Making Middleboxes Someone Else's Problem: Network Processing as a Cloud Service

Justine Sherry
UC Berkeley

Shaddi Hasan
UC Berkeley

Colin Scott
UC Berkeley

Arvind Krishnamurthy
University of Washington

Sylvia Ratnasamy
UC Berkeley

Vyas Sekar
Intel Labs

ABSTRACT

Modern enterprises almost ubiquitously deploy middlebox processing services to improve security and performance in their networks. Despite this, we find that today's middlebox infrastructure is expensive, complex to manage, and creates new failure modes for the networks that use them. Given the promise of cloud computing to decrease costs, ease management, and provide elasticity and fault-tolerance, we argue that middlebox processing can benefit from outsourcing the cloud. Arriving at a feasible implementation, however, is challenging due to the need to achieve functional equivalence with traditional middlebox deployments without sacrificing performance or increasing network complexity.

In this paper, we motivate, design, and implement APLOMB, a practical service for outsourcing enterprise middlebox processing to the cloud. Our discussion of APLOMB is data-driven, guided by a survey of 57 enterprise networks, the first large-scale academic study of middlebox deployment. We show that APLOMB solves real problems faced by network administrators, can outsource over 90% of middlebox hardware in a typical large enterprise network, and, in a case study of a real enterprise, imposes an average latency penalty of 1.1ms and median bandwidth inflation of 3.8%.

Categories and Subject Descriptors
C.2.0 [**Computer-Communication Networks**]: General—*Security and firewalls*; C.2.1 [**Network Architecture and Design**]: [Distributed applications]; C.2.3 [**Network Operations**]: [Network management]

General Terms
Design, Management, Measurement

Keywords
Middlebox, Cloud, Outsourcing

1. INTRODUCTION

Today's enterprise networks rely on a wide spectrum of specialized appliances or *middleboxes*. Trends such as the proliferation of smartphones and wireless video are set to further expand the range of middlebox applications [19]. Middleboxes offer valuable benefits, such as improved security (*e.g.*, firewalls and intrusion detection systems), improved performance (*e.g.*, proxies) and reduced bandwidth costs (*e.g.*, WAN optimizers). However, as we show in §2, middleboxes come with high infrastructure and management costs, which result from their complex and specialized processing, variations in management tools across devices and vendors, and the need to consider policy interactions between these appliance and other network infrastructure.

The above shortcomings mirror the concerns that motivated enterprises to transition their in-house IT infrastructures to managed cloud services. Inspired by this trend, we ask whether the promised benefits of cloud computing—reduced expenditure for infrastructure, personnel and management, pay-by-use, the flexibility to try new services without sunk costs, *etc.*—can be brought to middlebox infrastructure. Beyond improving the status quo, cloud-based middlebox services would also make the security and performance benefits of middleboxes available to users such as small businesses and home and mobile users who cannot otherwise afford the associated costs and complexity.

We envision enterprises outsourcing the processing of their traffic to third-party *middlebox service providers* running in the cloud. Our proposal represents a significant change to enterprise networks, and hence we first validate that this exercise is worthwhile by examining what kind of a burden middleboxes impose on enterprises. The research literature, however, offers surprisingly few real-world studies; the closest study presents anecdotal evidence from a single large enterprise [42]. We thus start with a study of 57 enterprise networks, aimed at understanding (1) the nature of real-world middlebox deployments (*e.g.*, types and numbers of middleboxes), (2) "pain points" for network administrators, and (3) failure modes. Our study reveals that middleboxes do impose significant infrastructure and management overhead across a spectrum of enterprise networks and that the typical number of middleboxes in an enterprise is comparable to its traditional L2/L3 infrastructure!

Our study establishes the costs associated with middlebox deployments and the potential benefits of outsourcing them. We then examine different options for architecting cloud-based middlebox services. To be viable, such an architecture must meet three challenges:

(1) Functional equivalence. A cloud-based middlebox must offer functionality and semantics equivalent to that of an on-site middlebox – *i.e.*, a firewall must drop packets correctly, an intrusion detection system (IDS) must trigger identical alarms, *etc.* In contrast to traditional endpoint applications, this is challenging because middlebox functionality may be topology dependent. For example, traffic compression must be implemented *before* traffic leaves the enterprise access link, and an IDS that requires stateful processing must see *all* packets in *both* directions of a flow. Today, these requirements are met by deliberately placing middleboxes 'on path' at network choke points within the enterprise – options that are not readily available in a cloud-based architecture. As we shall see, these topological constraints complicate our ability to outsource middlebox processing.

SIGCOMM'12, August 13–17, 2012, Helsinki, Finland.

(2) Low complexity at the enterprise. As we shall see, an outsourced middlebox architecture still requires some supporting functionality at the enterprise. We aim for a cloud-based middlebox architecture that minimizes the complexity of this enterprise-side functionality: failing to do so would detract from our motivation for outsourcing in the first place.
(3) Low performance overhead. Middleboxes today are located on the direct path between two communicating endpoints. Under our proposed architecture, traffic is instead sent on a detour through the cloud leading to a potential increase in packet latency and bandwidth consumption. We aim for system designs that minimize this performance penalty.

We explore points in a design space defined by three dimensions: the redirection options available to enterprises, the footprint of the cloud provider, and the complexity of the outsourcing mechanism. We find that all options have natural tradeoffs across the above requirements and settle on a design that we argue is the sweet spot in this design space, which we term APLOMB, the Appliance for Outsourcing Middleboxes. We implement APLOMB and evaluate our system on EC2 using real end-user traffic and an analysis of traffic traces from a large enterprise network. In our enterprise evaluation, APLOMB imposes an average latency increase of only 1 ms and a median bandwidth inflation of 3.8%.

To summarize, our key contributions are:

- A study of costs and concerns in 57 real-world middlebox deployments, across a range of enterprise scenarios.
- A systematic exploration of the requirements and design space for outsourcing middleboxes.
- The design, implementation, and evaluation of the APLOMB architecture.
- A case study of how our system would impact the middlebox deployment of a large enterprise.

A core question in network design is where network functionality should be embedded. A wealth of research has explored this question for various network functionality, such as endpoints *vs.* routers for congestion control [20, 35, 29] and on-path routers *vs.* off-path controllers for routing control plane functions [28, 39]. Our work follows in this vein: the functionality we focus on is advanced traffic processing (an increasingly important piece of the network *data* plane) and we weigh the relative benefits of embedding such processing in the cloud *vs.* on-path middleboxes, under the conjecture that the advent of cloud computing offers new, perhaps better, options for supporting middlebox functionality.

Roadmap: We present our study of enterprise middlebox deployments in §2. In §3 we explore the design space for outsourcing middleboxes; we present the design and evaluation of the APLOMB architecture in §4 and §5 respectively. We discuss outstanding issues in §6 and related work in §7 before concluding in §8.

2. MIDDLEBOXES TODAY

Before discussing outsourcing designs, we draw on two datasets to discuss typical middlebox deployments in enterprise networks and why their challenges might be solved by the cloud. We conducted a survey of 57 enterprise network administrators, including the number of middleboxes deployed, personnel dedicated to them, and challenges faced in administering them. To the best of our knowledge, this is the first large-scale survey of middlebox deployments in the research community. Our dataset includes 19 small (fewer than 1k hosts) networks, 18 medium (1k-10k hosts) networks, 11 large (10k-100k hosts) networks, and 7 very large (more than 100k hosts) networks.

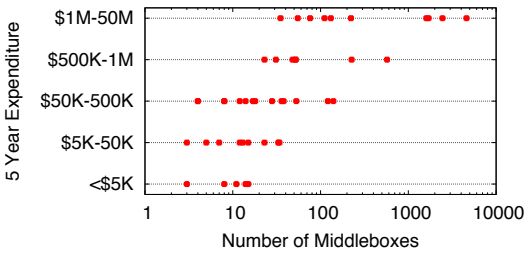


Figure 2: Administrator-estimated spending on middlebox hardware per network.

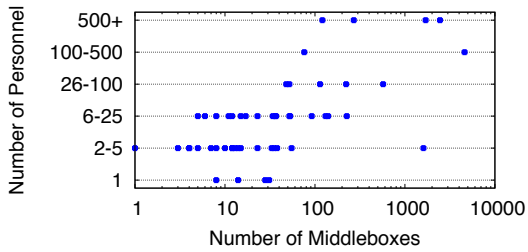


Figure 3: Administrator-estimated number of personnel per network.

We augment our analysis with measurements from a large enterprise with approximately 600 middleboxes and tens of international sites; we elaborate on this dataset in §5.3.

Our analysis highlights several key challenges that enterprise administrators face with middlebox deployments: large deployments with high capital expenses and operating costs (§2.1), complex management requirements (§2.2), and the need for overprovisioning to react to failure and overload scenarios (§2.3). We argue these factors parallel common arguments for cloud computation, and thus make middleboxes good candidates for the cloud.

