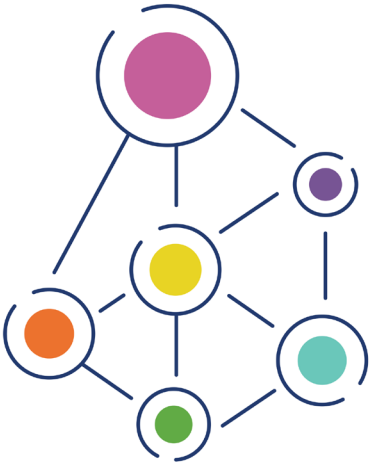
Aporeto

Cloud Security Gaps

Aporeto

Why Network Segmentation is Failing Your Cloud Application Security

How to Achieve Effective Application Segmentation



By now it's obvious to security-minded developers and IT professionals that as applications move to public cloud, the definition of the "perimeter" changes. You can no longer rely on network perimeters to manage access, prevent attacks, and control traffic since you no longer manage the network. In fact, you can only trust the network in the public cloud to the extent that you understand it and have predictable, efficient ways of managing it. Because the network perimeter essentially no longer exists, or at the very least is expanded to include the virtual boundaries of your cloud environment, your security models start to break down. Most organizations today are either just establishing teams and expertise to focus on the nuances and rapid evolution of cloud security, or in far too many cases they are only beginning to understand that things are different.

When organizations move application workloads to public cloud environments or build greenfield applications that are designed for the cloud from the beginning, there a number of security assumptions in the legacy, datacenter-centric security approach that tend to fail. In this paper, we will focus on the concept of segmentation as a security counter measure, and how old approaches are not translating successfully to cloud environments. Finally, we will offer an alternative of workload identity as a new control point for security applications in the cloud.

Limits of Network Segmentation

One of the most widely deployed security practices is "segmentation", or the process of separating end-points in different trust domains and controlling interactions between those domains through policy rules. Segmentation manages information flow between domains and therefore reduces the attack surface between them. In essence, it limits the "blast radius," or the portion of application that can be directly affected if an attacker manages to penetrate one of the trust domains. Segmentation is also commonly sought to reduce compliance scope, as it's a recognized method for easing PCI compliance.

Segmentation was initially based on IP subnets and VLANs. Operations assumed a static association between services and servers (or IP addresses) and by placing servers in different VLANs, administrators could enforce isolation between services. Firewalls were often deployed to enforce policy rules between VLANs.

Virtualization broke some of the basic assumptions of these implementations since it removed the static association of IP addresses to services. A new a set of proprietary and standard solutions (see IEEE VEB and VEPA) were developed to automate the mapping process of VM end-points to VLANs; firewalls struggled to keep up.

Security Groups and their Pitfalls

The cloud has further challenged the static assumptions of VLANs while imposing larger scaling requirements. Spanning VLANs across multiple racks quickly became the bottleneck. Simultaneously, concepts like security groups that AWS introduced were easy for developers to understand and expanded the segmentation model to a more granular level, where any set of hosts irrespective of IP subnets could be grouped together in different trust domains.

There is certainly value in the network isolation provided by AWS security groups and the analog provided by other public cloud providers, but there are also numerous pitfalls from trying to manage segmentation this way.

1. The first issue arises from the fact that very few organizations rely on a single public cloud provider for all infrastructure, requiring them to replicate security policy and network configuration across on-premise and cloud deployments, as well as across several public cloud providers. Many security teams are struggling with just defining consistent security policy across all of these environments, much less effectively implementing them.

2. Security groups are designed to allow you to enact firewall policies withing cloud instances, which reproduces many of the network segmentation challenges described in the previous section. When you try to do unnatural things with network configuration, it does not matter if you're doing it on-premise or in the cloud with security groups. There are also practical limits on the number of groups and flexibility of the segmentation design that can be modeled in this manner. Managing network configurations and ACLs is challenging enough when you are running the network, but can be even more challenging at scale when you're trying to model those network policies with a level of cloud abstraction.

3. Because of the nature of shared infrastructure in public cloud security groups create some unique challenges in addition to those inherent in network segmentation at scale. Elastically scaling and temporal workloads create a dynamism in IP address configuration that is difficult to manage as a control point for security policy. When workloads are replicated or re-scheduled across hosts, as is very typical in a scaled-out cloud environment, the security groups configuration does not handle it well. VPCs can also have overlapping addresses across instances and regions, so the granularity of security policy ends up compromising based on the IP address space and allocation schemes. Network access policies can also become problematic with NAT gateways or load balancers are in play.

All of these factors together make a relatively simple to implement and powerful firewalling feature of cloud computing ill-suited for effectively segmenting and securing modern cloud-native workloads. The next attempt that some security teams make is to miniaturize and virtualize their on-premise security controls and run them as instances in the public cloud to attempt to replicate their on-premise topology and policy. This results in a compounding mess of expensive virtualized security controls costing you additional license and compute costs, while not effectively segmenting your workloads any more than they are on-premise.

