Longitudinal Analysis of Network Complexity

Fatemah Panahi
Department of Computer Science, University of Wisconsin-Madison
fatemeh@cs.wisc.edu

May 19, 2010

Abstract

Managing an enterprise network's configuration is an error-prone and tedious task. As the enterprises are getting more complex over time, we should be able to identify the factors that contribute to networks getting complex to be able to avoid or mitigate them. Building on our previous work where we define different metrics that can help us interpret the complexity of the networks, we analyze how these complexity metrics of an enterprise network changes over time. These metrics include *Router Roles*, *Referential Complexity*, *Routing Instances*, and *Inherent Complexity*. We also analyze the behavior of individual roles, referential graphs, and routing instances that make these metrics in an effort to find the underlying reason for the change in the complexity metrics over time.

1 Introduction

Network configuration in enterprises is complex, and changes frequently [5]. Moreover, the configuration files are hard to deal with since they are written in low-level languages and there is not much automation in making configuration changes. Also, many files refer to each other and need to be changed in correlation with each other. Therefore, managing network configuration in enterprises is an error-prone and tedious task for the operators. As the complexity of the network increases, dealing with and detecting the errors in these files gets an even larger problem for the operators. It is imperative for us to try to understand why what a complex network is at the first place and then to understand what factors contribute to this complexity. This will help us to design less complex networks, and avoid networks getting more complex unnecessarily.

We build on our previous work where we identified 4 types of complexity metrics and analyzed different networks' complexity based on these metrics [ref]. In this work we try to understand how the network configuration changes over time which will help us to understand what factors contribute to the complexity in networks.

We first define complexity. We define complexity as how hard it is to manage a network. The more complex network configurations are harder to manage by the operators. The complexity metrics that we focus on in this work are as follows:

- Router Roles: Each router plays a role in the network. Routers that have similar configurations are defined to have the same role. As the number of roles increases, the network gets more complex to manage.
- Referential Complexity: Certain configurations span multiple devices or refer to another stanza in the same configuration file (Intra and Iner-file references). The more references between the devices or within a device's configuration file, the harder it is to make a change in the network configuration. The reason is that more steps are involved in each change and thus the network is more complex.
- Routing Instances: As the number of routing instances implemented in a network increases, the network gets more complex.
- Inherent Complexity: This metric analyzes the impact of reachability and access control policy on network complexity. Configurations with sophisticated policies which enable access between a set of hosts and deny access between others are harder to manage.

The rest of this paper is as follows: First we describe the related works in section 2. We present our implementation details and results in section 3 and 4. Section 5 and 6 are dedicated to discussion and future work. We conclude with section 7.

2 Related Work

Researchers have analyzed the configuration issues in networks from different angles. Works such as [1, 2] have focused on diagnosing errors and issues in the network. [1] proposed a tool called NetMedic which diagnoses issues as fine-grained as a firewall misconfiguration. [2] more specifically deals with detection of ineffective router configurations. These configurations can be redundant and make the files unnecessarily more complex, or can actually cause unintended network behaviors. What we try in this work is to find the underlying factors that cause this complexity in the network. These works might in fact reduce the complexity as some bugs add to the complexity in the network; as we found in our previous work [6]. However, our work is mostly considered a preventive attempt while these works focus on curing the issues in the network configuration.

Another group of research is focused on extracting information from the current configuration files of different networks. Accepting the fact that the network configuration files and their relationship to each other are very complex, these works make their best effort to understand what is lying under this data. [7] focuses on Virtual Private Networks (VPNs). Looking at the configuration files and how they have changed over time, they perform an analysis of correlated changes in the network in different levels: System-wide changes across the routers, and changes inside a router's configuration file. Understanding these correlations can help us develop monitoring tools for the network to detect erroneous changes. This work is very similar to our effort. We differ in that we focus on an enterprise network and that we aim to understand how the *complexity* changes over time rather than identifying the correlated changes.

On the same line of research, [3] tries to extract the underlying policies in the network configuration. As networks grow and get more complex, it is even not clear what parts of network are accessible from other sections of the network or from the outside. This work tries to find the reachability policies in a network from the static configuration files on hand. This can improve the understanding of a network at any state and help fix any misconfiguration or unintended behavior.

Our previous work [6] tries to define and quantify the complexity of a network. We propose an automated approach for understanding the underlying complexity. This work is performed using the metrics and tools that [6] uses. Thus, this paper is groundwork for our measurement study as it identifies the complexity metrics in a network.

