Visual Analysis of Complex Firewall Configurations

Florian Mansmann
University of Konstanz
Florian.Mansmann@uni-konstanz.de

Timo Göbel
University of Konstanz
timo.goebel@timogoebel.eu

William Cheswick
ches@cheswick.com

ABSTRACT

Firewalls have become essential components in the security concept of almost any modern computer network. Due to their relevance and central location in the network, their programming logic often survives several generations of administrators and hardware. Understanding the logic behind a firewall configuration is thus an important but challenging task for a network administrator. In general, there is a tendency to add new rules while old rules are only rarely changed or removed due to unexpected consequences in the network. In this paper we present a visualization tool to support the network administrator in this complex task of understanding firewall rule sets and object group definitions. The tool consists of a hierarchical sunburst visualization, which logically groups rules or object groups according to their common characteristics, a color-linked configuration editor and classical tree view components for rules and object groups. All these components are interactively linked to enable both exploratory and hypotheses testing tasks aimed at understanding the complex functionality of a firewall configuration. To verify our design, we present two case studies on the analysis of rule usage and on nested object groups and collected feedback from five firewall administrators.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.3.8 [**Computer Graphics**]: Application; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces

General Terms

Design, Human Factors, Security

Keywords

Firewall Visualization, Rule Set Visualization, Managing Firewalls

VizSec '12 October 15 2012, Seattle, WA, USA
Copyright 2012 ACM 978-1-4503-1413-8/12/10 ...$15.00.

1. INTRODUCTION

Firewalls have become an integral part of the network security infrastructure of almost every managed network. Their goal is to either protect the internal network infrastructure from attacks, theft of data and manipulation or to log security related events at scale. Ever since the inception of firewalls [4], their complexity has dramatically increased not only in functionality, but also in the ways that people use and configure these firewalls.

Due to the novel exploits and the increase of threats in the internet, administrators need to constantly adapt the protection of their network. While most of these changes involve patches of software on individual hosts, some other changes are more effective to be done through a traffic filtering rule on a central firewall. Because of changing personnel and responsibilities as common in many enterprises and institutions, the body of rules of a firewall often grows to an unmanageable size and complexity. Since there are many interdependencies in the network configuration, it is especially hard to change old rules. If I a rule gets deleted, what will break? What part of the business will suddenly have a problem? As a consequence, old rules are often kept in place.

We believe that there is a pressing need to visually support network administrators in their complex task to manage historically grown firewall rule sets. In our preliminary experiments on representing the interlinked information of a firewall rule set we used graph visualizations, but soon realized that ambiguities of linked elements (e.g., TCP appearing in a almost every 2nd rule) were created so that one could not follow a single path of a rule. However, we were convinced that the structural information implicitly contained in the rule set should be made explicit in the visualization and we thus reverted to hierarchical visualization techniques. While Treemaps nicely visualize leave nodes of a hierarchy, they were not appropriate for our case since the more important information is contained in the structure of the hierarchy. Therefore, we decided to use a hierarchical visualization technique called Sunburst [18] to present firewall rulesets. Besides adapting this technique to group firewall rules based on their common properties, the strength of our tool is the linkage of the textual firewall rules with their visual counterpart as well as the interaction to explore rulesets in detail and to easily test hypothesis about the rule set. Furthermore, through an adaption of the visual mapping our tool can be used to assess performance aspects of a firewall in a fine-granular way. Note that while we built our tool on top of a real-world dataset, for security reasons

all data in this study was anonymized.
The rest of this paper is structured as follows: Section 2 covers related work, Section 3 introduces our firewall visualization tool, which is then demonstrated in a case study in Section 4. Section 5 presents the critical design choices that we made when creating the firewall visualization tool and discusses user feedback. Finally, Section 6 concludes this paper and outlines future work.

