TIGERA

How to stop fighting - and start aligning - organizational silos

Attaining Organizational Alignment in the Cloud Native Age

Functional silos are an essential and unavoidable aspect of large organizations. In theory at least, these silos are what have allowed disparate groups to carry out their distinct missions without the involvement of other groups.

The problem is that these silos also necessitate serial approaches to IT initiatives. That may have been okay in the past, when centrally planned, “waterfall”-type projects that took a year or more to deploy were the norm. However, the landscape has changed fundamentally. Today, projects and solutions need to be deployed every day, if not every hour or every minute. In this current paradigm, serialization fails. That’s why a new model is required, one that enables independent groups to fulfill their specific responsibilities and roles, while fostering rapid, agile and parallel activities across the entire organization.

Table of Contents

Introduction

Anyone who has tried to get things done in a large corporation has come into contact with the dreaded organizational silo. For those tasked with advancing larger projects or initiatives, these silos can feel like the enemy, a force to be avoided, bypassed or subverted. These silos bring to mind the famous quote from the Prussian field marshal Helmuth von Moltke the Elder, who could have been alluding to battles with silos when he said: “No plan of operations extends with certainty beyond the first encounter with the enemy's main strength.”

When silos are such an obstacle to progress, why do they still exist? Do they serve a purpose or are they simply a byproduct of corporate inertia? Actually, silos do serve a valid purpose. Corporations require functional separation, which, in turn, begets specific departments and teams. These groups can include operations, development, compliance, risk management, network security, data security and so on. All of these functions and organizations actually serve a legitimate need in an organization and do bring value.

In the days of waterfall approaches to project management, the process flow was inherently serialized. Each group or function would be put in a serial path, and perform their specific function when called upon, before passing the baton to the next function in line. This arrangement helped to eliminate conflict between different groups with distinct missions and objectives, and, at least on some level, ensured that all requirements were addressed.

The problem is that the modern organization can’t continue to tolerate the rigidity of a waterfall process. Organizations must be more agile and responsive to their customers’ and partners’ changing requirements. In fact, in this new environment, the old serial waterfall model can fail quite spectacularly. Across the board, organizations need to be more agile. In the case of software and solutions, this agility imperative is leading to the widespread adoption of such approaches as agile software development, site reliability engineering (SRE) and DevOps. This creates a critical point of friction: the silo model, where decisions are serialized and made in isolation from other functions, no longer works.What we need is a new way of allowing different functional groups to carry out their necessary tasks, but to do so in a way that is decoupled and independent of any other changes in the system, while still delivering the correct end result.

The good news is that the new cloud native model actually provides the building blocks necessary to accomplish this goal. Even better: Tigera has assembled those building blocks in a solution. As a result, organizations can begin to relegate silos to what they’re still good for, that is, storing grain and protecting missiles, and keep them from impeding their agile development efforts.

> *“ No solution deployment plan extends with any certainty beyond the first contact with a (corporate) silo.”*

What we need is a new way of allowing different functional groups to carry out their necessary tasks, but to do so in a way that is decoupled and independent of any other changes in the system, while still delivering the correct end result.

The good news is that the new cloud native model actually provides the building blocks necessary to accomplish this goal. Even better: Tigera has assembled those building blocks in a solution. As a result, organizations can begin to relegate silos to what they’re still good for, that is, storing grain and protecting missiles, and keep them from impeding their agile development efforts.

Defining Policy by Capturing Intent

Before we discuss how this might be accomplished, let’s take a quick recap of how policy and metadata drive the cloud native ecosystem, and how that policy is described. A full discussion of network policy is beyond the scope of this paper, but there are quite a few resources out there that go into much more depth, including this video[1] by one of our engineers.

In cloud native environments, such as Kubernetes, metadata drives much of the environment's functionality. Workloads, services, nodes and more are tagged with key/value pair labels. These labels identify what the thing is, what resources it needs, what services it offers, how it scales and so on. In these environments, an orchestration system makes its scheduling and orchestration decisions based, in part, on these metadata labels.

