Mining Policies From Enterprise Network Configuration

Theophilus Benson*, Aditya Akella* and David A. Maltz†
*University of Wisconsin, Madison and †Microsoft Research
{tbenson,akella}@cs.wisc.edu, dmaltz@microsoft.com

ABSTRACT

Few studies so far have examined the nature of reachability policies in enterprise networks. A better understanding of reachability policies could both inform future approaches to network design as well as current network configuration mechanisms. In this paper, we introduce the notion of a *policy unit,* which is an abstract representation of how the policies implemented in a network apply to different network hosts. We develop an approach for reverse-engineering a network's policy units from its router configuration. We apply this approach to the configurations of five productions networks, including three university and two private enterprises. Through our empirical study, we validate that policy units capture useful characteristics of a network's policy. We also obtain insights into the nature of the policies implemented in modern enterprises. For example, we find most hosts in these networks are subject to nearly identical reachability policies at Layer 3.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network management

General Terms

Design, Management, Measurement

Keywords

Configuration Management

1. INTRODUCTION

Modern enterprises impose a variety of constraints on point-to-point network communication. These constraints limit an enterprise host's ability to access various network resources, including other enterprise hosts and various servers. In most enterprises, these restrictions are realized using a combination of different mechanisms in multiple network devices, including ACLs in firewalls and other middle-boxes, policy maps and packet filters in routers, and VLANs which cut across multiple network routers.

IMC'09, November 4–6, 2009, Chicago, Illinois, USA.

The network's high-level reachability policies — i.e. the specific rules that govern whether or not, and how, different network end-points can communicate — are seldom "written down" explicitly. The point-to-point reachability policies are implemented *indirectly* using the above mechanisms and hence they are buried in router configuration files, or worst still, in the minds of network engineers.

To date, there has been no systematic study of the range of reachability policies that modern enterprises implement. In large part this is because there have been no measurement tools or techniques to aid the analysis of this policy. Better understanding of enterprise reachability policies would have several benefits. In particular, it can inform the design of new approaches for implementing policies, such as clean-slate schemes [5, 4, 8, 2]. It can also point to simpler ways to configure current networks that would realize the same reachability policies while incurring significantly lower configuration complexity [3].

In this paper we introduce the notion of *policy units*, which is an abstract representation of how the global reachability policies implemented by an operator within her enterprise apply to different end-points in the enterprise (an end-point being an IP address in a subnet belonging to the network). Each policy unit is a maximal set of end-points that are all "treated alike" by the network in terms of the reachability constraints that apply to their communications with the rest of the network. Each network end-point belongs to one and only one policy unit.

Our paper makes two contributions: First, we show a method for how the policies of a network can be automatically extracted from the *static router configuration state* of the network. Second, we apply our method to 5 networks and verify the output with operators of some of the networks. Among these networks, we found the number of policy units implemented at Layer 3 varies from 2 to 40 and is largely independent of network size. However, in all these networks, over 70% of end-points are contained in just a few units. The operators reported they were interested to see which end-points fall into which policy units, indicating that the policy unit concept offers operators a new way to view their networks.

The rest of this paper is structured as follows: In the section that follows, we define a policy unit and explain our method for computing them. In §3 we present the results of our experiments on the 5 networks, and we show how the policy units vary across the networks. In §4 we describe the applications of policy atoms in network management. We discuss related work in §5 and conclude in§6.

2. DEFINITIONS AND APPROACH

In this section, we define a policy unit and describe a preliminary approach for extracting policy units from a network's configuration state.

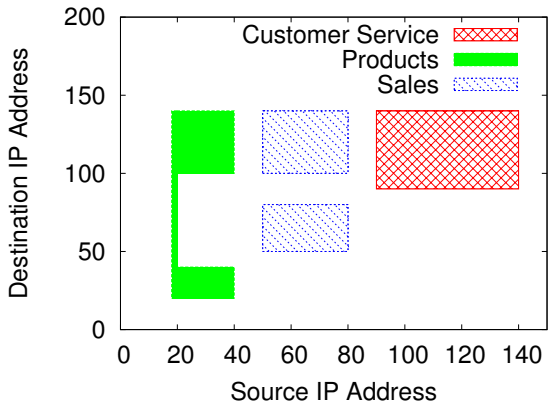


Figure 1: An enterprise with 3 departments and 4 policy units. The Products department consists of two units one of which corresponds to administrators.