2.1 Middlebox Deployments

Our data illustrates that typical enterprise networks are a complex ecosystem of firewalls, IDSes, web proxies, and other devices. Figure 1 shows a box plot of the number of middleboxes deployed in networks of all sizes, as well as the number of routers and switches for comparison. Across all network sizes, the number of middleboxes is on par with the number of routers in a network! The average very large network in our data set hosts 2850 L3 routers, and 1946 total middleboxes; the average small network in our data set hosts 7.3 L3 routers and 10.2 total middleboxes.[1]

These deployments are not only large, but are also costly, requiring high up-front investment in hardware: thousands to millions of dollars in physical equipment. Figure 2 displays five year expenditures on middlebox hardware against the number of actively deployed middleboxes in the network. All of our surveyed very large networks had spent over a million dollars on middlebox hardware in the last five years; the median small network spent between $5,000-50,000 dollars, and the top third of the small networks spent over $50,000.

Paralleling arguments for cloud computing, outsourcing middlebox processing can reduce hardware costs: outsourcing eliminates most of the infrastructure at the enterprise, and a cloud provider can provide the same resources at lower cost due to economies of scale.

[1]Even 7.3 routers and 10.2 middleboxes represents a network of a substantial size. Our data was primarily surveyed from the NANOG network operators group, and thus does not include many of the very smallest networks (*e.g.* homes and very small businesses with only tens of hosts).

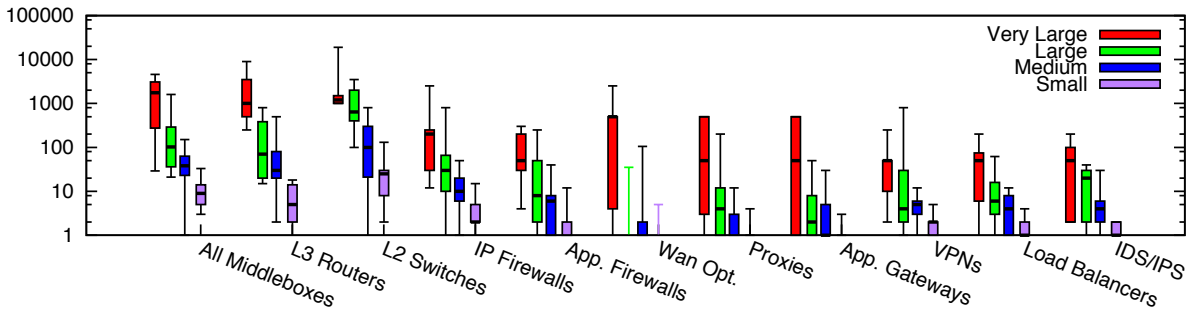


Figure 1: Box plot of middlebox deployments for small (fewer than 1k hosts), medium (1k-10k hosts), large (10k-100k hosts), and very large (more than 100k hosts) enterprise networks. Y-axis is in log scale.

2.2 Complexity in Management

Figure 1 also shows that middleboxes deployments are diverse. Of the eight middlebox categories we present in Figure 1, the median very large network deployed seven categories of middleboxes, and the median small network deployed middleboxes from four. Our categories are coarse-grained (*e.g.* Application Gateways include smartphone proxies and VoIP gateways), so these figures represent a *lower bound* on the number of distinct device types in the network.

Managing many heterogeneous devices requires broad expertise and consequently a large management team. Figure 3 correlates the number of middleboxes against the number of networking personnel. Even small networks with only tens of middleboxes typically required a management team of 6-25 personnel. Thus, middlebox deployments incur substantial operational expenses in addition to hardware costs.

Understanding the administrative tasks involved further illuminates why large administrative staffs are needed. We break down the management tasks related to middleboxes below.

Upgrades and Vendor Interaction. Deploying new features in the network entails deploying new hardware infrastructure. From our survey, network operators upgrade in the median case every four years. Each time they negotiate a new deployment, they must select between several offerings, weighing the capabilities of devices offered by numerous vendors – an average network in our dataset contracted with 4.9 vendors. This four-year cycle is at the same time both too frequent and too infrequent. Upgrades are too frequent in that every four years, administrators must evaluate, select, purchase, install, and train to maintain new appliances. Upgrades are too infrequent in that administrators are 'locked in' to hardware upgrades to obtain new features. Quoting one administrator:

> Upgradability is very important to me. I do not like it when vendors force me to buy new equipment when a software upgrade could give me additional features.

Cloud computing eliminates the upgrade problem: enterprises sign up for a middlebox *service*; how the cloud provider chooses to upgrade hardware is orthogonal to the service offered.

Monitoring and Diagnostics. To make managing tens or hundreds of devices feasible, enterprises deploy network management tools (e.g., [15, 9]) to aggregate exported monitoring data, *e.g.* SNMP. However, with a cloud solution, the cloud provider monitors utilization and failures of specific devices, and only exposes a middlebox *service* to the enterprise administrators, simplifying management at the enterprise.

Configuration. Configuring middleboxes requires two tasks. *Appliance configuration* includes, for example, allocating IP addresses, installing upgrades, and configuring caches. *Policy configuration* is customizing the device to enforce specific enterprise-wide policy goals (*e.g.* a HTTP application filter may block social network sites). Cloud-based deployments obviate the need for enterprise administrators to focus on the low-level mechanisms for appliance configuration and focus only on policy configuration.

	Misconfig.	Overload	Physical/Electric
Firewalls	67.3%	16.3%	16.3%
Proxies	63.2%	15.7%	21.1%
IDS	54.5%	11.4%	34%

Table 1: Fraction of network administrators who estimated misconfiguration, overload, or physical/electrical failure as the most common cause of middlebox failure.

Training. New appliances require new training for administrators to manage them. One administrator even stated that existing training and expertise was a key question in purchasing decisions:

> Do we have the expertise necessary to use the product, or would we have to invest significant resources to use it?

Another administrator reports that a lack of training limits the benefits from use of middleboxes:

> They [middleboxes] could provide more benefit if there was better management, and allocation of training and lab resources for network devices.

Outsourcing diminishes the training problem by offloading many administrative tasks to the cloud provider, reducing the set of tasks an administrator must be able perform. In summary, for each management task, outsourcing eliminates or greatly simplifies management complexity.

2.3 Overload and Failures

Most administrators who described their role as engineering estimated spending between one and five hours per week dealing with middlebox failures; 9% spent between six and ten hours per week. Table 1 shows the fraction of network administrators who labeled misconfiguration, overload, and physical/electrical failures as the most common cause of failures in their deployments of three types of middleboxes. Note that this table is *not* the fraction of failures caused by these issues; it is the fraction of administrators who estimate each issue to be the *most common* cause of failure. A majority of administrators stated misconfiguration as the most common cause of failure; in the previous subsection we highlight management complexity which likely contributes to this figure.

On the other hand, many administrators saw overload and physical/electrical problems as the most common causes of errors. For example, roughly 16% of administrators said that overload was the most common cause of IDS and proxy failure, and 20% said that physical failures were the most common cause for proxies.

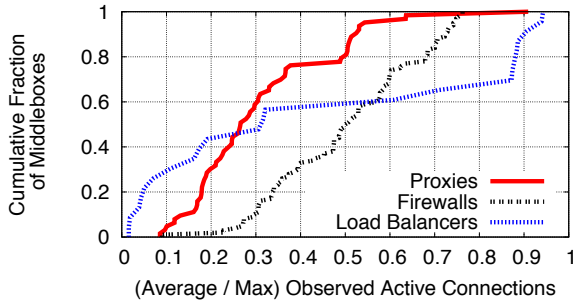


Figure 4: Ratio of average to peak active connections for all proxies, firewalls, and load balancers in the very large enterprise dataset.

A cloud-based capability to elastically provision resources avoids overload by enabling on-demand scaling and resolves failure with standby devices – without the need for expensive overprovisioning.

2.4 Discussion

To recap, our survey across 57 enterprises illuminates several middlebox-specific challenges that cloud outsourcing can solve: large deployments with high capital and operating expenses, complex management requirements inflating operation expenses, and failures from physical infrastructure and overload. Cloud outsourcing can cut costs by leveraging economies of scale, simplify management for enterprise administrators, and can provide elastic scaling to limit failures.

Outsourcing to the cloud not only solves challenges in existing deployments, but also presents new opportunities. For example, resource elasticity not only allows usage to scale *up*, but also to scale *down*. Figure 4 shows the distribution of average-to-max utilization (in terms of active connections) for three devices across one large enterprise. We see that most devices operate at moderate to low utilization; e.g., 20% of Load Balancers run at <5% utilization. Today, however, enterprises must invest resources for peak utilization. With a cloud solution, an enterprise can lease a large load balancer only at peak hours and a smaller, cheaper instance otherwise. Furthermore, a pay-per-use model democratizes access to middlebox services and enables even small networks who cannot afford up-front costs to benefit from middlebox processing.

These arguments parallel familiar arguments for the move to cloud computation [23]. This parallel, we believe, only bolsters the case.