Through these metadata labels, organizational policies, including security policies, can be enforced. For example, in Kubernetes a security policy for running LDAP servers may appear as follows:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: ldapServer-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      ldap: server
  Ingress:
  - ports:
    - protocol: TCP
      port: 636
    from:
    - podSelector:
        matchLabels:
          ldap: client
```

What this policy describes is a certain *intent* of the author. It says that workloads that are labeled '`ldap: server`' should allow traffic on TCP port 636 (the LDAP SSL port) from workloads labeled `ldap: client`'. It does not specifically identify particular workloads or nodes that this

[1] https://www.projectcalico.org/videos/network-policy-in-kubernetes-with-project-calico

policy should apply to, nor how the policy should be enforced. For this reason, we call this an intent-based policy model.

This example describes an intent for how the network should behave in cases in which there are LDAP servers and LDAP clients instantiated in the cluster. It does not matter if there are one or a million of these elements: if any exist, this policy will be enforced. If there are no matching workloads, the policy will remain latent.

The benefit of this approach is that policy authors (whether developers, operations teams or compliance teams) do not need to know what IP addresses are being used by the workloads, or any other details about the implementation. They can use labels to state which systems can communicate with other systems, and over which protocols.

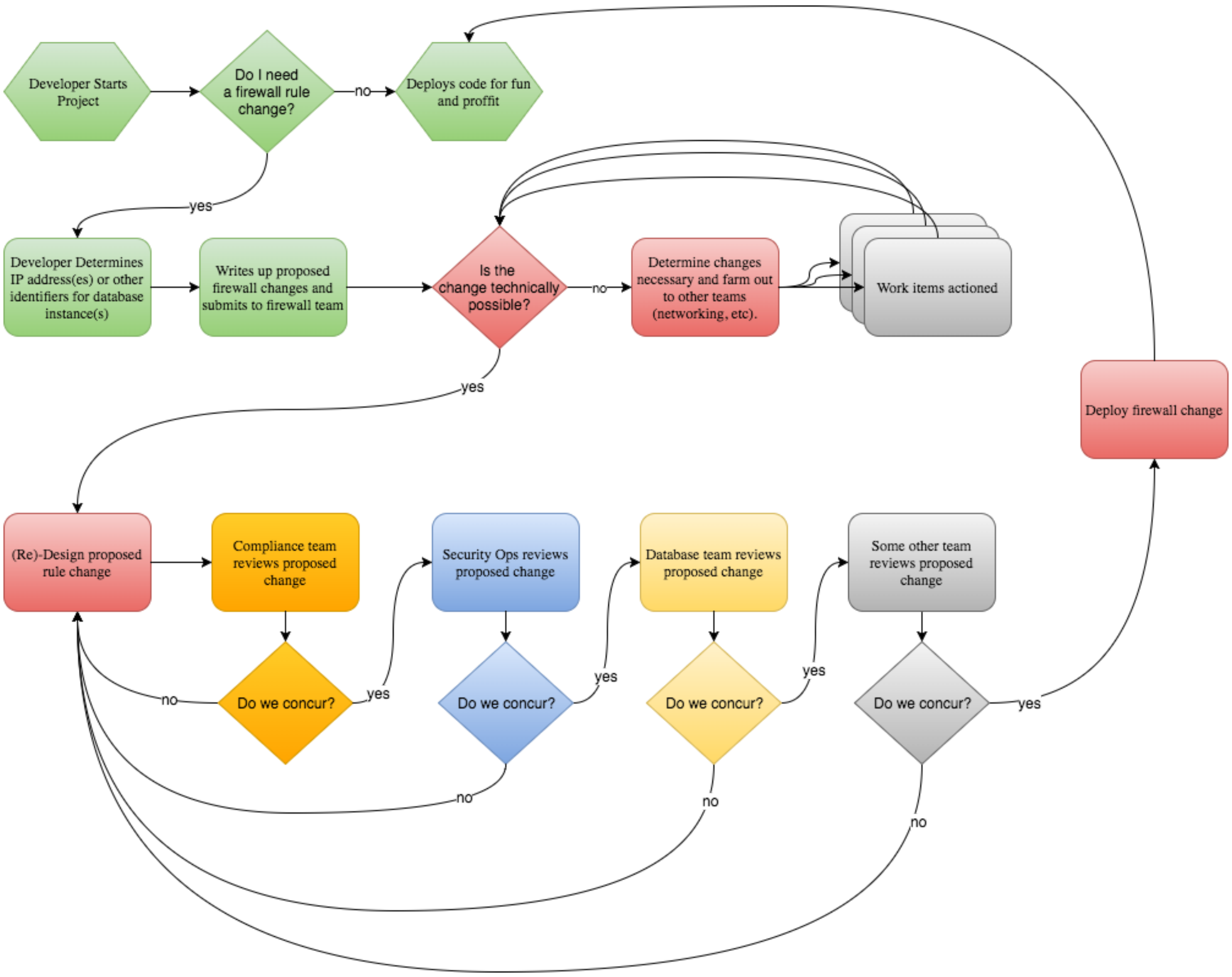
Organizational Requirements

This intent-based policy model is a significant advancement over IP-based firewall filtering, or isolating workloads by means of coarse-grained virtual LAN segments. However, this model doesn't really appear to address the topic of this paper: organizational alignment. No matter how the policies are expressed, an organization still needs to allow various functions to perform their activities. To understand how this is done today, let's look at how the silos currently enforce their requirements in a hypothetical organization and the effect this approach has within that organization.

> *" In many organizations this [traditional firewall configuration process] can take weeks. In a world where features need to be deployed in a rapid, agile manner, this level of latency is unacceptable."*

Current Silo-driven Workflow

Let’s say a developer needs a specific workload to access a database as part of a feature enhancement. That access is currently blocked by a firewall policy. The following diagram illustrates a common workflow for how this firewall change would be made.


Developer Starts Project
Do I need a firewall rule change?
no
Deploys code for fun and proffit
yes
Developer Determines IP address(es) or other identifiers for database instance(s)
Writes up proposed firewall changes and submits to firewall team
Is the change technically possible?
no
Determine changes necessary and farm out to other teams (networking, etc).
Work items actioned
yes
Deploy firewall change
(Re)-Design proposed rule change
Compliance team reviews proposed change