2.1 What are Policy Units?

Policy units model how a network differentiates amongst its hosts in terms of the reachability constraints it imposes on them. Two end-points in an enterprise network belong to a policy unit if and only if the same set of reachability constraints apply to them when sending traffic to all network end-points. Policy units divide the set of all end-points in an enterprise network into disjoint subsets, where every end-point belongs to one and only one policy unit.

Enterprises can differ significantly in the number and kind of policy units they implement. For example, the simplest enterprise network could treat all hosts in an identical fashion – all end-points in the network would then belong to the same policy unit. In a slightly more complex scenario, policy units could align with departmental boundaries – e.g. all hosts in the CS Department could belong in one policy unit and those in EE could belong in another unit. In other more complex scenarios, enterprises may impose fine-grained distinction among hosts in a department as well as across departments. For example, consider an enterprise with three departments: Sales, Products and Customer Support. Suppose that hosts in each department can access different sets of end-points in the rest of the network, as illustrated in Figure 1. For instance, hosts in Sales have IP addresses ranging from 50 to 80, and they can reach IP address ranges 50-80 and 100-140. Similarly, hosts in Customer Service can reach IP address ranges 90-140. Suppose further that a small group of hosts in the Products department, with IP addresses 18-20, have greater reachability to the rest of the network (i.e, they can reach IP addresses 20-140) than others hosts in the department (who can only reach 20-40 and 100-140); the small group of hosts could be machines used by administrators for the entire network. Such an enterprise network would have at least four different policy units, one each corresponding to the following source IP address ranges: 18-20, 20-40, 50-80 and 90-140 (see Figure 1). Our empirical study in Section 3 shows that enterprises range from the very simple to the very complex in terms of the policy units they implement in their networks.

Formally, each policy unit is an *equivalence class* $\mathcal{E}$ on the relation $\mathcal{R}$: $\mathcal{H} \times Pow(\hat{\mathcal{H}} \times \mathcal{C}_1 \times ... \times \mathcal{C}_m \times \mathcal{A})$. Here $\mathcal{H}$ is some subset of the set of all end-points in the network, and $\hat{\mathcal{H}}$ is some subset of the set of all end-points in the network that supersedes $\mathcal{H}$. $\mathcal{A} = \{permit, deny\}$, is set of all actions that the network takes on any communication. $\mathcal{C}_i$ are the characteristics of the packets the network policy cares about, for example source port, destination port and protocol. Finally, $Pow(\hat{\mathcal{H}} \times \mathcal{C}_1 \times ... \times \mathcal{C}_m \times \mathcal{A})$ is the *power set* (the set containing all subsets of a set) of the set of all end-point/packet-characteristic/action tuples. The policy units for a network are then the equivalence classes that partition the domain of $\mathcal{R}$ (i.e., the network end-points). An equivalence class $\mathcal{E}$ of the relation $\mathcal{R}$ is a set of end-points whose communication with any end-point in the rest of enterprise is treated in exactly the same way by the network as a whole.

2.2 Deriving Policy Units from Configuration

We describe an approach for extracting policy units from a network's static configuration state.

Our strawman approach described below applies to policy units implemented in Layer 3 in enterprise networks. Thus, our scheme takes as input router configuration files. Reachability constraints at this layer are often implemented using control plane mechanisms such as routing protocol adjacencies and route filters [10, 12], and data plane mechanisms such as router ACLs [10, 12]. Since we only focus on Layer 3, each policy unit we derive is a union of otherwise disjoint IP address ranges. In other words, our approach treats each IP address in the enterprise as representing a single end-host, and tries to group together IP addresses that are similar in terms of reachability constraints into a single policy unit.

Our scheme works in three stages. First we calculate the extent of reachability between pairs of routers in the network, i.e., set of packets that can be exchange between routers. Then we calculate the reachability between pairs of subnets in the network. From this subnet-level information, we finally derive policy units using a geometric heuristic.

2.2.1 *Router-Level Reachability Sets (RRS)*

For the first stage, we employ a reachability analysis tool developed in our prior work [3]. The tool models the impact of both control and data plane mechanisms to compute the set of packets that can be exchanged between a pair of routers. This tool has two components:

(1) *Control Plane Simulation*: This simulates the interactions between routing protocols and control plane ACLs (route maps) to determine the forwarding entries for routers in the network. Our simulator accounts for the presence of VLANs and multiple routing protocols such as RIP, OSPF and BGP. The core idea is to simulate the exchange of routes from the local RIBs (route information bases) of the various routing protocols defined on each network router. Whether or not routes can be exchanged between two routers is based on the configured routing protocol adjacencies as well as physical adjacencies in the topology. We apply control plane filters before routes are exchanged to model control plane restrictions on route propagation. Whenever multiple route options are available, our tool break ties in favor of the shortest path, but this could be extended to accommodate more complex choices of routes. In the end, we generate a *forwarding table (FIB)* for each router in the network, i.e., the list of next hop routes for each destination subnet.