2. RELATED WORK

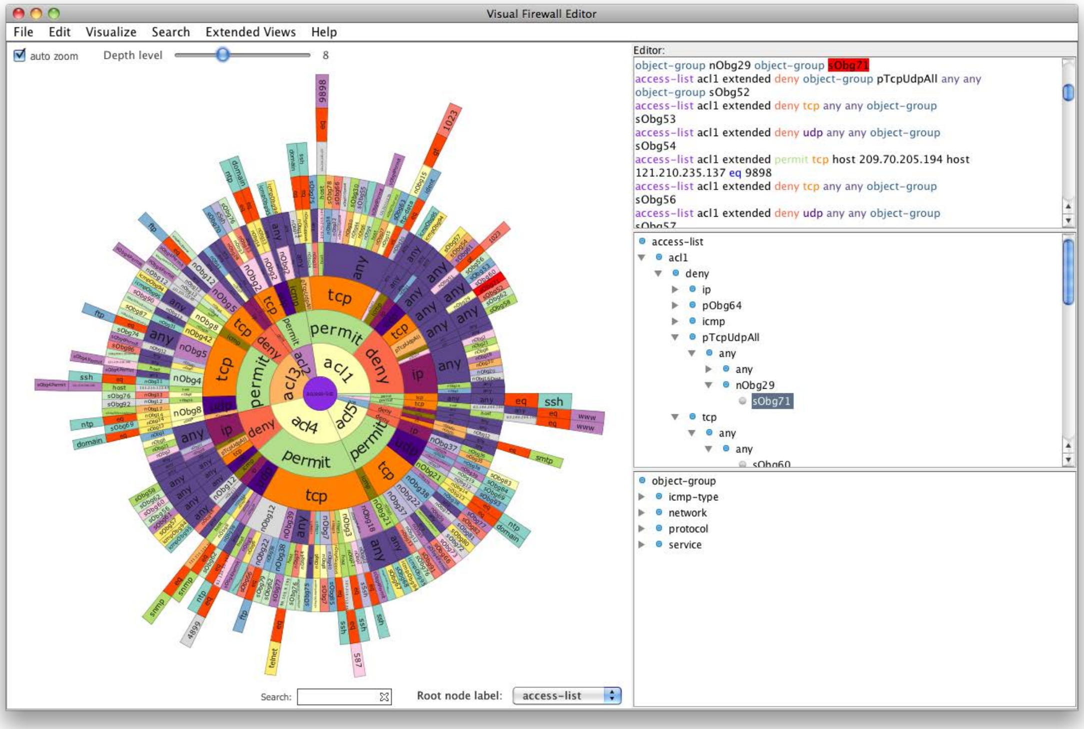
The structure and traffic of large computer networks is one of the most challenging things to visualize not only due to the size and interlinkage of the networks, but also due to the fact that data needs to be captured at different vantage points within the network to obtain a comprehensive picture of the network's underlying dynamics. Some of the early approaches at scale were the internet graphs by Cheswick et al. [3] and the "Visualizing Network Data" study [1] by Becker, Eick and Wilks. While the first approach focused on the topological aspects of the internet using a scalable node-link representation, the second approach aimed at presenting a high-level overview of the actual traffic loads within large networks using maps and matrix representations.
PortVis [15] was one of the first approaches to show the activity of the more than 65,000 application ports in a pixel visualization. After locating interesting ports, the tool offered a time series visualization to investigate port activity over time. TNV [8] put its focus on temporal aspects of host communication patterns. Its basic visual metaphor was a matrix representation of time versus hosts in which traffic was indicated through the cell color and linkage information of selected hosts was drawn on top. Furthermore, an alternative visualization showed the relation between source and destination ports. Later, the Radial Traffic Analyzer [10] aimed at simultaneously investigating more dimensions of the protocol information (e.g. source/target IP, source/target port, time, etc.) in a hierarchical Sunburst [18] visualization. The Hierarchical Network Maps [14] then considered the IP address space in a hierarchical fashion by taking into account the autonomous systems, countries and continents to which individual IP prefixes belonged and displayed traffic information in a color-encoded TreeMap visualization. Finally, the Knowledge Representation in [21] introduced an interactive pattern specification component and processing in the network traffic analysis domain. For a more extensive coverage of the general topic of network security visualization, we refer the reader to a recently published survey [17].