Security Ops reviews proposed change
Database team reviews proposed change
Some other team reviews proposed change
no
Do we concur?
yes
Do we concur?
yes
Do we concur?
yes
Do we concur?
yes
no
no
no

In a very efficient organization, this might take hours to a day or two. In many organizations, this process can take weeks. In a world where features need to be deployed in a rapid, agile manner, this level of latency is unacceptable.

> *“ The key to unlocking this paradox is to model the separate concerns in a hierarchical, intent-based policy model that works with, rather than against, the existing organizational structures.”*

A New Approach

What if the intent policy model we demonstrated above was extended to allow for different *tiers* of policy? Each tier would be assigned to a functional area. For example, the hierarchy could be as follows:

1. Operations Emergencies
2. Compliance
3. SRE Dev/Ops Management
4. Developer

Each tier could write the same kind of policies that were discussed above, but related to their functions. Those policies would be latent until/unless the metadata-driven conditions matched. Those policies could allow for matched traffic to be allowed, dropped, passed through to lower priority tiers, logged, and so on. In this example, the following rules would be placed:

1. Normally this would be empty, but in the case of an incident (such as a data center failure or DDoS attack), broadly scoped rules to block attack traffic, disallow low priority traffic, and more could be deployed.
2. Compliance could write the following PCI enforcement rules:

    a. If both ends of a flow are labeled `pciCompliant:true` then *fall through* to tier 3.

    b. If both ends of a flow are labeled `pciCompliant:false` then *fall through* to tier 3.

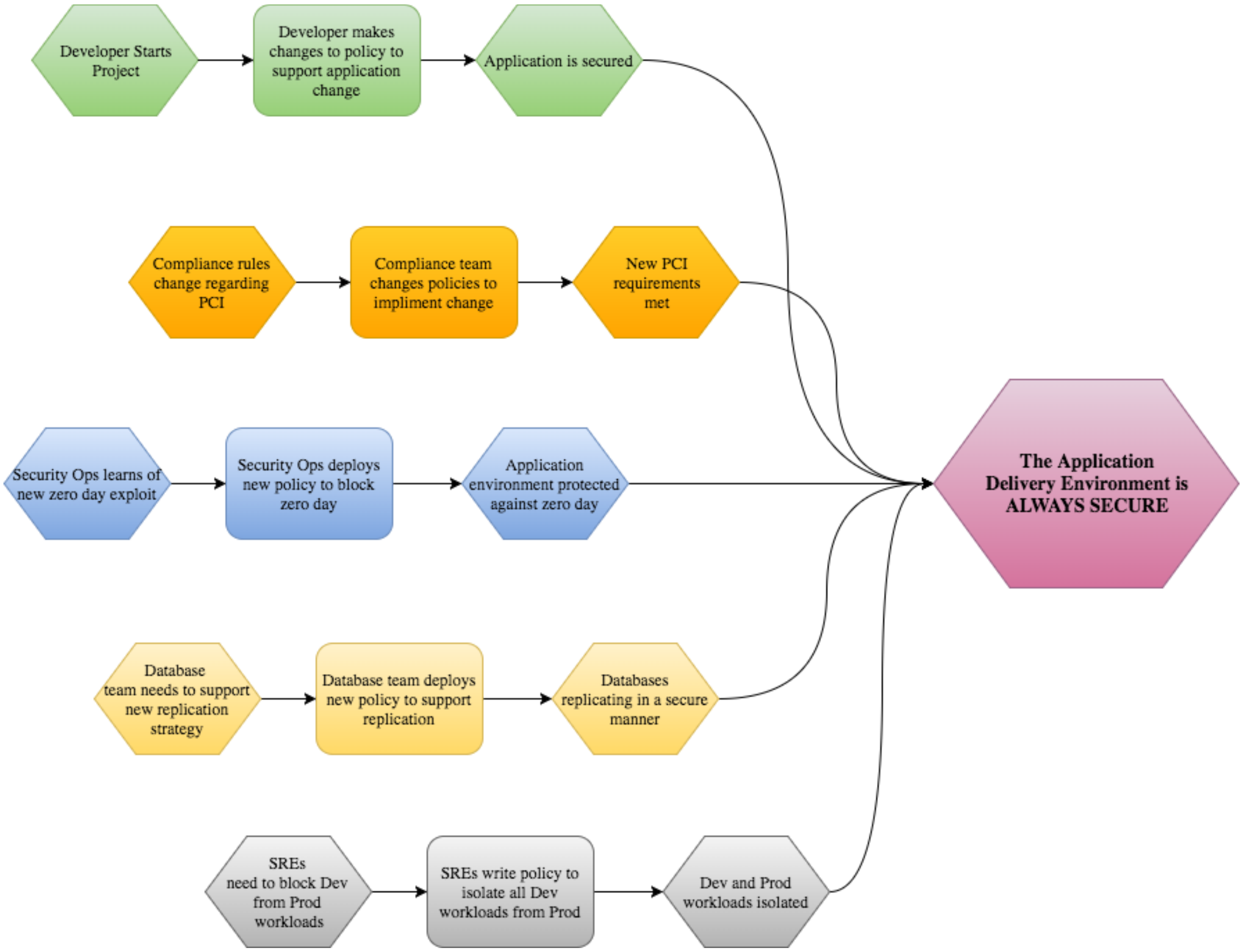
    c. If one end is labeled `pciCompliant:true` and the other is labeled `pciCompliant:false` then drop the packet and log the event.

3. The SRE or DevOps management team could write a policy similar to the PCI rules above, only blocking production workloads from communicating with test or development workloads.

4. The developer would write policies that described just how specific applications and workloads need to communicate — without needing to wait for approval from other organizational silos.

This can also be visualized in the flowchart included below.

The key benefit of such an approach is that members of each organizational silo can write their own rules independently of the others—parallelizing a previously serialized process. The result is that deployment times can be significantly reduced, without compromising security. In fact, the organization's overall security posture will most likely be enhanced.


Developer Starts Project
Developer makes changes to policy to support application change
Application is secured
Compliance rules change regarding PCI
Compliance team changes policies to impliment change
New PCI requirements met
Security Ops learns of new zero day exploit
Security Ops deploys new policy to block zero day
Application environment protected against zero day
The Application Delivery Environment is ALWAYS SECURE
Database team needs to support new replication strategy