(2) *Applying data plane constraints*: This component models how the data plane ACLs defined in other routers on the path between a pair of routers, and filtering rules defined in on-path firewalls and middle-boxes, impact which packets are filtered before reaching the destination router. Define the path between routers $R1$ and $R2$, $path(R1, R2)$, as an ordered list of router interfaces that packets originating from router $R1$ have to transit to arrive at router $R2$ (the path is obtained by examining the FIBs in the network's routers or from the control plane simulation above). For $path(R1, R2)$, we can define the *Data Plane* and *Control Plane sets* for the router-

interfaces appearing on the paths. The Data Plane sets for an interface i on a router d appearing on $path(R1, R2)$ are defined as the sets of packets from $R1$ to $R2$ that the inbound packet filter defined for the interface allows into that interface, and the outbound packet filter allows out of that interface. Similarly, data plane sets for on-path packet filtering mechanisms like firewalls and other middleboxes. Note that because the data plane set is determined by filtering rules, each set is naturally represented as an ACL, i.e. a sequence of permit statements followed by a blanket deny statement. The Control Plane set for an interface i on a router d is the union of all packets that are routed out of interface i to various destinations, and this is a function of d's FIB entries. Note that the control plane set can also be represented as an ACL, i.e. a sequence of permit statements one corresponding to each FIB entry, followed by a blanket deny statement.

Using the control and data plane sets defined per interface on $path(R1, R2)$, the router-level reachability set between $R1$ and $R2$, or $\mathcal{RRS}(R1, R2)$, can be obtained as the intersection of the control and data plane sets for the routers and interfaces in $path(R1, R2)$. Given the ACL representations of each set above, intersections are easy to compute because the intersection of two ACLs is simply the intersection of each pair of rules in the two ACLs. It follows that the final router reachability set is also represented in ACL form.

2.2.2 Subnet-Level Reachability Sets (SRS)

The routes obtained above only yield binary information regarding the reachability between the subnets attached to a pair of source, destination routers: if a route exists to a destination subnet, then the above assumes that all subnets attached to the source router can have unfettered access to the destination subnet. However, even if a route exists, a subnet may not have unfettered access due to data-plane restrictions at the interfaces connecting the subnets.[1]

To model reachability more accurately, from the router-level reachability sets we obtain the exact subset of packets that can be exchanged between a given pair of subnets. We refer to this as the *subnet-level reachability set*. To compute the subnet-level reachability set, $\mathcal{SRS}(s1, s2)$, between subnets $s1, s2$ attached to routers $R1$ and $R2$ respectively, we first calculate the set of packets that $s1$ can send into the network given limitations created by data plane ACLs (in-bound ACLs) on the interface connecting $s1$ to its gateway router $R1$; we call this $entry(s1)$. Second, we similarly define the $exit(s2)$ as the set of packets the network will deliver to $s2$ given the limitations created by the ACLs (out-bound ACLs) on the interface connecting the subnet to router $R2$.

$\mathcal{SRS}(s1, s2)$ is then the intersection of $entry(s1)$, $exit(s1)$ and $\mathcal{RRS}(R1, R2)$. As before $\mathcal{SRS}(s1, s2)$ can be computed using ACL intersections, first between $entry(s1)$ and $exit(s1)$, and then between the result of the previous intersection and $\mathcal{RRS}(R1, R2)$.

Suppose a network owns n subnets. Using the above approach, we can derive subnet-level reachability sets between every pair of subnets and the n^2 such sets can be represented using an $n \times n$ matrix. We refer to this as the *SRS matrix*.

An example of an *SRS matrix* for a hypothetical network with three subnets is shown and explained in Figure 2. In this example, there are 9 subnet-level reachability sets that can be interpreted as follows. Consider the SRS for packets originating at subnet A (column 1) and destined for subnet B (row 2). The first half of subnet A has unfettered access to subnet B. In contrast the second half of subnet A can only communicate with the bottom half of subnet B. Unlike the other two subnets, subnet C has unfettered access to all subnets in the enterprise (col 3).