3. DESIGN SPACE

Having established the potential benefits of outsourcing middleboxes to the cloud, we now consider how this might be achieved. To understand the challenges involved in such outsourcing, it is useful to reflect on how middleboxes are deployed today within an enterprise. Consider a middlebox m that serves traffic between endpoints a and b. Our proposal changes the *placement* of m – moving m from the enterprise to the cloud. This eliminates three key properties of today's middlebox placement:

1. *on-path*: m lies on the direct IP path between a and b
2. *choke point*: all paths between a and b traverse m
3. *local*: m is located inside the enterprise.

The challenges in outsourcing middleboxes arise as a result of losing these three properties. First, being on-path and at a choke point makes it easy for a middlebox to obtain the traffic it must process and specifically ensures that the middlebox sees both directions of traffic flow between two endpoints. (Bidirectional visibility is critical since most middleboxes operate at the session level.) Second, being on-path implies that a middlebox introduces no additional latency into the path. In contrast, sending traffic on a detour through the cloud could increase path latency. Third, middleboxes such as proxies and WAN optimizers, which we dub *location dependent* boxes, rely on being physically local to reduce latency and bandwidth costs. For example, proxies effectively terminate communication from an enterprise host a to an external host b thus reducing communication latency from that of path a-m-b to that of a-m. Similarly, by using redundancy elimination techniques, WAN optimizers avoid transmitting data over the wide area.

This raises three natural questions that together define the overall design space for outsourcing middleboxes to the cloud:

1. What is the effective *complexity* of the network architecture at the enterprise after outsourcing – *e.g.*, what types of middleboxes can be outsourced and what enterprise-side functionality is needed to achieve such outsourcing?
2. What *redirection* architecture is required to retain the functional equivalence and low latency operation; *e.g.*, ensuring that both directions of traffic between a and b via the cloud consistently traverse the same cloud PoP?
3. What type of *provider footprint* is needed for low latency operation; *e.g.*, is an Amazon-style footprint with a few distributed PoPs sufficient or do we need a larger, Akamai-scale footprint?

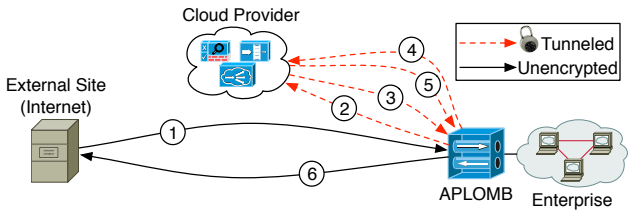
At a minimum, we need some generic device to redirect the enterprise's traffic to the cloud; we call this an *Appliance for Outsourcing Middleboxes* or *APLOMB*. We explore options for redirection in §3.1 and discuss strategies for low latency operation and evaluate the impact of the provider footprint in §3.2. In addition, to retain the savings in bandwidth consumption that local proxies and WAN optimizers offer, we consider extending APLOMB to have compression capabilities (which we call APLOMB+) in §3.3.

3.1 Redirection

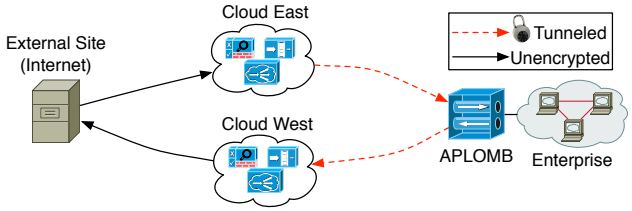
We consider three natural approaches for redirecting the traffic to the cloud for middlebox processing and analyze their latency vs. complexity tradeoffs.

Bounce Redirection: In the simplest case, the APLOMB gateway at the enterprise tunnels both ingress and egress traffic to the cloud, as shown in Figure 5(a). Incoming traffic is bounced to the cloud PoP (1), processed by middleboxes, and then sent back to the enterprise (2,3). Outgoing traffic is similarly redirected (4-6). An immediate drawback of this architecture is the increase in end-to-end latency due to an extra round trip to and from the cloud PoP for each packet. Nevertheless, this design is feasible – especially if cloud providers have a large footprint such that the RTT to/from the cloud is small – and provides an attractive benefit in the simplicity of the architecture. Only the APLOMB gateway needs to be cloud-aware and no modification is required to existing enterprise applications. Naming and addressing for enterprise hosts are unchanged. Furthermore, bounce redirection requires minimal functionality and configuration at the gateway – a few static rules to redirect traffic to a cloud PoP.

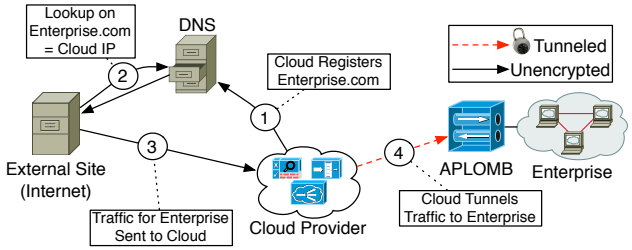
IP Redirection: To avoid the extra round-trips in Figure 5(a), we can route traffic directly to/from the cloud, such that traffic goes from an Internet user directly to the cloud, and then to the enterprise. One possible design to achieve this is to have the cloud provider announce IP prefix P on the enterprise's behalf. Hosts communicating with the enterprise direct their traffic to P and thus their enterprise-bound traffic is received by the cloud provider. The cloud provider, after processing the traffic, tunnels the traffic to the



(a) "Bounce" redirection inflates latency.



(b) Direct IP redirection in multi-PoP deployments cannot ensure that bidirectional traffic traverses the same PoP.



(c) DNS-based redirection minimizes latency and allows providers to control PoP selection for each request.

Figure 5: Comparison of redirection architectures.

enterprise gateways. However, in practice enterprises may want to redirect through several of the cloud provider's datacenters for improved latency, load distribution, and fault tolerance. (We'll discuss this "multi-pop" selection in depth in the following subsection.) For this, the cloud provider might advertise P from multiple PoPs so that client traffic is effectively anycasted to the closest PoP. Unfortunately, IP-based redirection in a multi-PoP scenario will break the semantics of stateful middleboxes as it cannot ensure that traffic from a client a to enterprise b will be routed to the same cloud PoP as that from b to a, as shown in Figure 5(b). Furthermore, because traffic is redirected at the network layer based on BGP path selection criteria (*e.g.*, AS hops), the enterprise or the cloud provider has little control over which PoP is selected and cannot, for example, pick PoPs to optimize end-to-end latency.

DNS Redirection: To allow redirection using multiple cloud PoPs, we can rely on DNS-based redirection similar to its use in CDNs (as shown in Figure 5(c)). Here, the cloud provider runs DNS resolution on the enterprise's behalf and registers DNS names for the client's external services, in this example '`MyEnterprise.com`' (step 1) [2]. A user accessing `MyEnterprise.com` (step 2) is directed to the cloud PoP via DNS (step 3). The traffic is then processed by relevant middleboxes and tunneled to the enterprise (step 4). Outbound return traffic from the enterprise (and traffic initiated by enterprise hosts) is also easy to control; the APLOMB gateway device uses a simple lookup to send traffic to the same PoP the inbound client would receive from DNS. This ensures that traffic between the enterprise and an external host will always traverse the same cloud PoP, even when network-level routing changes. However, DNS-based redirection introduces a challenge in outsourcing traffic for legacy applications which provide external clients with IP addresses rather than DNS names.

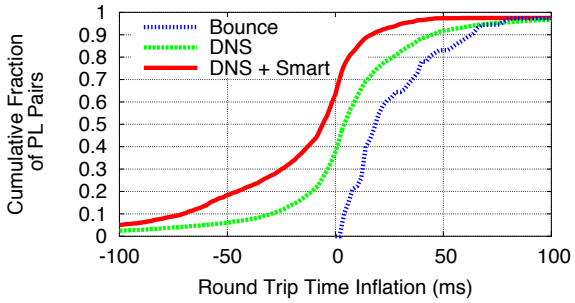


Figure 6: Round Trip Time (RTT) inflation when redirecting traffic between US PlanetLab nodes through Amazon PoPs.

In our system architecture, we choose DNS redirection because it avoids the latency penalty of bounce redirection and provides more control over redirection than pure IP redirection.

3.2 Minimizing Latency

DNS redirection can force bidirectional traffic through the same cloud provider PoP, thus bringing traffic to a 'choke point' where network processing can occur. However, any redirection has the potential to inflate end to end latency between an outsourcing enterprise and external clients – even if one avoids obvious pitfalls like the 'bounce' architecture. We now consider a DNS-based outsourcing architecture where the cloud provider has multiple PoPs, and discuss latency-optimized PoP selection strategies as well as how large a footprint a cloud provider should possess in order to provide redirection services that do not excessively inflate latency.

3.2.1 Smart Redirection

We quantify the latency impact of the bounce, IP, and DNS redirection options in Figure 6 using measurements from over 300 PlanetLab nodes and twenty Amazon CloudFront locations. We consider an enterprise "site" located at one of fifty US-based PlanetLab sites while the other PlanetLab nodes emulate "clients". For each site e, we pick the closest Amazon CloudFront PoP $P_e^* = \arg\min_P Latency(P, e)$ and measure the impact of tunneling traffic to/from this PoP.