On the other extreme, works such as [4], try to rethink the network management, and propose architectures that are easier to manage. These works leverage the data provided by the previously mentioned bottom-up approaches to take a top-down look at the network architecture and answer the question: *How can we design the enterprise network architecture to make it more manageable?*. Our effort would provide food for thought for such a design by identifying the causes of complexity in a network.

3 Implementation

The results described in the next section are captured by a series of Perl scripts that go through each snapshot of the network and analyze different aspects of it. The code is embedded in the ConfCap tool that was developed in [6]. Thus, the scripts can be combined with the ConfCap tool that we currently use for getting the different comlexity metrics in the network, and can be readily applied to any other network.

4 Results

The data for our analysis is the configuration files from the University of Wisconsin-Madison campus network. We analyze 52 snapshots of this network from 2005-01 to 2009-04 (one snapshot per month).In this section we first describe general observations of the network, and then focus on analyzing the changes in the 4 complexity metrics introduced in the previous sections over time.

4.1 General observations

Our general observations are 3 fold:

1. Total number of devices
2. Average configuration file size
3. Stanza types

As shown in figure 1 total number of devices that are in the network increase over time. There are total number of 15 devices at the end of this period. This figure also shows the average size of configuration files for each snapshot. The mean and median for the average size are both about 55 Kbs.

An interesting observation is that the average file size generally increases over time, but whenever there has been an increase in the number of devices we see a decrease in the average file sizes. This might be due to the fact that the new device has a relatively simple configuration file at first which brings down the average. The overall increasing trend might be because usually stanzas are not deleted from these files and only new ones are added. We found in [6] that not all the ACLs and interfaces that are defined are actually used. This means that many times simply new stanzas are added to the file without deleting the unused ones. However, the abnormal sudden decreasing trend is correlated with sudden addition of few core routers, but the exact reason beyond this trend should be further investigated.

It is intuitive for most people that with the increase in the configuration file size and the number of devices the network should get more complex. However, as we will describe in the following section, our data suggests that complexity metrics are not directly correlated with these network characteristics. Therefore, it is important for us

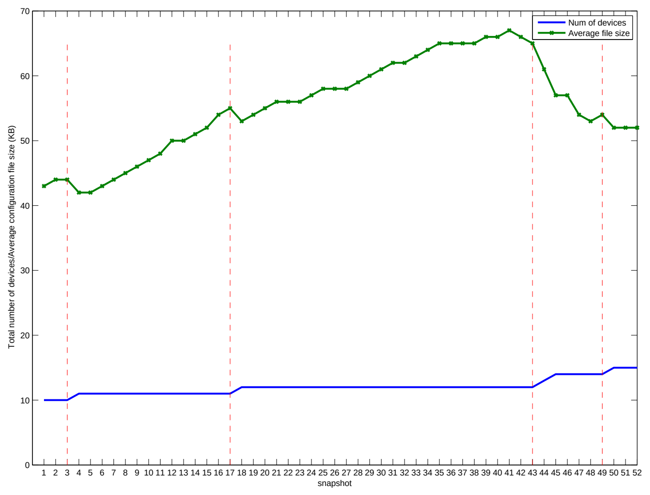

Figure 1: Top: Average configuration file size over time/ Bottom: Total number of devices

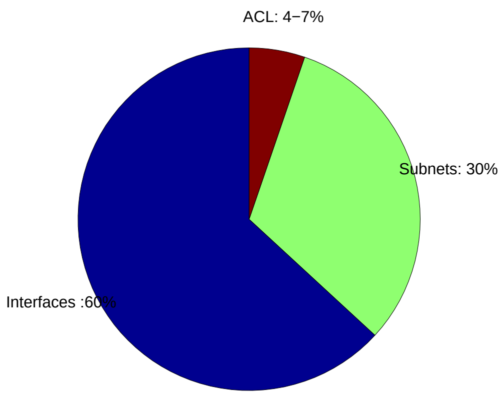

Figure 2: Distribution of types of stanzas in the configuration files

to find accurate complexity metrics and understand the underlying factors for creating the complexity.

In terms of the content of the configuration files we analyze the stanza types. As shown in figure 2 in all snapshots about 60% of the stanzas are of type *Interfaces*, 30% are of type *subnets*, and 4-7% are *ACL* or *Extended ACL*. The other stanza types constitute a negligible part of the configuration files.