[1] Prior work doesn't examine data-plane filters on the interfaces connecting the subnets, it only examines filters on the interfaces along the path between the gateway routers

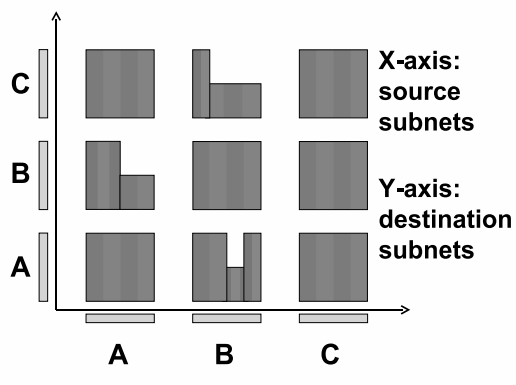


Figure 2: An SRS matrix for a network with three subnets. For this example, the constraints placed by the network apply only to the source and destination IPs.

Since we use ACLs to represent the reachability sets, each SRS in Figure 2 is represented by one ACL with multiple rules in it. For instance, the SRS between A and B (the set in col 1, row 2) could be represented by an ACL with two rules, one covering the reachability between the first half of the IPs in A and all of B, and the other covering that between the second half of A and the lower half of B.[2]

2.2.3 Policy Unit Extraction

Using the formal definition from Section 2.1, we seek to find sets of IP addresses $\mathcal{H}$ such that each set is as large as possible, the sets partition the space of all IP addresses in the network, and for each IP address in set $\mathcal{H}$ the values of $Pow(\hat{\mathcal{H}} \times \mathcal{C}_1 \times ... \times \mathcal{C}_m \times \mathcal{A})$ that map to it are identical.

Identifying the maximal sets $\mathcal{H}$ is made harder by the fact that the boundaries of $\mathcal{H}$ will most likely not align with the subnet boundaries in the network. For instance, in the example above (Figure 2), the first and second halves of subnet A differ in the constraints that the network imposes on them, and hence they belong to different policy units. In contrast, the network treats the first half of subnet A identical to the (disjoint) subnet C and hence they should belong to the same policy unit. As a result, a policy unit may consist of multiple subnets, and a subnet may contain several policy units.

We propose a *geometric heuristic* to identify the exact granularity at which policies are defined within a subnet. For ease of description, we assume below that constraints apply only on source and destination IP addresses on packets, and none apply on ports/protocols, but our heuristic can be easily generalized to also consider ports/protocols. We present an intuitive description. A formal specification of our heuristic can be found in Figure 3 where we assume that the SRSes are represented using ACLs.[3]

Intuitively, we can extract policy units from the SRS matrix obtained above by identifying groups of source addresses for which the subnet-level reachability sets are equivalent. In particular, we decompose the subnet-level reachability sets corresponding to a source subnet (i.e. each column in Figure 2), into "rectangles" such that that all the source addresses in the x-dimension of a rectangle can

[2] In practice, we may end up using many more rules to represent these sets; however, we employ ACL optimization techniques [1, 6] to result in ACLs with the fewest number of non-overlapping rules.

[3] Our current implementation optimizes the ACLs first to improve the run-time and space requirements of this heuristic.

```
EXTRACTPOLICYUNITS(SRSMatrix)
    // Assume each SRS is represented using an ACL
 1  Initiate lists src_ticks and dst_ticks to empty.
 2  for each subnet-level reachability set srs ∈ SRSmatrix
 3     for each rule rule ∈ srs
 4        Populate src_ticks with start and end source IP address in rule.
 5        Order the addresses in src_ticks in ascending numeric order.
 6        Populate dst_ticks with start and end destination IP address in rule.
 7        Order the addresses in src_ticks in ascending numeric order.
    // Create subunitmatrix from the src and dst ticks.
    // Initially, the matrix is empty.
 8  subunitmatrix ← MATRIX(||src_ticks || by || dst_ticks||)
 9  for each element (s_i, d_j) in subunitmatrix
10     Set a 1 if there is a rule in any SRS such that the space of packets it permits...
       ... supersedes the space of packets in (s_i, s_{i+1}) × (d_j, d_{j+1}).
    // All subunits in a bucket have identical reachability and
    // can be coalesced together into a policy unit
11  for each s_i ∈ the first dimension of subunitmatrix
12     keystring ← HASH(d_j's for which (s_i, d_j) = 1).
13     Insert the subunit into a hashtable using keystring.
14  Coalesce all subunits in a bucket into a policy unit.
```

Figure 3: Pseudocode for extracting policy units from subnet reachability matrix, assuming each SRS is represented using an ACL.

reach exactly the same set of destination addresses. Thus, each rectangle is the reachability set for a contiguous sequence of IP addresses within a source subnet that have identical reachability to the rest of the network. Examples of the rectangles are shown in Figure 4.