Figure 6 shows that the bounce redirection can increase the end-to-end RTT by more than 50ms for 20% of inter-PlanetLab paths. The basic DNS-based redirection, where the enterprise tunnels traffic to and from the cloud PoP to which it has the minimum RTT, reduces the 80^{th} percentile of latency inflation $2\times$ compared to bounce redirection. In fact, for more than 30% of the pairwise measurements, the latency is actually lower than the direct IP path. This is because of well-known triangle inequality violations in inter-domain routing and the fact that cloud providers are well connected to tier-1/2 ISPs [31].

To reduce latency further, we redirect traffic not through the cloud PoP with the minimum RTT to and from the enterprise, but redirect traffic on a per-destination basis through the PoP that minimizes end-to-end latency. That is, instead of using a single fixed PoP P_e^* for each enterprise site e, we choose the optimal PoP for each client and site c, e combination (i.e., $\arg\min_P Latency(P, c) + Latency(P, e)$). We quantify the inflation with this redirection using the earlier setup with Amazon CloudFront sites as PoPs and PlanetLab nodes as enterprise sites. Figure 6 shows that with "Smart Redirection", more than 70% of the cases have zero or negative inflation and 90% of all traffic has less than 10ms inflation.

Smart redirection requires that the APLOMB appliance redirect traffic to different PoPs based on the client's IP and maintain persistent tunnels to multiple PoPs instead of just one tunnel to its clos-

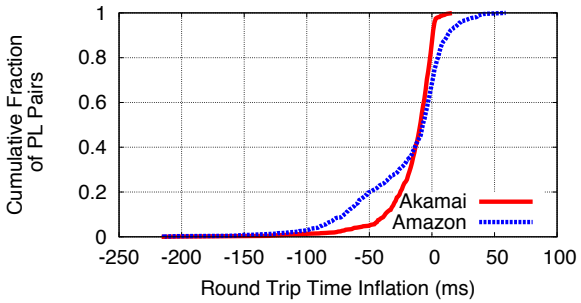


Figure 7: PlanetLab-to-PlanetLab RTTs with APLOMB redirection through Amazon and Akamai.

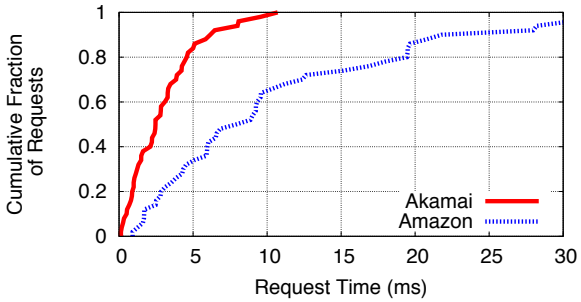


Figure 8: Direct RTTs from PlanetLab to nearest Akamai or Amazon redirection node.

est PoP. This requirement is modest: mappings for PoP selection can be computed at the cloud provider and pushed to APLOMB appliances, and today's commodity gateways can already support hundreds of persistent tunneled connections.

3.2.2 Provider Footprint

Next, we analyze how the middlebox provider's choice of geographic footprint impacts latency. Today's clouds have a few tens of global PoPs and expand as new demand arises [4]. For greater coverage, we could envision a middlebox provider with a footprint comparable to CDNs such as Akamai with thousands of vantage points [45]. While it is clear that a larger footprint provides lower latency, it is not clear how large a footprint is required in the context of outsourcing middleboxes.

To understand the implications of the provider's footprint, we extend our measurements to consider a cloud provider with an Akamai-like footprint using IP addresses of over 20,000 Akamai hosts [26]. First, we repeat the the end-to-end latency analysis for paths between US PlanetLab nodes and see that a larger, edge-concentrated Akamai footprint reduces tail latency, but the overall changes are marginal compared to a smaller but well connected Amazon-like footprint. End-to-end latency is the metric of interest when outsourcing most middleboxes – all except for 'location dependent' appliances. Because roughly 70% of inter-PlanetLab node paths actually experience *improved* latency, these results suggest that a middlebox provider can service most customers with most types of middleboxes (*e.g.*, NIDS, firewalls) with an Amazon-like footprint of a few tens of PoPs.

Some 'location dependent' middleboxes such as cache/proxies and protocol accelerators, are best served by cloud PoPs that are close to enterprise (minimizing latency to and from the middlebox). To evaluate whether we can outsource even these location dependent middleboxes without a high latency penalty (we discuss bandwidth penalties in §3.3), we look at the RTT between each PlanetLab node and its closest Akamai node in Figure 8. In this case, we see a more dramatic impact of Akamai's footprint as it

Type of Middlebox	Enterprise Device	Cloud Footprint
IP Firewalls	Basic APLOMB	Multi-PoP
Application Firewalls	Basic APLOMB	Multi-PoP
VPN Gateways	Basic APLOMB	Multi-PoP
Load Balancers	Basic APLOMB	Multi-PoP
IDS/IPS	Basic APLOMB	Multi-PoP
WAN optimizers	APLOMB+	CDN
Proxies	APLOMB+	CDN

Table 2: Complexity of design and cloud footprint required to outsource different types of middleboxes.

provides sub-millisecond latencies to 20% of sites, and less than 5 ms latencies to almost 90% of sites. An Amazon-like footprint provides only 30% of sites with an RTT <5 ms. Our results suggest that an Amazon-like footprint can provide low latency only for a limited portion of US clients; to provide low latency service f nation-wide client base, an Akamai-like footprint is necessary.

3.3 Location Dependent Services

As discussed earlier, location dependent appliances optimize both latency and bandwidth consumption. The above results show that, with an appropriate provider footprint, these appliances can still offer significant latency savings. We now consider whether we can retain bandwidth savings. Unfortunately, this is a harder problem since bandwidth optimizations must fundamentally be implemented before the enterprise access link in order to be useful. We discuss three possible options next.

The first is to simply not outsource these appliances. From the enterprises we surveyed and Figure 1, we see that WAN optimizers and proxies are currently only deployed in large enterprises and that APLOMB is of significant value even if it doesn't cover proxies and WAN optimizers. Nevertheless, we would like to do better and hence ask whether full-fledged middleboxes are really needed or whether we could achieve much of their benefit with a more minimal design.

The second option we consider is to embed *general-purpose* traffic compression capabilities into the APLOMB appliance – we call this augmented appliance APLOMB+. We evaluate APLOMB+ against traditional WAN optimizers in §5.3 using measurements from a large enterprise and show that protocol-agnostic compression [21] can provide similar bandwidth savings.

We cannot, however, claim that such a minimal capability exists for every conceivable middlebox (*e.g.*, consider an appliance that encodes outgoing traffic for packet loss protection), nor that APLOMB+ can fully replicate the behavior of dedicated appliances. Thus, our third option considers more general support for such functions at the APLOMB gateway. For this, we envision a more "active" appliance architecture that can run specialized software modules (*e.g.*, a FEC encoder) that are dynamically installed by the cloud provider or the enterprise administrator. Although more general, this increases device and configuration complexity, and we do not consider this option in this paper.

3.4 Discussion

In the following section, we describe our APLOMB design, which implements all of the above design choices. We now reflect on our original goal—outsourcing as many middleboxes as possible—and investigate how well our architecture achieves this goal.

Table 2 identifies the design option (and hence its associated complexity) that is needed to retain the functional equivalence of the middleboxes observed in our survey.[2] We consider "outsource-

[2]A subtle issue is whether load balancers really need to be physically close to backend servers; *e.g.*, for identifying load imbalances

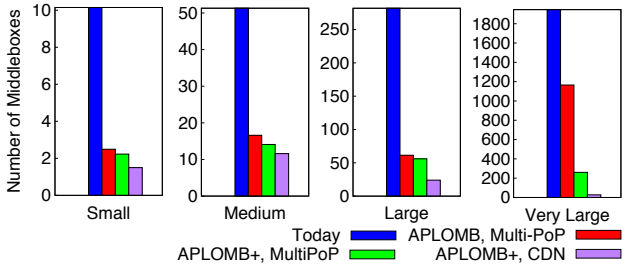


Figure 9: Average number of middleboxes remaining in enterprise under different outsourcing options.

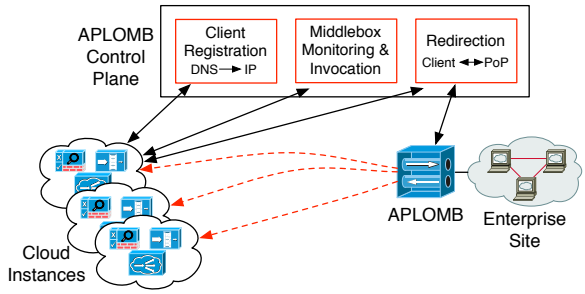


Figure 10: Architectural components of APLOMB.

able" middleboxes to be those whose functionality can be performed in the cloud with little (<5ms) or no latency inflation for at least 80% of PoPs. Most middleboxes can be outsourced with a 'Basic APLOMB' device (tunneling to and from the cloud using smart redirection, but without compression) and only an Amazon-like 'Multi-PoP' footprint. The only exceptions are WAN optimizers and proxies; these require an APLOMB+ (compression-performing) gateway and may require an Akamai-like 'CDN' cloud footprint for more remote enterprise locations. We also note that with an Akamai-like footprint, the bounce redirection becomes a feasible and potentially much simpler outsourcing solution.