4.2 Router Roles

We analyze 2 metrics that were identified in [6] for this section:

1. Total number of roles.

2. Median number of devices that fit in each role.

These metrics are interpreted as follows: The more roles in the network, the more complex the network is. The higher the median number of devices in each role is the less the complexity in network is (means that more devices fit in the same role).

Figure 3 shows these metrics over time. The total number of roles tends to increase over time, thus we can conclude that the complexity of the network increases. One reason for such a behavior is that the network gets fine-tuned over time. For example, at first all departments follow the same policies, but as the time passes by they make requests for policies that are specific to their department. This fine-tuning of the policies causes increase in the number of roles as the same role cannot be applied to all the aspects of the network. Ideally, we could have only one role which would have aspects that we can pick and choose for addressing different requests. Since the demand for policies keep changing and not defined upfront, this requires periodic update of the configuration files to follow the same ideal role mentioned above. This consistency will make managing the network easier for the operators.

Note that the intuition that complexity correlates with the number of devices or the average configuration file size does not hold for this metric. There are periods when the number of devices remain unchanged, but the total roles in the network keeps growing. More over, there are periods where the number of roles increases while the average configuration file size decreases.

Another step further, we looked at individual roles and how they change over time. Figure 4 shows 4 types of analysis. They are as follows from top to bottom:

- Percent of the roles in the snapshot that their stanzas move together to another role in the next snapshot. This means that they are not scattered across different roles in the next snapshot.

- Percent of the roles in the snapshot that remain unchanged and move to the next snapshot.

- Percent of the roles in the snapshot that are newly added to the network. None of the stanzas in these roles existed before.

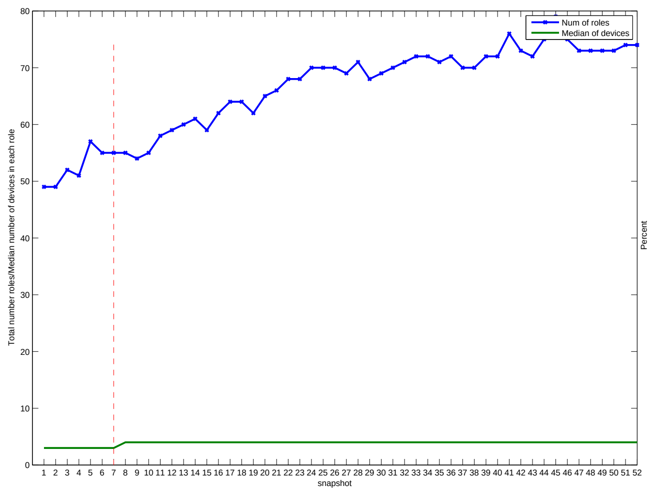

Figure 3: Top: Total number of roles/ Bottom: Median number of devices in each role

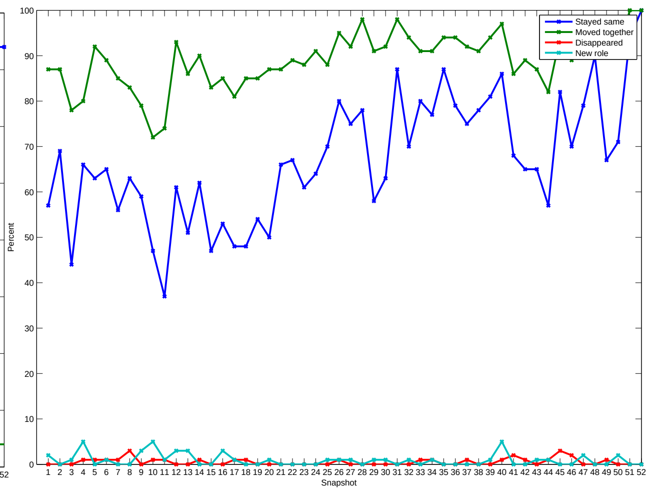

Figure 4: Analysis of changes in roles

- Percent of the roles in the snapshot that all of their stanzas completely disappear in the next snapshot. This role is defined to have disappeared.

As the figure shows, roles are changing about 50% of the time from one snapshot to the next. Changes in the roles does not necessarily correlate with increase in the complexity. By this we mean that we do not see a great jump in the complexity (such as 50%). The reason is that as you can see in the top line in the graph, for most of the roles (about 85% of them) their stanzas move together to the next snapshot. So even though roles might change, these changes do not always lead to addition of new roles.