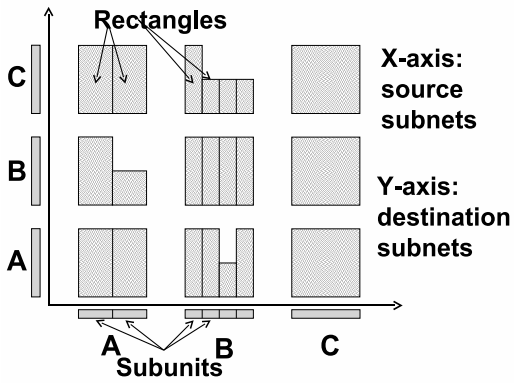


Figure 4: The rectangles and policy subunits after the subnet-level reachability sets have been decomposed.

We refer to the projections of the rectangles unto the x-axis (i.e., the source subnets) as "policy subunits", also shown in Figure 4. In other words, each policy subunit is a *maximal* contiguous sequence of addresses in the source subnet — all of which can reach the same portions of the destination subnets. In Figure 3, the source IP address ranges extracted at lines 4 and 5 make up the network's policy subunits.

This decomposition step increases the number of columns (and rows) in the SRS matrix transforming it into a $subunitmatrix$. In this $subunitmatrix$, each column corresponds to a policy subunit within a subnet. For instance, from three columns in Figure 2, we reach a decomposed $subunitmatrix$ with 7 columns shown in Figure 4. The single column corresponding to source subnet B in Figure 2 is now decomposed into four columns[4]. We show how the $subunitmatrix$ is created in lines 8–10 of Figure 3.

To extract the policy units from the decomposed version of the SRS matrix, we simply coalesce the policy subunits that have the same reachability profile to the rest of the network, i.e. they can reach the same set of destination addresses. This is specified more formally in lines 11–14 in Figure 3.

In Figure 5, we see that the first policy subunit within source subnet A, the first policy subunit within source subnet B and the only policy subunit for source subnet C all have the same profile and hence get coalesced into a single policy unit. Upon applying the above approach to the example in Figure 2, we see that the network has four policy units; policy unit 1 is shown at the bottom in Figure 5.

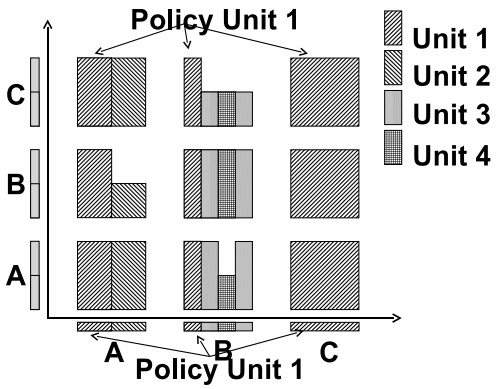


Figure 5: The final policy units

3. EMPIRICAL STUDY

In this section, we apply our technique for mining policy units to router configuration data obtained from five production networks. We study two private corporate networks (which we call Enet-1 and Enet-2) and three university networks (Univ-1, Univ-2, and Univ-3). In all five networks, communication between hosts is restricted using a mix of control and data plane mechanisms, implemented at both layer-3 and layer-4. Three networks employed VLANs. The goals of this study are: (1) to validate policy units can be automatically extracted from network configurations: (2) that policy units reveal useful information about the policies a network implements, and (3) to catalog patterns employed by enterprise networks in defining their network-wide policy. Our observations also indicate how operators can use policy units when reasoning about network configuration.

High-level properties of networks and their policies. Table 1 summarizes the properties of the networks we examined. The networks differ significantly in the number of routers they use, the number of allocated subnets, and the number of IP addresses. Absent more accurate information from operators, we use the number of IP addresses as an admitedly rough indication of the number of hosts in the network.

The table also shows the number of policy units that our framework identified in each network. Several important observations can be made from this. First, the number of policy units is much less than the number of subnets, indicating that policy units are able to concisely summarize the reachability policies of the network.

Second, there is no obvious correlation between the number of

[4]Our heuristic may cause the number of rows to also increase; this depends on how the rules representing each SRS partition the space of destination addresses, or the composition of dst_{ticks} in Figure 3

Name	# Routers	# Subnets	# Policy units	# Addresses
Univ-1	12	942	2	100K
Univ-2	24	869	2	130K
Univ-3	24	617	15	100K
Enet-1	7	98	1	20K
Enet-2	83	142	40	24K

Table 1: Properties of the 5 networks studied, including the number of policy units found in each.

