically assign the appropriate address group membership to new workloads. This can be done leveraging information from VM tags, IPAM systems, CMDB repositories, ticketing systems, or other sources of data.

In the example above:

1. The Resource Manager, typically an orchestration layer, detects the need for additional instances.
2. The Workload Manager, possibly a tool such as Puppet, Chef, or Ansible, instructs vArmour that additional workloads will be created, either directly (via REST API) or through leveraging the Source of Record system and the ongoing vArmour connection (option “2a” above).
3. The Workload manager creates the new workloads.
4. vArmour automatically applies policy to the workload as defined.
4. Quarantine of Workloads

One common design pattern on the operations side of DevSecOps is the use of quarantine - effectively blocking all communications to and from a workload. This is easily achievable using vArmour DSS.

The typical scenario here is that the Security Monitoring tooling or even the Application Performance Management component of the DevOps environment has determined that a particular workload should now be blocked off from the rest of the environment. This might be a response to a host or application-level security incident, or just that the software component on the application is not producing the right results and should not affect the rest of the environment.

At that point, any component can leverage the vArmour API to instruct vArmour DSS to effectively block all communications to/from that workload. This can be achieved in a number of ways, including:

- Adding the workload to a predefined ‘blacklist’ of quarantined workloads.
- Changing the workload information in Source of Record (IPAM, CMDB, ...) to indicate it should be quarantined.
Sample workflow is as follows:

1. Performance or Security Monitoring is monitoring individual workloads.

2. The need for a quarantine is detected. Two key approaches are possible:
   a. Tag the workload as “to be quarantined” in SoR, and vArmour can pick up the change.
   b. Directly instruct the Director to place a workload in quarantine via dedicated policy or
      address group membership.

3. vArmour applies policy to the workloads to enforce quarantine, keeping the workload in an
   operational state, but preventing further potential damage.
Conclusion
As a 100% software-based security solution, vArmour DSS enables the agility required by DevSecOps processes to build security into new workloads and their supporting applications, as well as ongoing policy updates, and quarantine capabilities. Learn more at www.varmour.com.

For more information about the broader tenets of Dev[Sec]Ops, you can refer to http://www.devsecops.org/.