Also, we can see that there are few roles that their stanzas are totally new, or that their stanzas totally disappear from the snapshot. Interestingly, there are periods where no completely new roles are added, but the total number of templates have increased. We can conclude that the increase in the total number of roles is due to mix and matching the existing stanzas in the previous snapshot and not due to adding totally new stanzas. Our statistics show that about 10% of the roles in each snapshot are made up of a composition of stanzas from different templates in the previous snapshot. An example for this would be a policy change that certain section of the departments such as the libraries should be treated differently from other parts. This might cause a split in a role where similar stanzas are moved into a new role which is only different in part with the role that it was separated from. This increases the total number of roles without adding a completely new role since the stanzas are not completely new.

When discussing roles in the network, the first set of roles that come in mind are ACLs. ACLs are one of the most important set of policies that get implemented in the network, and can be intuitively divided into different roles. For example, one role would be denying access from outside the network to certain parts of the network. Therefore, we analyzed the same metrics a second time using the ACL only stanzas. The results are shown in figure 5. The trend seems to be similar to the previous figure.

4.3 Referential Complexity

The metric regarding referential complexity is the average number of links in each snapshot. Figure 6 shows how this metric changes over time. As the dashed line shows, the average number of links is decreasing over time, with periods where there is some fluctuation in the complexity but it is always around the same range. Reduced number of links means that operators need less steps in performing configuration changes and thus the network is less complex to manage for them. Similar to the roles analysis, we analyze how the referential graphs change over time in figure 7. The graphs in the figure are as follows from top to bottom:

- Percent of the referential graphs in the snapshot that their stanzas move together to another graph in the next snapshot. This means that they are not scattered across different graphs in the next snapshot.
- Percent of the referential graphs in the snapshot that remain unchanged and move to the next snapshot.

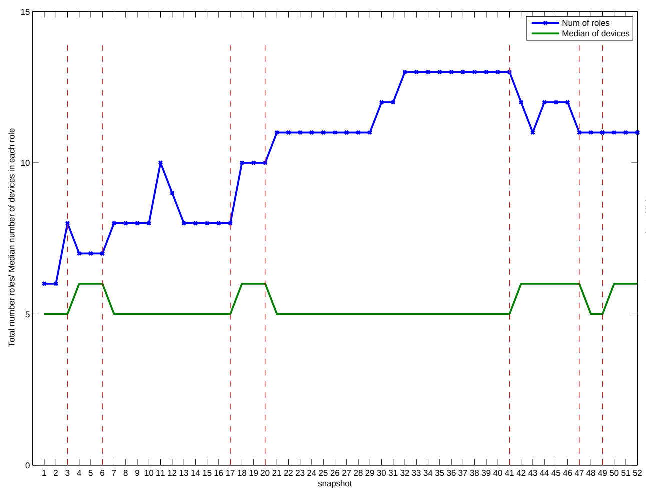

Figure 5: Roles metrics for ACL only stanzas: Total number of roles, Median number of devices in each role

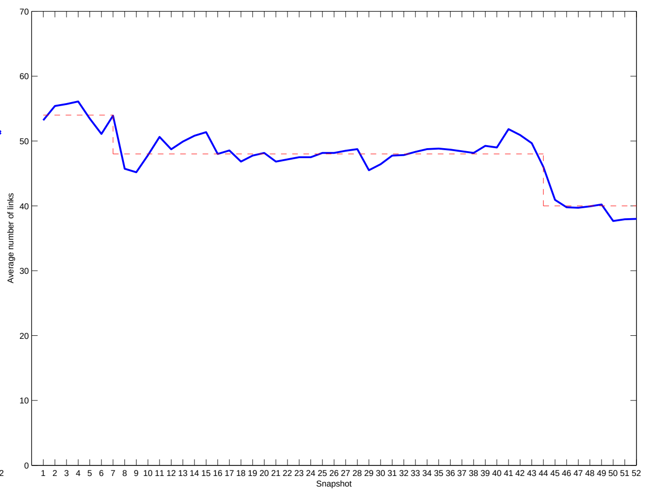

Figure 6: Average number of links over time

- Percent of the referential graphs in the snapshot that are newly added to the network. None of the stanzas in these graphs existed before.

- Percent of the referential graphs in the snapshot that all of their stanzas completely disappear in the next snapshot. This graph is defined to have disappeared.