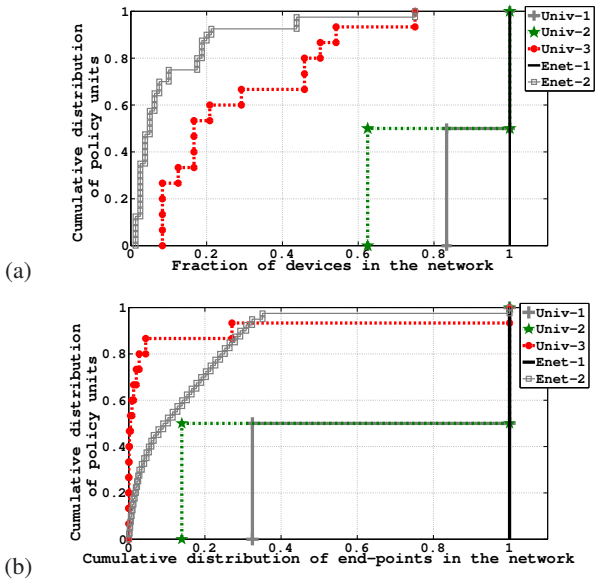


Figure 6: Structure of policies in the networks studied. Figure (a) is a CDF of policy units as a fraction of the network devices spanned. Figure (b) shows the cumulative fraction of end-points in policy units (as a CDF).

policy units and the network size, in terms of the number of end-points or subnets. In particular, the largest network in terms of the number of end-points (Univ-2) has just two policy units, and one of the smallest networks (Enet-2) has 40 policy units. This indicates the diversity of reachability policies that modern enterprise employ and that reachability policy is driven by network mission, not size.

Classes of Networks. From Table 1, we see that the networks cluster into two groups: *policy-lite* networks (Univ-1, Univ-2, Enet-1) that implement 1 or 2 policy units and *policy-heavy* networks (Univ-3, Enet-2) that implement 14 and 40 policy units respectively. We delve deeper into the policies in these networks below.

Structure of policies. In Figure 6, we examine the structure of policy units inside each network. We study how many end-points participate in each policy unit and how the end-points in each unit are spread across the network.

We say a policy unit "spans" a router if any of the end-points that are part of the policy unit are connected directly to that router. We observe from Figure 6(a) that in the policy-lite networks, each policy unit spans over 50% of the routers in the network. We also notice that in these networks, at least of one the policy units spans all routers. However, in policy-heavy networks, about 60% of the policy units each span 20% or fewer routers in the network. Also, we find that none of the units in the policy-heavy networks span all routers. This illustrates how the policy units expose the compartmentalization imposed by the network, with hosts connected to different parts of the network being subject to different policies and able to reach different sets of end-points.

In figure 6(b) we examine the cumulative fraction of end-points in the networks' policy units. We find that all networks have at least

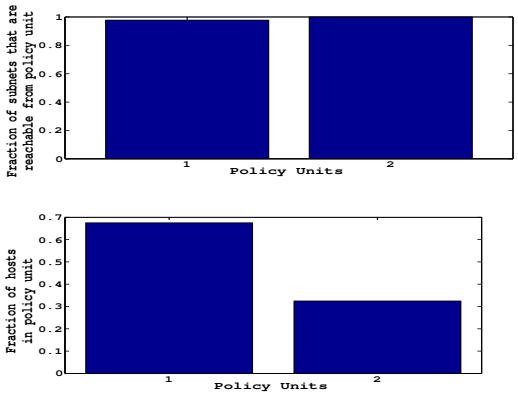


Figure 7: Policy in Univ-1 is "on by default," with 70% of hosts able to reach nearly all subnets and 30% able to reach all subnets.

one unit to which over 60% of the end-points in the network belong. This means that most end-points in all the networks we studied are subject to identical policies. Furthermore, for all networks, most end-points members of only a handful of units. For instance, we observe that about 70% of the end-points in the policy-heavy networks (Univ-3 and Enet-2) fall into 15% of the units (this corresponds to 2 and 4 policy units in Univ-3 and Enet-2, respectively). Likewise, in the policy-lite networks, we find that all end-points are members of one or two policy units. This suggests that policy is unevenly applied in the networks we studied. Most hosts experience the same policies, but certain special hosts are selected for extra constraints or extra freedom.

Next, we now dig deeper into the the policy units for a policy-lite and a policy-heavy network. In particular, we focus on the differences among the policy units and identify key properties of the reachability policies that the networks are trying to implement.