Based on this analysis, Figure 9 shows the number of middleboxes that remain in an average small, medium, and large enterprise under different outsourcing deployment options. This suggests that small and medium enterprises can achieve almost all outsourcing benefits with a basic APLOMB architecture using today's cloud providers. (We revisit the remaining middleboxes, which are largely 'internal firewalls', in §5.3.) The same basic architecture can outsource close to 50% of the appliances in very large enterprise networks; using APLOMB+ increases the percentage of outsourced appliances to close to 90%.

4. APLOMB: DETAILED DESIGN

In describing the detailed design of the APLOMB architecture, we focus on three key components as shown in Figure 10: (1) a APLOMB gateway to redirect enterprise traffic, (2) the corresponding functions and middlebox capabilities at the cloud provider, and (3) a control plane which is responsible for managing and configuring these components.

4.1 Enterprise Configuration

Redirecting traffic from the enterprise client to the cloud middlebox provider is simple: an APLOMB gateway is co-located with the enterprise's gateway router, and enterprise administrators supply the cloud provider with a manifest of their address allocations.

at sub-millisecond granularity. Our conversations with administrators suggest that this is not a typical requirement and thus that load balancers can be outsourced.

APLOMB changes neither routing nor switching, and end hosts require no new configuration.

4.1.1 Registration

APLOMB involves an initial registration step in which administrators provide the cloud provider with an *address manifest*. These manifests list the enterprise network's address blocks in its *private* address space and associates each address or prefix with one of three types of address records:

Protected services: Most private IP addresses are registered as protected services. These address records contain an IP address or prefix and the public IP address of the APLOMB device at the gateway to the registered address(es). This registration allows inter-site enterprise traffic to traverse the cloud infrastructure (*e.g.* a host at site A with address 10.2.3.4 can communicate with a host at site B with address 10.4.5.6, and the cloud provider knows that the internal address 10.4.5.6 maps to the APLOMB gateway at site B). The cloud provider allocates no permanent public IP address for hosts with 'protected services' addresses; Internet-destined connections instead undergo traditional NAPT.

DNS services: For hosts which accept incoming traffic, such as web servers, a publicly routeable address must direct incoming traffic to the appropriate cloud PoP. For these IP addresses, the administrator requests DNS service in the address manifest, listing the private IP address of the service, the relevant APLOMB gateway, and a DNS name. The cloud provider then manages the DNS records for this address on the enterprise client's behalf. When a DNS request for this service arrives, the cloud provider (dynamically) assigns a public IP from its own pool of IP addresses and directs this request to the appropriate cloud PoP and subsequent APLOMB gateway.

Legacy IP services: While DNS-based services are the common case, enterprise may require legacy services that require fixed IP addresses. For these services, the enterprise registers the internal IP address and corresponding APLOMB gateway, and the cloud provider allocates a static public IP address at a single PoP for the IP service. For this type of service, we fall back to the single-PoP Cloud-IP solution rather than DNS redirection discussed in §3.

4.1.2 APLOMB gateway

The APLOMB gateway is logically co-located with the enterprise's gateway router and has two key functions: (1) maintaining persistent tunnels to multiple cloud PoPs and (2) steering the outgoing traffic to the appropriate cloud PoP. The gateway registers itself with the cloud controller (§4.3), which supplies it with a list of cloud tunnel endpoints in each PoP and forwarding rules (5-tuple → cloud PoP Identifier) for redirection. (The gateway router blocks all IP traffic into the network that is not tunneled to a APLOMB gateway.) For security reasons, we use encrypted tunnels (e.g., using OpenVPN) and for reducing bandwidth costs, we enable protocol-agnostic redundancy elimination [21]. Note that the functionality required of the APLOMB gateway is simple enough to be bundled with the egress router itself or built using commodity hardware.

For scalability and fault tolerance, we rely on traditional load balancing techniques. For example, to load balance traffic across multiple APLOMB gateways, the enterprise's private address space can be split to direct traffic to, *e.g.* 10.1.0.0/17 to one gateway, and 10.1.128.0/17 to another. To handle gateway failures, we envision APLOMB hardware with fail-open NICs configured to direct the packets to a APLOMB replica under failure. Since each APLOMB box keeps almost no per-flow state, the replica receiving

traffic from the failed device can start forwarding the new traffic without interruption to existing flows.

4.2 Cloud Functionality

The cloud provider has three main tasks: (1p publicly addressable IP addresses to the appropriate enterprise customer and internal private address, (2) apply middlebox processing services to the customers' traffic according to their *policies* (§4.3), and (3) tunnel traffic to and from the appropriate APLOMB gateways at enterprise sites. Thus, the core components at the cloud PoP are:

- *Tunnel Endpoints* to encapsulate/decapsulate traffic from the enterprise (and to encrypt/decrypt and compress/decompress if enabled)
- *Middlebox Instances* to process the customers' traffic
- *NAT Devices* to translate between publicly visible IP addresses and the clients' internal addresses. NAT devices manage statically configured IP to IP mappings for DNS and Legacy IP services, and generate IP and port mappings for Protected Services (§4.1).
- *Policy switching* logic to steer packets between the above components.

Fortunately, it is possible to realize each of these components with existing solutions and there are many research and commercial solutions to provide these features (*e.g.* [34, 25, 8]). These solutions differ along two key dimensions depending on whether the middlebox services are: (1) provided by the cloud infrastructure provider (e.g., Amazon) or by third-party cloud service providers running within these infrastructure providers (e.g., [17]), and (2) realized using hardware- (e.g., [14, 11]) or software-based middleboxes (e.g., [41, 46, 13, 42]. Our architecture is agnostic to these choices and accommodates a broad range of deployment scenarios as long as there is some feasible path to implement the four components described above. The specific implementation we present in this paper runs as a third-party service using software-based middleboxes over an existing infrastructure provider.

4.3 Control Plane

A driving design principle for APLOMB is to keep the new components introduced by our architecture that are on the critical path – *i.e.*, the APLOMB gateway device and the cloud terminal endpoint – as simple and as stateless as possible. This not only reduces the enterprise's administrative overhead but also enables seamless transition in the presence of hardware and network failures. To this end, the APLOMB Control Plane manages the relevant network state representing APLOMB gateways, cloud PoPs, middlebox instances, and tunnel endpoints. It is responsible for determining optimal redirection strategies between communicating parties, managing and pushing middlebox policy configurations, and dynamically scaling cloud middlebox capacity to meet demands.

In practice, the control plane is realized in a *cloud controller*, which manages every APLOMB gateway, middlebox, tunneling end point, and the internals of the cloud switching policy.[3] Each entity (APLOMB device, middlebox, *etc.*) registers itself with the controller. The controller sends periodic 'heartbeat' health checks to each device to verify its continued activity. In addition, the controller gathers RTTs from each PoP to every prefix on the Internet (for PoP selection) and utilization statistics from each middlebox (for adaptive scaling). Below we discuss the redirection optimization, policy management, and middlebox scaling performed by the cloud controller.

[3]While the cloud controller may be in reality a replicated or federated set of controllers, for simplicity this discussion refers to a single logically centralized controller.

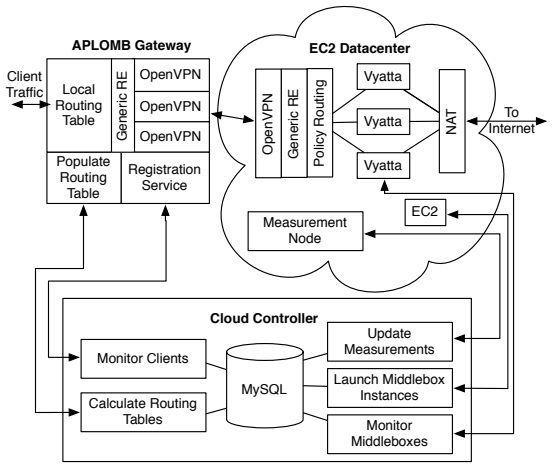


Figure 11: Software architecture of APLOMB.

Redirection Optimization. Using measurement data from the cloud PoPs, the cloud controller pushes the current best (as discussed in §3.2) tunnel selection strategies to the APLOMB gateways at the enterprise and mappings in the DNS. To deal with transient routing issues or performance instability, the cloud controller periodically updates these tunneling configurations based on the newest measurements from each cloud PoP.

Policy Configuration. The cloud controller is also responsible for implementing enterprise- and middlebox-specific *policies*. Thus, the cloud provider provides a rich policy configuration interface that exports the available types of middlebox processing to enterprise administrators and also implements a programmatic interface to specify the types of middlebox processing required [33]. Enterprise administrators can specify different *policy chains* of middlebox processing for each class of traffic specified using the traditional 5-tuple categorization of flows (i.e., source and destination IPs, port values and the protocol). For example, an enterprise could require all egress traffic to go through a firewall $\rightarrow$ exfiltration engine $\rightarrow$ proxy. and require that all ingress traffic traverse a firewall $\rightarrow$ IDS, and all traffic to internal web services further go through an application-level firewall. If appropriate, the provider may also export certain device-specific configuration parameters that the enterprise administrator can tune.