The Promise of Micro-Segmentation

As a response to the cloud, the networking industry reinvented itself once more and borrowed concepts from VPNs by moving isolation to tunnels and creating micro-segmentation. GRE, VXLAN and Geneve were invented to create virtual VLAN-type structures that could span multiple racks within or or across data centers without the routing and management complexities of L2 technologies. SDN controllers, protocols (VTEP, OVSDB, etc.), and software gateways became necessary as the requirements of segmentation solutions increased. The term "micro-segmentation" was coined to show an ever-larger number of segments within a larger network.

If we take a step back, we can quickly see that all of these segmentation solutions are based on the same concept. They start with a fundamental assumption that applications, services (or end-points) are identified by their IP addresses and potentially port numbers. Then, they introduce one level of indirection by mapping sets of end-points to an identifier (VLAN, VXLAN ID, MPLS label, etc.) and they make policy decisions based on the identifier. Finally, they use a control plane to disseminate state about these associations and corresponding policy rules. In virtualization and cloud worlds, mapping of end-points to identifiers is programmed through orchestration tools. Secondary control planes optimize state distribution information for each virtual network.

The adoption of containers, micro-services, and serverless architectures once more challenges the assumptions behind these solutions. Applications are disaggregated to smaller and often ephemeral components that are launched based on events. Segmentation needs to account for the dynamic nature of these applications; troubleshooting tunnels and control planes is becoming increasingly complex. The right tools are still evolving, and the quick convergence of a network is an ever-larger problem.

A new set of solutions (still at their infancy) appear to be challenging network virtualization benefits by removing the need for network-level segmentation. These solutions assume a simple, flat network, where every process or workload is still uniquely identified through its IP address and port numbers. They attempt to solve the segmentation problem by distributing ACLs to every host. The concept is quite simple: If there are two domains of workloads (call them A and B), the policy problem can be solved by installing an ACL on every server that hosts workload A with a rule that allows communication between A and B. Assuming a control and management plane that allows identification of workloads and distribution of state, segmentation problems are pretty much "solved."

But, flattening the networks comes with its own penalties. Let us consider a simple scenario where workloads A and B are composed of 20 containers each and placed in 40 different hosts. A flat topology requires creating and maintaining 400 ACL rules. Each server that hosts a container of type A must have an ACL to allow connectivity to any of the containers of type B (20 per server). Now, if we are to consider a data center with 1000 workloads each with 20 containers, we quickly realize that there is the management nightmare of hundreds of thousands of ACLs.

There are two key issues at this juncture:

1. First, there is a question of the convergence. Even if we are successful at distributing ACLs every time a workload is activated, when a new container is added to a target set, the control plane would need to update the ACL rules on all servers that host related containers. In application environments with ephemeral micro-services or event-based mechanisms (see AWS Lambda), where a process (container) is only instantiated to complete a specific task and terminates afterwards, the control plane will have to sustain an activation rate of several thousand ACLs per second.

2. Second, on the performance front, introducing 1,000s of ACLs on every host has its own challenges since ACL lookup algorithms are essentially point location problems in a multi-dimensional space with significant per-packet complexity. Current Linux systems with ipsets give the illusion of simplicity with a space-time trade off, but still require several CPU cycles for every packet.

Continuous Deployment (CD) with a Break for Security/Compliance

The business value of microservices lies in improving deployment velocity for applications allowing companies to innovate at a higher rate. Key to improving deployment velocity is an automated continuous deployment (CD) pipeline. One of the biggest hurdles to achieving this automation is the antiquated ticket based nature of ensuring security rules (IP address and port) are defined appropriately before deploying an application in production. This hurdle exists because security teams are the gate keeper – as they should be – to ensure proper security and compliance is in place. This manual nature of managing and implementing policies does not scale and requires re-thinking for this microservice and cloud era .

So, one has to ask the question: If network-based segmentation (VLANs, network virtualization, etc.) creates operational complexity, if flat networks with ACLs cannot scale and if deployment velocity is hampered because of manual intervention from security teams are there any alternatives?

Application Identity-Based Segmentation

As infrastructure has evolved and the cloud has dramatically transformed IT, our attention must turn to the application workloads themselves. We simply cannot rely on the infrastructure or the network to provide effective security in context with application workloads, because by their nature they are increasingly shared resources that do not understand application context. Instead, we have been pursuing networking solutions for years to solve a trust problem between domains instead of focusing on understanding the problem itself. The key "requirement" is to control information flow between the components (services) that form an application. Services need to be identified by their characteristics and not by their IP addresses that should only be used as location and routing identifiers. Policies need to control information exchange between services and not between IP addresses. We essentially need to solve an authentication and authorization problem and not a packet ACL problem. The mechanisms used to forward packets and organize the network for operational reasons do not need to be coupled with security policies; the two must evolve independently.

Aporeto introduces a multi-attribute, contextual application identity as a new security paradigm for applying distributed security policy to distributed systems. A unique workload identity allows security teams to define network access control policies independent of network Infrastructure and allows them to do so in a declarative manner that is part of the automated continuous deployment pipeline. This has substantial implications to security teams when deploying dynamic microservices, enabling a "Secure once, Run Anywhere" security policy. Security is tied to your workload, in context with what it is, where it runs, how it behaves and how risky it is, rather than the increasingly unreliable IP address. Only then can you effectively segment workloads and securely build and run applications in the Zero Trust landscape of the public cloud. The ultimate segmentation is application segmentation based on application identity, and only Aporeto can do this.

For more information, visit:
www.aporeto.com