Compared to the roles, referential graphs change more frequently. Only about 30% of the graphs stay exactly the same and transfer to the next snapshot. Moreover, for only about 50% of the graphs their stanzas remain together and transfer to one graph in the next snapshot. Our statistics show that about 50% of the templates in each snapshot is made of composition of stanzas from different graphs in the previous snapshot. Again, not many graphs are newly added or completely disappeared.

The reason for frequent change is that inter and intra-file references are by definition easier to change and sometimes depend on how the operators write or modify the configuration files. Whereas changes in the roles are design changes in the network which are not as frequent. Also, some changes in the configuration files are not that critical. For example, operators might choose to leave unused stanzas in the files or delete them. This will change the number of references but the overall effect on the network will be the same.

4.4 Routing Instances

The next complexity metric that we analyze is the total number of routing instances. Figure 8 shows how the complexity changes over time. An interesting observation that we can make from this figure is that, as the dashed lines show, all the major changes in the number of routing instances are at the end of the academic year or during the summer time. The reason is that in order to change routing instances in the network, we need to change the control plane of the routers and this change is harder to make, and in case something goes wrong will have a greater effect. Therefore, these changes are planned during the summer when the load on the network is lower.

We have also analyzed how individual routing instances change over time in each snapshot. Figure 9 shows similar graph that we saw for the roles and referential graphs. The graphs in the figure are as follows from top to bottom (the top 2 lines overlap):

- Percent of the routing instances in the snapshot that their stanzas move together to another graph in the next snapshot. This means that they are not scattered across different instances in the next snapshot.

- Percent of the routing instances in the snapshot that remain unchanged and move to the next snapshot.

- Percent of the routing instances in the snapshot that are newly added to the network. None of the stanzas in these routing instances existed before.

- Percent of the routing instances in the snapshot that all of their stanzas completely disappear in the next snapshot. This routing instance is defined to have disappeared.

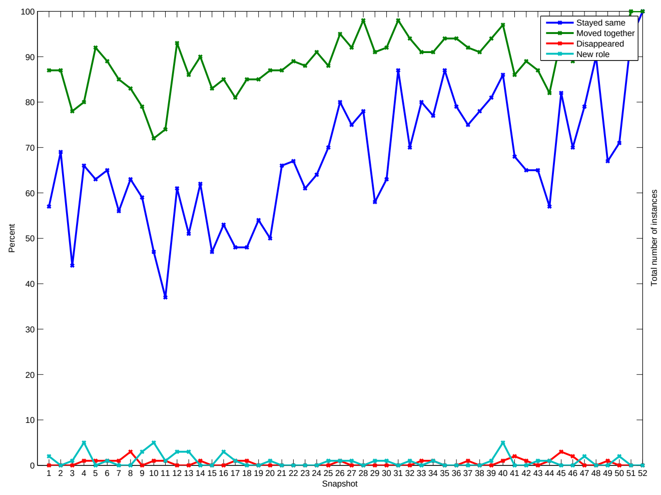

Figure 7: Analysis of changes in referential graphs

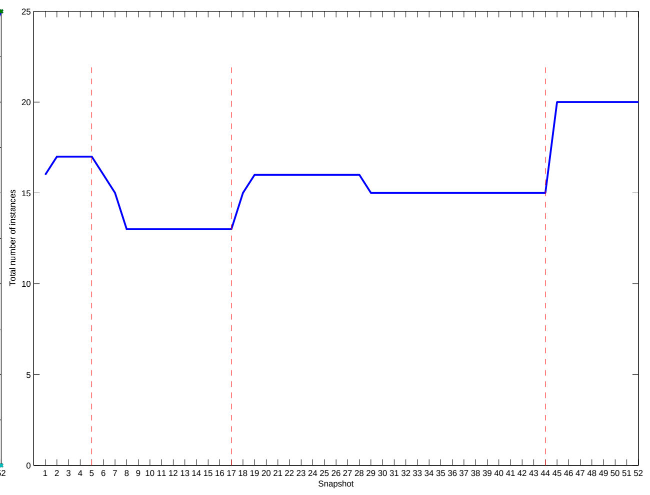

Figure 8: Total number of routing instances over time

The fact that the top 2 lines overlap with each other tells us that all the routing instances that their stanzas end in one routing instance in the next snapshot are the ones that actually stayed exactly the same. This means that for example some of their stanzas did not get deleted. In fact, our statistics show that 0% of the routing instances merge with each other, 0% grow over time, and 0% of them are composed of stanzas from different routing instances in the previous snapshot. This is also captured in figure 9. As the dashed lines show, the changes in the routing instances in snapshots are directly correlated with new routing instances being added, or current routing instances being deleted.