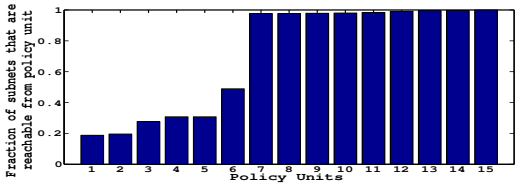
Univ-1. In Figure 7 (a) we examine the differences among the policy units in Univ-1 in terms of the destinations they can reach. Recall that Univ-1, which is a policy-lite network, divides its end-points into two policy units. Of these, hosts in one unit (policy unit 2 on the right) are able to reach all subnets in the network, while those in the other (unit 1) are unable to reach about 2% of the subnets in the network. From Figure (b), we note that unit-1 is comprised of 70% of the network's end-points, while the remaining 30% are in unit 2.

We were able to discuss our empirical observations with the network's operators. The operators validated that the network indeed implements two such units. In particular, the network controls the reachability to a specific collection of subnets (all of which were attached to a given router) by preventing route announcements to the subnets from reaching other routers (this is achieved using the appropriate route filters). This small collection of subnets, however, were reachable from the rest of the subnets attached to the router.

On the whole these observations indicate that Univ-1 implements a fairly uniform "on by default" policy across all end-points, meaning that hosts can access almost all network resources unless prevented by some explicit mechanism.

Univ-3. Univ-3 provides a more complex case study, with 15 distinct policy units. The results from our study of this network are summarized in Figure 8. From Figure 8 (a), we observe that 9 of the 15 units in this network, units 7 through 15, have almost complete reachability to the rest of the subnets in the network, with each unit

(a)



(b)

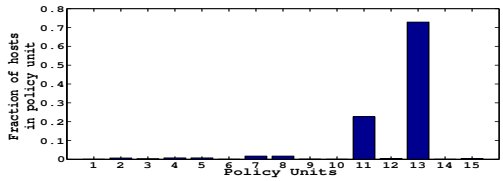


Figure 8: Policy in Univ-3 places most hosts into "default on" units 7-15, and a few hosts into "default off" units.

being able to reach at least 98% of the subnets in the network. In contrast, units 1 through 6 have very limited reachability, being able to reach only between 20 and 45% of the subnets in the network.

In Figure 8 (b), we illustrate the distributions of end-points across various policy units. The policy units 7 through 15, all of which have roughly universal reachability, vary significantly in the number of end-points in them: unit 13 contains 70% of the end-points, unit 11 contains 20% of the end-points, while units 9, 10, 12, 14 and 15 contain a miniscule fraction of end-points each. In contrast, the total number of end-points in policy units 1 through 6, which can reach a much smaller fraction of network resources each, is $< 5\%$.

These results show that the policy implemented by Univ-3 is characterized by an interesting dichotomy. The policy divides the network into two unequal parts. One part, which contains an overwhelming fraction of end-points is "on by default", meaning that hosts in this part can access almost all network resources. A few special cases restrict the specific set of resources that small collections of hosts can access. In the other part of the network, which contains a much smaller fraction of end-points, the policy is closer to being "off by default", meaning that the hosts cannot access most network resources by default. Upon examining the configuration files, we noticed that Univ-3 used data plane mechanisms (i.e. packet filters) to implement the special cases in the first part. However, to implement the off-by-default access policy in the second part, the network used both packet filters as well as control plane mechanisms such as route filters.

.

4. APPLICATION TO NETWORK MANAGEMENT

Below, we provide a brief outline of two network management tasks where policy units derived from the network configuration state may prove valuable.

1. Making informed changes to configuration. When installing or altering configuration in large networks, operators have no way to systematically reason about how their changes may impact the network's policies. Operators employ a combination of ad hoc reasoning, designing time -consuming management of change (MOC) reports or waiting for users' complaints or even attacks, before determining that their changes have had an undesirable interaction with the intended network-wide policies. Operators can use policy units to validate changes by comparing policy units before and after the change is made. Operators can use the "diffs" between the two states to debug the network change.

2. Examining trends in network policy evolution. Over time networks grows in size and complexity, making it difficult for operators to determine if the overall policies can be implemented using simpler mechanisms. For instance, a network may start out with conservative communication rules requiring a multitude of policy units, but over time, more and more end-points are granted a uniform set of privileges to access a common set of resources. Although the configuration becomes complex, since more and more end-points are granted a common set of privileges, the network's policy units may coalesce and *shrink* in total number. By monitoring policy units, a network operator can examine how the network-wide policy evolves and whether, at some point, the number of units and the nature of the PPS naturally lend themselves to an alternate, much simpler network-wide configuration.