Middlebox Scaling. APLOMB providers have a great deal of flexibility in how they actually implement the desired middlebox processing. In particular, as utilization increases on a particular middlebox, the APLOMB provider simply increases the number of instances of that middlebox being utilized for a client's traffic.

Using data from heartbeat health checks on all middleboxes, the cloud controller detects changes in utilization. When utilization is high, the cloud controller launches new middleboxes and updates the policy switching framework; when utilization is low, the cloud controller deactivates excess instances. While scaling is simpler if all middlebox processing is performed in software on standard virtual machines, providers using hardware middleboxes could achieve the same result using policy switching alone. Techniques for dynamic scaling under load are well-known for cloud computing applications like web servers [12]; as such we do not go into detail here.

4.4 Implementation

We built a prototype system for cloud middlebox processing using middlebox processing services running on EC2 and APLOMB

endpoints in our lab and at the authors' homes. We consciously choose to use off-the-shelf components that run on existing cloud providers and end host systems. This makes our system easy to deploy and use and demonstrates that the barriers to adoption are minimal. Our APLOMB endpoint software can be deployed on a stand-alone software router or as a tunneling layer on an end host; installing and running the end host software is as simple as connecting to a VPN.

Figure 11 is a software architecture diagram of our implementation. We implement a cloud controller on a server in our lab and use geographically distributed EC2 datacenters as cloud PoPs. Our cloud controller employs a MySQL database to store data on middlebox nodes, RTTs to and from cloud PoPs, and registered clients. The cloud controller monitors APLOMB devices, calculates and pushes routing tables to the APLOMB devices, requests measurements from the cloud PoPs, monitors middlebox instances, and scales middlebox instances up or down as demand varies.

At the enterprise or the end host, the APLOMB gateway maintains several concurrent VPN tunnels, one to a remote APLOMB at each cloud PoP. On startup, the APLOMB software contacts the cloud controller and registers itself, fetches remote tunnel endpoints for each cloud PoP, and requests a set of initial tunnel redirection mappings. A simple tunnel selection layer, populated by the cloud controller, directs traffic to the appropriate endpoint tunnel, and a redundancy elimination encoding module compresses all outgoing traffic. When run on a software router, ingress traffic comes from an attached hosts for whom the router serves as their default gateway. Running on a laptop or end host, static routes in the kernel direct application traffic to the appropriate egress VPN tunnel.

EC2 datacenters host tunnel endpoints, redundancy elimination decoders, middlebox routers, and NATs, each with an inter-device switching layer and controller registration and monitoring service. For tunneling, we use OpenVPN [10], a widely-deployed VPN solution with packages for all major operating systems. We use a Click [36] implementation of the redundancy elimination technique described by Anand et al [21]. For middlebox processing, we use Vyatta [16], a customizable software middlebox. Our default Vyatta installation performs firewalling, intrusion detection, caching, and application-layer web filtering. Policy configurations (§4.3) are translated into Vyatta configurations such that each client can have a unique Vyatta configuration dependent on their needs. Finally, each cloud PoP also hosts one 'measurement node', which periodically issues ping measurements for RTT estimation to assist in PoP selection.

5. EVALUATION

We now evaluate APLOMB. First, we present performance benchmarks for three common applications running over our implementation (§5.1).We then demonstrate APLOMB's dynamic scaling capability and its resilience to failure (§5.2). Having shown that APLOMB is practical, we return to our goal of outsourcing all middlebox functionality in an enterprise with a trace-driven evaluation of middlebox outsourcing using APLOMB, applied to data from a middlebox deployment in a large enterprise (§5.3).

5.1 Application Performance

We first demonstrate that APLOMB's architecture is practical for enterprise use with performance benchmarks for common applications using our APLOMB implementation.

HTTP Page Loads: In Figure 12, we plot page load times (fetching the front page and all embedded content) from a university network for the Alexa top 1,000 most popular web pages with and without APLOMB processing. We performed this experiment with

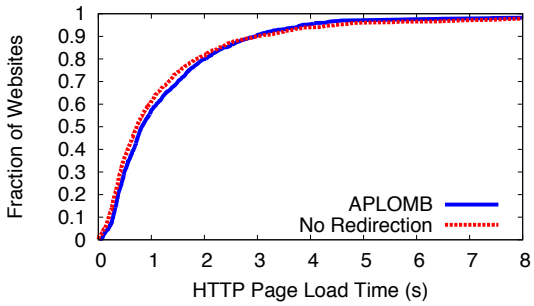


Figure 12: CDF of HTTP Page Load times for Alexa top 1,000 sites with and without APLOMB.

a vacant cache. For pages at the 50^{th} percentile, page loads without APLOMB took 0.72 seconds, while page loads with took 0.82 seconds. For pages at the 95^{th} percentile, using APLOMB results in shorter page load times: 3.85 seconds versus 4.53 seconds.

BitTorrent: While we don't expect BitTorrent to be a major component of enterprise traffic, we chose to experiment with Bit Torrent because it allowed us to observe a bulk transfer over a long period of time, to observe many connections over our infrastructure simultaneously, and to establish connections to non-commercial endpoints. We downloaded a 698MB public domain film over BitTorrent with and without APLOMB from both a university network and from a residential network, five times repeatedly. The average residential download took 294 seconds without APLOMB, with APLOMB the download speed increased 2.8% to 302 seconds. The average university download took 156 seconds without APLOMB, with APLOMB the average download took 165 seconds, a 5.5% increase.

Voice over IP: Voice over IP (VoIP) is a common enterprise application, but unlike the previously explored applications, VoIP performance depends not only on low latency and high bandwidth, but on low *jitter*, or variance in latency. APLOMB easily accommodates this third demand: we ran VoIP calls over APLOMB and for each call logged the jitter estimator, a running estimate of packet interarrival variance developed for RTP. Industry experts cite 30ms of one-way jitter as a target for maximum acceptable jitter [7]. In the first call, to a residential network, median inbound/outbound jitter with APLOMB was 2.49 ms/2.46 ms and without was 2.3 ms/1.03 ms. In the second, to a public WiFi hotspot, the median inbound/outbound jitter with APLOMB was 13.21 ms/14.49 ms and without was 4.41 ms/4.04 ms.

In summary, these three common applications suffer little or no penalty when their traffic is redirected through APLOMB.

5.2 Scaling and Failover

To evaluate APLOMB's dynamic scaling, we measured traffic from a single client to the APLOMB cloud. Figure 13 shows capacity adapting to increased network load over a 10-minute period. The client workload involved simultaneously streaming a video, repeatedly requesting large files over HTTP, and downloading several large files via BitTorrent. The resulting network load varied significantly over the course of the experiment, providing an opportunity for capacity scaling. The controller tracks CPU utilization of each middlebox instance and adds additional capacity when existing instances exceed a utilization threshold for one minute.

While middlebox capacity lags changes in demand, this is primarily an artifact of the low sampling resolution of the monitoring infrastructure provided by our cloud provider. Once a new middlebox instance has been allocated and initialized, actual switchover time to begin routing traffic through it is less than 100ms. To handle failed middlebox instances, the cloud controller checks for reacha-

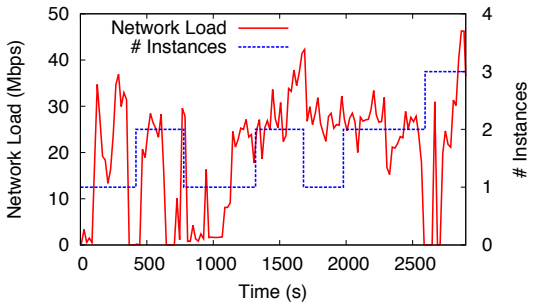


Figure 13: Network load (Y_1) and number of software middlebox instances (Y_2) under load. Experiment used low-capacity instances to highlight scaling dynamics.

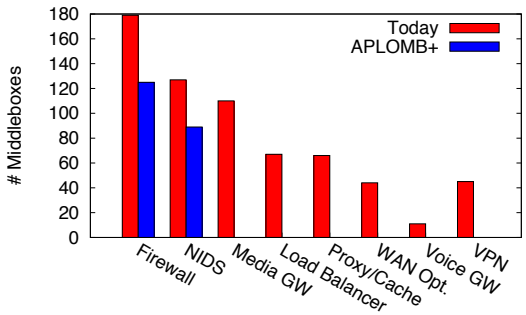


Figure 14: Number of middleboxes in the enterprise with and without APLOMB+. The enterprise has an atypical number of 'internal' firewalls and NIDS.

bility between itself and individual middlebox instances every second; when an instance becomes unreachable, APLOMB ceases routing traffic through it within 100ms. Using the same mechanism, the enterprise APLOMB can cope with failure of a remote APLOMB, re-routing traffic to another remote APLOMB in the same or even different cloud PoP, providing fault-tolerance against loss of an entire datacenter.