Database team deploys new policy to support replication
Databases replicating in a secure manner
SREs need to block Dev from Prod workloads
SREs write policy to isolate all Dev workloads from Prod
Dev and Prod workloads isolated

Securing the Tiers

The reality is that establishing these tiers is only as effective as the controls governing who can change a tier's policies. As a result, access controls that match the roles within the organization must be applied to the policy model. In a Kubernetes environment, each policy object and each policy tier object should be tied to one or more users and/or groups in the Kubernetes *Role Based Access Control System* (RBAC). This system can be linked to many common organizational authentication systems. Through this approach, a compliance group can ensure that individuals in other groups can't change rules in the compliance tier, for example. This prevents anyone not in the *compliance* group from changing rules in the *compliance* policy tier.

Conclusion

The cloud native age is characterized by agile development and continuous deployment of applications. This cannot be achieved if every operational decision is serialized through multiple organizational silos. Yet these silos exist for a reason and can't simply be ignored. The key to unlocking this paradox is to model the separate concerns of different groups in a hierarchical, intent-based policy model that works with, rather than against, existing organizational structures.

The use of RBAC-secured policy tiers provides a new way for functional groups in an organization to continue to perform their necessary tasks and responsibilities, without inserting themselves in the application development and deployment chain. Through this approach, developers can make the development-level changes they need, without accidentally (or maliciously) being able to subvert higher-priority policies.

Further, owners of other policy tiers don't have to review each and every policy change. Each group can make changes at their own pace, in a way that's decoupled from other groups. By combining asynchronous, decoupled design and secure, tiered policy enforcement, this approach enables organizations to establish alignment across functions—while allowing agile, even continuous, application development and deployment. Quite simply, staff can start working with, rather than against, organizational silos.

About Tigera

Tigera delivers solutions for secure application connectivity for the cloud native world. Tigera technology is used by the world's largest enterprises and public cloud providers to power connectivity for application development and deployment and to address the connectivity and security challenges that arise in at-scale production. Tigera Secure meets enterprise needs for zero trust network security, multi-cloud and legacy environment support, organizational control and compliance, and operational simplicity. Secure builds on leading open source projects Kubernetes, Calico, and Istio, which Tigera engineers help maintain and contribute to as active members of the cloud native community.

tigera.io

email: contact@tigera.io

phone: +1.415.612.9546

Tigera, Inc. 58 Maiden Lane, Fifth Floor, San Francisco CA 94018 USA