5. RELATED WORK

Prior work [12, 3] developed approaches for modeling the reachability between routers. Our approach builds on these proposals, but it coalesces and extracts the high-level network policies implemented in a network configuration. The policy units we extract are a natural match with how operators view and design their networks, while the reachability sets between routers computed by prior work do not expose the commonality or structure of the policies applied to hosts. Second, network modifications, such as movement of department across buildings or movement of filters from core to edge, can greatly affect a network's reachability sets. However, the policy units remain unchanged as the constraints applied to the network's hosts remain constant.

Several projects [2, 5, 13] attempt to simplify network management by using clean-slate approaches to represent and implement global policies. Policy units can be used to unearth the structure of policies in an existing network that need to implemented in the clean-slate design, and help select between "default on" and "default off" strategies.

A few studies [9, 11] examined traffic characteristics in enterprises, and other recent work developed models for design and configuration in enterprises [3, 7]. Our work adds to these by shedding light on the nature of reachability policies in enterprises.

6. SUMMARY

While there has been a growing interest in understanding the design and operation of enterprise networks, few studies, if any, have examined the nature of reachability policies implemented by enterprises today. In this paper, we introduced the notion of policy units that form an abstract representation of how enterprise reachability policies segregate end-points into distinct privilege classes. We presented an initial algorithm for extracting policy units from router configurations. We applied the algorithm to five production networks and verified our observations with the operators of some of the networks. Through our study, we obtained unique insights into the current implementation of reachability policies in enterprises. In particular, we found that most hosts in these networks are subjected to a uniform set of reachability policies, while a few special case hosts have very restricted reachability.

We argued that our empirical observations and the policy unit extraction framework are useful to inform clean-slate approaches, to support network re-design efforts, and to augment current approaches to network configuration and management. We expect policy units will be valuable as an aid for visualizing the network-wide configuration state, and are exploring these directions.

7. REFERENCES

[1] S. Acharya, J. Wang, Z. Ge, T. Znati, and A. Greenberg. Simulation study of firewalls to aid improved performance. In

ANSS '06: Proceedings of the 39th annual Symposium on Simulation, pages 18–26, Washington, DC, USA, 2006. IEEE Computer Society.

[2] H. Ballani and P. Francis. CONMan: A Step towards Network Manageability. In *Proc. of ACM SIGCOMM*, 2007.

[3] T. Benson, A. Akella, and D. A. Maltz. Unraveling the complexity of network management. In *NSDI*, April 2009.

[4] M. Casado, M. Friedman, J. Pettitt, N. Mckeown, and S. Shenker. Ethane: Taking Control of the Enterprise. In *SIGCOMM '07*.

[5] M. Casado, T. Garfinkel, A. Akella, M. Friedman, D. Boneh, N. Mckeown, and S. Shenker. SANE: A Protection Architecture for Enterprise Networks. In *USENIX Security*, Vancouver, BC, Canada, Aug. 2006.

[6] A. Feldmann and S. Muthukrishnan. Tradeoffs for packet classification. In *INFOCOM*, 2000.

[7] P. Garimella, Y.-W. E. Sung, N. Zhang, and S. Rao. Characterizing VLAN usage in an operational network. In *INM '07*.

[8] A. Greenberg, G. Hjalmtysson, D. A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. A clean slate 4d approach to network control and management. *SIGCOMM Comput. Commun. Rev.*, 35(5):41–54, 2005.

[9] S. Guha, J. Chandrashekar, N. Taft, and K. Papagiannaki. How healthy are today's enterprise networks? In *IMC*, 2008.

[10] D. A. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjálmtýsson, and A. Greenberg. Routing design in operational networks: a look from the inside. *SIGCOMM Comput. Commun. Rev.*, 34(4):27–40, 2004.

[11] R. Pang, M. Allman, M. Bennett, J. Lee, V. Paxson, and B. Tierney. A first look at modern enterprise traffic. In *IMC*, 2005.

[12] G. G. Xie, J. Zhan, D. A. Maltz, H. Zhang, A. G. Greenberg, G. Hjálmtýsson, and J. Rexford. On static reachability analysis of ip networks. In *INFOCOM*, pages 2170–2183. IEEE, 2005.

[13] H. Yan, D. A. Maltz, T. S. E. Ng, H. Gogineni, H. Zhang, and Z. Cai. Tesseract: A 4d network control plane. In *NSDI*. USENIX, 2007.