5.3 Case Study

We set out with the goal of outsourcing as many middleboxes as possible and reducing enterprise costs, all the while without increasing bandwidth utilization or latency. We revisit this using the data from the very large enterprise to determine:

- How many middleboxes can the enterprise outsource?
- What are the gains from elastic scaling?
- What latency penalty will inter-site traffic suffer?
- How much does the enterprise's bandwidth costs increase?

Middleboxes Outsourced: Figure 14 shows that the large enterprise can outsource close to 60% of the middleboxes under a CDN footprint with APLOMB+.

This high fraction of outsourceability comes despite an atypically high deployment of "internal" firewalls and NIDS at this enterprise. Internal firewalls protect a host or subnetwork not only from Internet-originated traffic, but from traffic originated within the enterprise; the most common reason we found for these deployments was PCI compliance for managing credit card data. While the average enterprise of this size deploys 27.7 unoutsourceable internal firewalls, this enterprise deploys over 100 internal firewalls. From discussions with the network's administrators, we learned these were installed in the past to protect internal servers against worms that preferentially scanned internal prefixes, *e.g.* CodeRed and Nimda. As more IT infrastructure moves to the cloud (see §6), many internal firewalls will be able to move to the cloud as well.

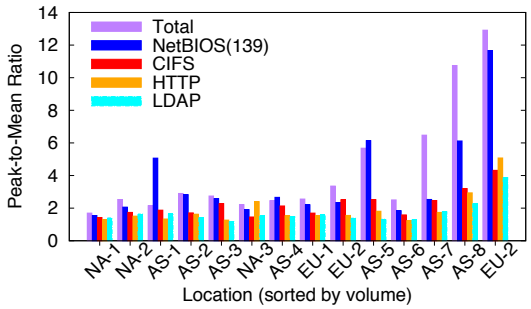


Figure 15: Ratio of peak traffic volume to average traffic volume, divided by protocol.

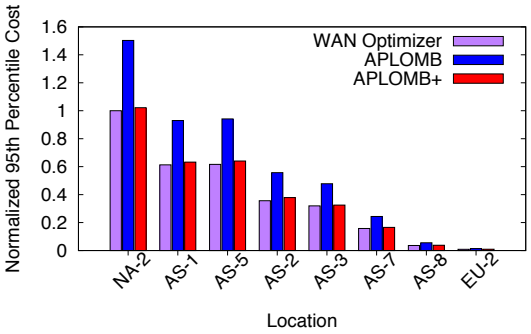


Figure 16: 95^{th} percentile bandwidth without APLOMB, with APLOMB, and with APLOMB+.

Cost Reduction: To evaluate benefits from elastic scaling, in Figure 15 we focus on each site of the enterprise and show the ratio of peak-to-average volumes for total inter-site traffic. We use sites across three continents: North America (NA-x), Asia (AS-x), and Europe (EU-x). The peak represents a conservative estimate of the traffic volume the enterprise has provisioned at the site, while the average is the typical utilization; we see that most sites are provisioned over 2× their typical load, and some of the smaller sites as much as 12×! In addition, we show peak-to-average values for the top four protocols in use. The per-protocol numbers are indicative of elasticity savings per middlebox, as different protocols are likely to traverse different middleboxes.

Latency: We measured redirection latency for inter-site traffic between the top eleven sites of the enterprise through the APLOMB infrastructure by pinging hosts at each site from within EC2. We found that for more than 60% of inter-site pairs, the latency with redirection is almost identical to the direct RTT. We found that most sites with inflated latency were in Asia, where EC2 does not have a wide footprint.

We also calculated a weighted inflation value, weighted by traffic volume and found that in expectation a typical redirected packet experiences only 1.13 ms of inflation. This results from the fact that the inter-site pairs with high traffic volume actually have negative inflation, by virtue of one or both endpoints being in the US or Europe, where EC2's footprint and connectivity is high.

Bandwidth: Last, we evaluate bandwidth inflation. We ran a traffic trace with full packet payloads collected at a different small enterprise [38] through our APLOMB prototype with and without generic redundancy elimination. Without Generic RE, the bandwidth utilization increased by 6.2% due to encryption and encapsulation overhead. With Generic RE, the bandwidth utilization reduced by 28%, giving APLOMB+ a 32% improvement over basic APLOMB.

As we observed in §2, many larger enterprises already compress their inter-site traffic using WAN optimizers. To evaluate the im-

pact of switching compression for inter-site traffic from a traditional WAN optimizer solution to APLOMB+, we compared our observed benefits to those provided by WAN optimizers at eight of the large enterprise sites. In Figure 16, we measure the bandwidth cost of a given site in terms of the 95^{th} percentile of the total traffic volume with a WAN Optimizer, with APLOMB, and with APLOMB+. With APLOMB, the worst case inflation is 52% in the median case and at most 58%; APLOMB+ improves this to a median case of 3.8% inflation and a worst case of 8.1%.

6. DISCUSSION

Before concluding, we mention some final thoughts on the future of "hybrid" enterprise/cloud architectures, potential cost models for bandwidth, and security challenges that continue to face APLOMB and cloud computing.

IT Outsourcing and Hybrid Clouds: APLOMB complements the ongoing move by enterprises from locally-hosted and managed infrastructure to outsourced cloud infrastructure. A network administrator at one large enterprise we surveyed reported their company's management had issued a broad mandate to moving a significant portion of their IT infrastructure to the cloud. Federal government agencies are also rapidly moving their IT infrastructure to the cloud, in compliance with a mandate to adopt a "cloud first" policy for new services and to reduce the number of existing federal datacenters by 800 before 2015 [37]. As these services move to the cloud, the middleboxes protecting them (including internal firewalls, which APLOMB itself cannot outsource) will move to the cloud as well.

Nevertheless, many enterprises plan to keep at least some local infrastructure, citing security and performance concerns for applications currently deployed locally [18]. Further, user-facing devices such as laptops, desktops, smartphones, and printers will always remain within the enterprise – and the majority of middlebox services benefit these devices rather than servers. With some end hosts moving to the cloud, and the majority remaining behind in the enterprise, multiple vendors now offer services for integrating public cloud services with enterprises' existing infrastructure [1, 3], facilitating so-called "hybrid clouds" [32]. APLOMB allows administrators to evade the middlebox-related complexity in this hybrid model by consolidating middleboxes in only one deployment setting.

Bandwidth Costs: APLOMB reduces the cost of middlebox infrastructure, but it may increase bandwidth costs due to current cloud business models. Today, tunneling traffic to a cloud provider necessitates paying for bandwidth twice – once for the enterprise network's access link, and again at the cloud provider. Nevertheless, this does not mean that APLOMB will double bandwidth costs for an enterprise. We observed earlier that redundancy elimination and compression can reduce bandwidth demands at the enterprise access link by roughly 30%. This optimization is not possible without redirection through a cloud PoP, and could allow a lower capacity, less expensive access link to satisfy an enterprise's needs.

The largest factor in the cost of APLOMB for an enterprise is the bandwidth cost model used by a cloud provider. Today, cloud providers price bandwidth purely by volume; for example, Amazon EC2 charges between \$0.05-\$0.12 per GB of outgoing traffic, decreasing as volume increases (all incoming traffic is free). On the other hand, a dedicated APLOMB service provider would be able to take advantage of wholesale bandwidth, which is priced by transfer rate. We convert between the two pricing strategies (per-GB and per-Mbps) with the rough conversion factor of 1Mbps sustained monthly throughput equaling 300GB per month. This is in comparison with "wholesale" bandwidth prices of \$3-\$5 per Mbps for high-volume customers. As a result, though current pricing strategies are not well-suited for APLOMB, a dedicated APLOMB provider could offer substantially lower prices. Indeed, Amazon offers a bulk-priced bandwidth service, "DirectConnect", which offers substantially lower per-GB costs for high-volume customers [1]. Table 3 provides a comparison of the bandwidth costs for a hypothetical enterprise which transfers 500TB of traffic per month to and from a cloud service provider under each of these models. These charges a minimal compared to expected savings in hardware, personnel, and other management costs.

Pricing Model	Total Cost	\$/GB	\$/Mbps
Standard EC2	30003.20	0.0586	17.58
Amazon DirectConnect	11882.50	0.0232	6.96
Wholesale Bandwidth	6826.70	0.0133	4.00

Table 3: Cost comparison of different cloud bandwidth pricing models given an enterprise with a monthly transfer volume of 500TB (an overestimate as compared to the very large enterprise in our study); assumes conversion rate of 1Mbps of sustained transfer equals 300GB over the course of a month.

Security Challenges: Adopting APLOMB brings with it the same security questions as have challenged cloud computing. These challenges have not stopped widespread adoption of cloud computing services, nor the willingness of security certification standards to certify cloud services (for example, services on Amazon EC2 can achieve PCI-1 compliance, the highest level of certification for storing credit card data). However, these challenges remain concerns for APLOMB and cloud computing in general. Just as cloud storage services have raised questions about providing a cloud provider unencrypted access to data, cloud middlebox services give the cloud provider unencrypted access to traffic flows. Although VMs and other isolation techniques aim to protect customers of a cloud service from other, malicious, customers, some have demonstrated in the cloud computing context information leakage, *e.g.* through side-channels [40]. APLOMB encrypts tunneled traffic to and from the enterprise to protect against man-in-the-middle attacks, and allocates each client it's own set of VMs for middlebox processing, but ultimately it will not appeal to companies whose security policies restrict them from cloud computing in general.