Contrary to the roles and referential graphs where the changes are mostly not related to roles or graphs being added or deleted, the changes in the routing instances are always due to addition of completely new routing instances or complete deletion of existing routing instances. Looking at the routing instances for different snapshots, we found that each of the routing instances consists of one routing process that might span different devices. For example, all of the stanzas in the routing instance have the same type *ospf 10*. This is the reason why the routing instances either get deleted totally or stay the same, and that their stanzas do not move around between different instances.

4.5 Inherent Complexity

Inherent complexity measures how uniform the reachability sets for end-to-end paths are in the network. The metric for inherent complexity is the reachability entropy. The higher the entropy the more complex the network is. This metric is not independent and is related to the number of devices in the network. The ideal entropy for the network of size N is log(N) where the reachability sets between all pairs of routers are identical. The worst entropy is equal to log($N\hat{2}$) where the reachability between each pair of routers is different. This metric is explained in detail in [6]. Figure 10 shows how the entropy changes over time and the max and min values for the entropy for each snapshot.

As the figure shows, unfortunately the entropy that we got for the network over time is over the max limit and lower than the min limit all the time. We understood that there has been a bug in the reachability analysis code that caused this error. Once this issue was identified, there was not enough time to run the code again for the 52 snapshots. Therefore, we are bound to use this data for our analysis.

Looking at the figure, it seems that the network is getting less complex based on this metric. The entropy is decreasing over time and this shows that the reachability policies are getting more uniform. We will further analyze this metric once we have access to more accurate data.

4.6 Other Analysis

4.6.1 Changes grouped by stanza types

Regarding the roles analysis which we saw the metric had frequent changes, we analyzed the types of stanzas that are changing most. The changes are defined as follows: if the stanza appears in one of the roles in one snapshot and disappears in the next snapshot, or if the stanza does

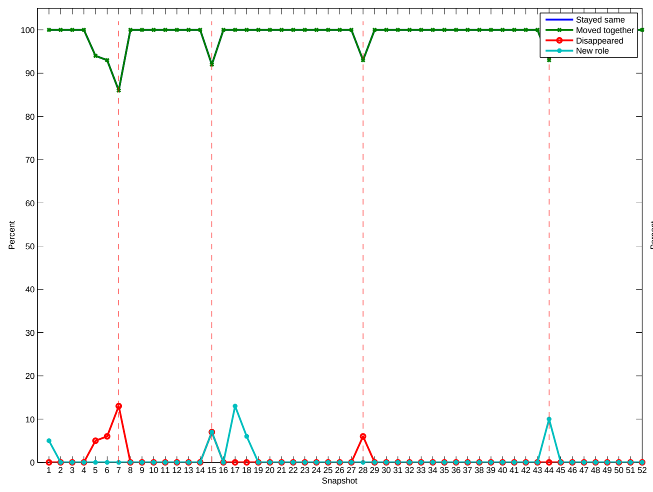

Figure 9: Analysis of changes in routing instances

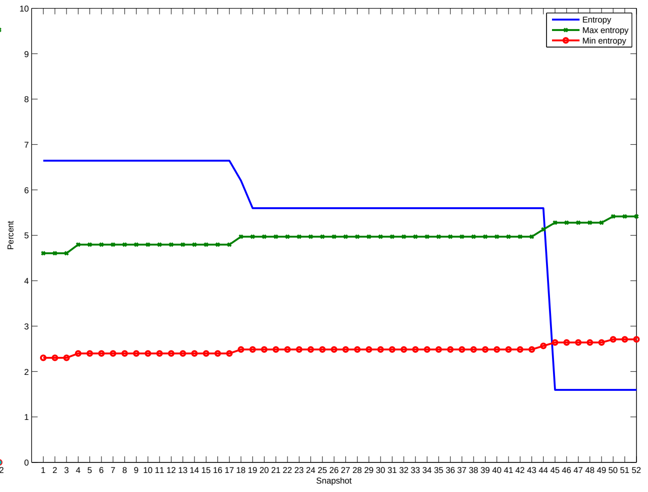

Figure 10: Inherent complexity over time

not appear in one of the roles in the first snapshot, but gets added to a template in the second snapshot. The types of the changing stanzas and their respective percentages are as follows:

- Interfaces 71
- ACL: 21
- ExtendedACL: 4
- StaticRoutes 3
- RipRouting 2

Interfaces and the ACLs contribute to more than 95% of the changes. This is similar to the conclusions of the related work [7] where they found that most of the changes are regarding to interfaces and ACLs are in the top 10 changing types.