Visualization of association rules or frequent patterns is a field closely related to our study. In this domain, we want to highlight two basic approaches. First, Wong et al. [20] argued against the representation of both antecedent and consequent of a rule as axes of a matrix and settled for a 3D matrix visualization, which links rules to a selected set of items (i.e. topics) using color to distinguish between antecedent and consequent. The summary columns then show support and confidence of the respective rules. This approach was applied to a collection of 3,000 text documents. For the design of the firewall rule visualization tool described in this paper, we were inspired by FP-Viz [11], which also builds upon the concept of the Sunburst visualization. The basic idea is to derive a hierarchy of common rule components. These components are then drawn as ring segments from the inside to the outside. Collecting the segments from the outside to the center of the Sunburst then corresponds to the components of one rule.
Rules also play an important role in the field of network security. So-called intrusion detection systems often use a set of rules to classify network traffic and then output detailed hit statistics. SnortView [12] is a visualization system to analyze the output of the popular intrusion detection system Snort. This visualization system mostly uses matrix representations of source IP address vs. time or source vs. destination ip address. Statistical figures are represented in the background whereas glyph representations in the matrix cells give more details about the events. VisAlert [7] is another approach to visualize such event information. Its design principle is to visualize the network structure in the center and events on several concentric rings that represent time. The angle determines the kind of alert. The linkage between events and network entities is only shown for one selected timeframe. Yet another approach in this field is SprialView [2], which presents network events on a time spiral. Regular events form patterns in the visualization and are thus easy to identify.
To the best of our knowledge, there are currently only three research papers that deal with the visualization of firewall data. Visual Firewall [13] uses line charts, parallel coordinates and matrix views with a combination of color-coding, glyphs, linking and animation to represent the activity of a firewall. In the essence, this work focuses on visualizing network traffic whereas our work concentrates on visualizing the components of firewall rules. In contrast to this PolicyVis [19] presents a visualization tool to interactively validate the rules of a firewall system. Its basic visual component is a configurable matrix visualization, which allows to set a choice of two dimensions from source/destination IP, source/destination port as the axes of a matrix and a third dimension for labeling. Semi-transparent bands in red and green then depict accept and deny rules. Created Voids [16] is yet another approach, which can be used to assess the relationship between different rules by transforming the value ranges of a rule into a geometric object. Respectively two dimensions of the rules create a plane in 3D and the value ranges define a rectangle of this plane. Linking several of these planes with each other creates a geometric object. Rendering several rules as geometric objects then allows to assess the interrelation between these rules. Parallel coordinates are used as an alternative representation of this information space. In contrast to this work, our approach offers a better overview of the ruleset since it displays rules in a non-overlapping manner without occlusion.

While network traffic visualization has been extensively studied, there is only little work on the topic of visualizing firewall rule sets. Since appropriate tools for gaining an overview of a firewall rule set are still missing, we show that our mapping of the rule set into a hierarchical data space will help network administrators to better manage the intrinsic complexity of firewall configurations in the following sections.

3. VISUAL FIREWALL EDITOR

This section includes a description of the firewall rule set data that we use, a general description of our system and a detailed explanation of our visualization of the rule and object groups.



Editor:

```
object-group nObg29 object-group sObg71
access-list acl1 extended deny object-group pTcpUdpAll any any
object-group sObg52
access-list acl1 extended deny tcp any any object-group
sObg53
access-list acl1 extended deny udp any any object-group
sObg54
access-list acl1 extended permit tcp host 209.70.205.194 host
121.210.235.137 eq 9898
access-list acl1 extended deny tcp any any object-group
sObg56
access-list acl1 extended deny udp any any object-group
sObg57
```

```
access-list
▼ acl1
    ▼ deny
        ▶ ip
        ▶ pObg64
        ▶ icmp
        ▼ pTcpUdpAll
            ▼ any
                ▶ any
                ▼ nObg29
                    sObg71
        ▼ tcp
            ▼ any
                ▼ any
                    sObg60
```

```
object-group
▶ icmp-type
▶ network
▶ protocol
▶ service
```

Figure 1: Visual Firewall Editor showing the Sunburst visualization (left), the editor (top right) and two interactive tree views for access lists (middle right) and object groups (bottom right) respectively.

3.1 Firewall Rule Set

Firewall rules are usually specified in a structured textual form. To maintain an overview, individual rules are grouped in *access control lists* (sometimes simply called "access lists") and their structural elements such as protocols, source/target hosts or ports can be grouped in *object groups*. Therefore, one single rule can express more general concepts. As an example "all traffic to the three webservers of the company on the ICMP and UDP protocols should be dropped" can be expressed in one rule rather than in six separate rules.

Object groups define groupings of similar objects, which can then be used within access control lists to define policies for the access control lists. By using object groups there is, in most cases, no need to use individual IP addresses, protocols and ports for the access control lists. The object groups allow so-called multiple access control entries (ACEs) to grant or deny, for example, an entire object group of users access to an object group of servers.