7. RELATED WORK

Our work contributes to and draws inspiration from a rich corpus of work in cloud computing, redirection services, and network management.

Cloud Computing: The motivation for APLOMB parallels traditional arguments in favor of cloud computing, many of which are discussed by Armbrust et al. [23]. APLOMB also adapts techniques from traditional cloud solutions, *e.g.* utilization monitoring and dynamic scaling [12], and DNS-based redirection to datacenters with optimal performance for the customer [44].

Middlebox Management: Others have tackled middlebox management challenges within the enterprise [33, 34, 24, 27, 42]. Their solutions offer insights we can apply for managing middleboxes within the cloud – *e.g.*, the policy-routing switch of Joseph et al. [34], the management plane of Ballani et al. [24], and the consolidated appliance of Sekar et al. [42]. None of these proposals consider moving middlebox management out of the enterprise entirely, as we do. Like us, ETTM [27] proposes removing middleboxes from the enterprise network but, where we advocate moving them to the cloud, ETTM proposes the opposite: pushing middlebox processing to enterprise end hosts. As such, ETTM still retains the problem

of middlebox management in the enterprise. Sekar et al [42] report on the middlebox deployment of a single large enterprise; our survey is broader in scope (covering a range of management and failure concerns) and covers 57 networks of various scales. They also propose a consolidated middlebox architecture that aims to ameliorate some of the administrative burden associated with middlebox management, but they do not go so far as to propose removing middleboxes from the enterprise network entirely.

Redirection Services: Traffic redirection infrastructures have been explored in prior work [22, 43, 47] but in the context of improving Internet or overlay routing architectures as opposed to APLOMB's goal of enabling middlebox processing in the cloud. RON showed how routing via an intermediary might improve latency; we report similar findings using cloud PoPs as intermediaries. Walfish et al. [47] propose a clean-slate architecture, DOA, by which end hosts explicitly address middleboxes. Gibb et al. [30] develop a service model for middleboxes that focuses on service-aware routers that redirect traffic to middleboxes that can be in the local network or Internet.

Cloud Networking: Using virtual middlebox appliances [16] reduces the physical hardware cost of middlebox ownership, but cannot match the performance of hardware solutions and does little to improve configuration complexity. Some startups and security companies have cloud-based offerings for specific middlebox services: Aryaka [5] offers protocol acceleration; ZScalar [17] performs intrusion detection; and Barracuda Flex [6] offers web security. To some extent, our work can be viewed as an extreme extrapolation of their services and we provide a comprehensive exploration and evaluation of such a trend. CloudNaaS [25] and startup Embrane [8] aim at providing complete middlebox solutions for enterprise services that are *already* in the cloud.

8. CONCLUSION

Outsourcing middlebox processing to the cloud relieves enterprises of major problems caused by today's enterprise middlebox infrastructure: cost, management complexity, capacity rigidity, and failures. Our survey of 57 enterprise network managers guides the design of APLOMB, a practical system for middlebox processing in the cloud. APLOMB succeeds in outsourcing the vast majority of middleboxes from a typical enterprise network without impacting performance, making scalable, affordable middlebox processing accessible to enterprise networks of every size.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback; our shepherd, Jacobus Van der Merwe, for his guidance in developing the final draft of our paper; the NANOG operators for their enthusiastic participation in our survey; Neil Doran, Zhixin Tang, and Mark Poepping for helping us refine initial versions of our survey; and Ashok Anand for sharing the WAN optimization modules.

9. REFERENCES

[1] Amazon Direct Connect. http://aws.amazon.com/directconnect/.
[2] Amazon Route 53. http://aws.amazon.com/route53.
[3] Amazon Virtual Private Cloud. http://aws.amazon.com/vpc/.
[4] Amazon Web Services launches Brazil datacenters for its cloud computing platform. http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-newsArticle&ID=1639908.
[5] Aryaka WAN Optimization. http://www.aryaka.com.
[6] Barracuda Web Security Flex. http://www.barracudanetworks.com/ns/products/web_security_flex_overview.php.
[7] Cisco: Quality of Service Design Overview. http://www.ciscopress.com/articles/article.asp?p=357102.
[8] Embrane. http://www.embrane.com/.
[9] Network Monitoring Tools. http://slac.stanford.edu/xorg/nmtf/nmtf-tools.html.
[10] OpenVPN. http://www.openvpn.com.
[11] Palo Alto Networks. http://www.paloaltonetworks.com/.
[12] Rightscale Cloud management. http://www.rightscale.com/.
[13] Riverbed Virtual Steelhead. http://www.riverbed.com/us/products/steelhead_appliance/virtual_steelhead.php.
[14] Symantec: Data Loss Protection. http://www.vontu.com.
[15] Tivoli Monitoring Software. http://ibm.com/software/tivoli/products/monitor.
[16] Vyatta Software Middlebox. http://www.vyatta.com.
[17] ZScaler Cloud Security. http://www.zscaler.com.
[18] Cloud computing - 31 companies describe their experiences. http://www.ipanematech.com/information-center/download.php?link=white-papers/White%20Book_2011-Cloud_Computing_OBS_Ipanema_Technologies_EBG.pdf, 2011.
[19] Enterprise Network and Data Security Spending Shows Remarkable Resilience. http://www.abiresearch.com/press/3591, 2011.
[20] M. Allman and V. Paxson. TCP congestion control. RFC 5681.
[21] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker. Packet Caches on Routers: The Implications of Universal Redundant Traffic Elimination. In *Proc. of SIGCOMM*, 2008.
[22] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. In *SOSP*, 2001.
[23] M. Armbrust et al. A view of cloud computing. *Commun. ACM*, April 2010.
[24] H. Ballani and P. Francis. CONMan: a step towards network manageability. In *SIGCOMM*, 2007.
[25] T. Benson, A. Akella, A. Shaikh, and S. Sahu. Cloudnaas: a cloud networking platform for enterprise applications. In *Proc. SOCC*, 2011.
[26] D. R. Choffnes and F. E. Bustamante. Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems. In *SIGCOMM*, 2008.
[27] C. Dixon, H. Uppal, V. Brajkovic, D. Brandon, T. Anderson, and A. Krishnamurthy. ETTM: a scalable fault tolerant network manager. In *NSDI*, 2011.
[28] N. Dukkipati and N. McKeown. Why flow-completion time is the right metric for congestion control. *CCR*, January 2006.
[29] S. Floyd. HighSpeed TCP for large congestion windows. RFC 3649.
[30] G. Gibb, H. Zeng, and N. McKeown. Outsourcing network functionality. In *HotSDN*, 2012.
[31] K. P. Gummadi, H. V. Madhyastha, S. D. Gribble, H. M. Levy, and D. Wetherall. Improving the reliability of Internet paths with One-hop Source Routing. In *Proc. OSDI*, 2004.
[32] M. Hajjat, X. Sun, Y.-W. E. Sung, D. A. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani. Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud. In *SIGCOMM*, 2012.
[33] D. Joseph and I. Stoica. Modeling middleboxes. *Network, IEEE*, 22(5), 2008.
[34] D. A. Joseph, A. Tavakoli, and I. Stoica. A policy-aware switching layer for data centers. In *SIGCOMM*, 2008.
[35] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *SIGCOMM*, 2002.
[36] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM ToCS*, August 2000.
[37] V. Kundra. 25 Point Implementation Plan to Reform Federal Information Technology Management. Technical report, US CIO, 2010.
[38] M57 packet traces. https://domex.nps.edu/corp/scenarios/2009-m57/net/.
[39] N. McKeown et al. OpenFlow: enabling innovation in campus networks. *CCR*, March 2008.
[40] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In *CCS*, 2009.
[41] M. Roesch. Snort - Lightweight Intrusion Detection for Networks. In *LISA*, 1999.
[42] V. Sekar, S. Ratnasamy, M. K. Reiter, N. Egi, and G. Shi. The middlebox manifesto: enabling innovation in middlebox deployment. In *HotNets*, 2011.
[43] I. Stoica et al. Internet indirection infrastructure. *ToN*, April 2004.
[44] A. Su, D. Choffnes, A. Kuzmanovic, and F. Bustamante. Drafting behind Akamai (Travelocity-based detouring). In *SIGCOMM*, 2006.
[45] V. Valancius, N. Laoutaris, L. Massouli'e, C. Diot, and P. Rodriguez. Greening the internet with nano data centers. In *Proc. CoNEXT*, 2009.
[46] Visolve. Transparent caching using Squid. http://www.visolve.com/squid/whitepapers/trans_caching.pdf, 2006.
[47] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker. Middleboxes no longer considered harmful. In *OSDI*, 2004.