4.6.2 Commands involved in changes

We looked deeper to the stanzas that change from one snapshot to the next snapshot and got some statistics regarding the commands that are added or deleted from the stanzas that cause the change in the stanza. The commands were similarly distributed between the types (as shown above). However, the commands regarding interfaces have more variety. The most used command in the interfaces was *switchport trunk* and constitutes 15% of the commands involved in changes. The most used command in the ACLs was *access-list permit* which constitutes 18% of the commands involved in changes. We were hoping to be able to categorize the commands and find some correlation between the commands and changes in the complexity. However, the current data is not conclusive and this remains to be worked on in future work.

5 Discussion

There are certain aspects of this analysis that is not clear at this point. For example, when one complexity metric such as the total number of roles increases, we observed that anther complexity metric such as the referential complexity decreases. The question comes into mind that how do we define the overall complexity of the network? This requires an additional metric that encompasses these metrics to tell us the overall complexity. Moreover, we should be able to weight how important each of the complexity metrics are. This information should be gathered with cooperation with the network operators and cannot be extracted simply from the configuration files.

Also, some metrics such as the total number of roles in the network, tend to change more frequently. On the other hand, major changes in other metrics such as the average links in the referential complexity, total number of routing instances, or the inherent complexity of the network span multiple snapshots. It would be interesting to look into the underlying reasons for such behavior.

6 Future Work

As we have been mentioning over and over again in this report, network configurations are complex. Therefore, it is very hard to find exact correlations between certain changes in the network configuration and the change in the complexity metrics. In the scope of this project, we were not able to identify the specific reason for the change in the complexity for most of the metrics. This requires more understanding of what each stanza/role actually accomplishes in the network and looking more in depth into the configuration files. This is the direction that we will be going in our future work.

In order to accomplish this goal, it would be very helpful to interview some of the operators, show them the results, and get some insight on what could be the major change in the network that is causing a certain increase/decrease in the complexity metrics. For example, regarding the routing instances, we could guess that major changes in the complexity has happened during the summer time. However, these correlations are not as exact for the other metrics.

A logical next step to this analysis is to apply the tool that we have to other networks and to compare the results with each other. This will help us identify if there are any particular trend that every network goes through during its life time in terms of complexity. Moreover, we are interested to see if the same changes that make this network more complex are contributing to other networks getting more complex or not. Identifying this will make operators more aware of what changes will lead to more/less complexity, and will help us develop tools to automatically analyze the network and provide feedback to the operators.

7 Conclusion

We analyzed the complexity metrics defined in [6] over time on an enterprise network. More over, we looked at how different components that contribute to this complexity change from snapshot to snapshot. We were able to identify certain trends in changes in the complexity. More over, the tools that we have developed for analyzing the complexity metrics over time can be applied to other networks to get better insight on how complexity changes. This will help us to get more reliable results by comparing these networks' behaviors.

References

[1] S. Kandula , R. Mahajan , P. Verkaik , S. Agarwal , J. Padhye , and P. Bahl. Detailed diagnosis in enterprise networks. *ACM SIGCOMM Computer Communication Review*, v.39 n.4, October 2009.

[2] S. Lee, T. Wong, and H. Kim. Netpiler: detection of ineffective router configurations. *IEEE Journal on Selected Areas in Communications*, 27(3): 291-301, 2009.

[3] T. Benson, A. Akella, and D. Maltz. Mining policies from enterprise network configuration. *Internet Measurement Conference*, 136-142, 2009.

[4] M. Casado, M. Freedman, J. Pettit, J. Luo, N. Gude, N. McKeown, and S. Shenker. Rethinking enterprise network control. *IEEE/ACM Transactions on Networking*, 17(4):1270-1283, 2009.

[5] X. Sun, Y. Sung, S. Krothapalli, and S. Rao. A systematic approach for evolving VLAN designs. In *INFOCOM*, 2010.

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

[7] Y. Sung, S. Rao, S. Sen, and S. Leggett. Extracting Network-Wide Correlated Changes from Longitudinal Configuration Data. In *PAM*, 111-121, 2009.