The rule set consisting of access control lists and object groups are usually stored in a firewall configuration file, which is then loaded and compiled by the actual firewall to efficiently process network traffic according to the given rules. Firewall configuration files can become complex and contain a large number of interdependent entries. Therefore, it can be very difficult to configure and manage the access control lists. Access control lists which use object groups tend to be smaller and more readable, which makes the configuration and management easier [5].

3.2 System

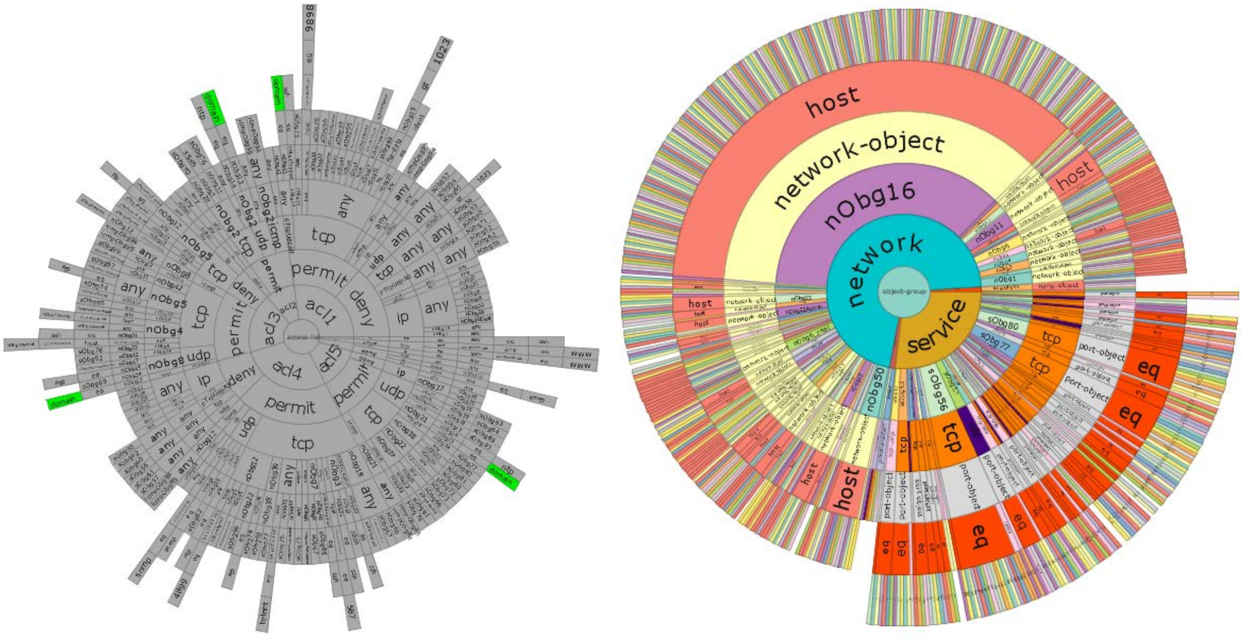
The Visual Firewall Editor is programmed in Java and based on the interactive visualization toolkit Prefuse [9]. In particular, our tool uses an adaption of the DocuBurst [6] code for the Sunburst visualization. Figure 1 shows the Visual Firewall Editor with the visualization of access lists of a firewall configuration file. In principle, our tool is composed of three interface components:

1. The *Sunburst visualization* depicts a hierarchical view on the firewall rules of the loaded configuration file.

2. The *editor* shows the textual representation of this firewall configuration file.

3. The *interactive tree views* show the created access list and object group hierarchies and enable navigation in them.

While the Sunburst visualization will be discussed in detail in the next subsection, we will focus here on the other two interface components and their interactivity. The *editor* on the top right of Figure 1 can be used to display a complete firewall configuration file in textual form containing both object groups and access control lists. In addition to these functional elements, configuration files commonly contain comments written by the authors of the rules serving the purpose of documenting the semantic function of the rules in the historic context of the network infrastructure, which are not visible in the visualizations. To enable easier interpretation of the functional elements of the rules and object groups, we created a consistent coloring scheme for



(a) Search inside the visualization showing the highlighted keyword “domain” in a set of 160 rules.

(b) Completely expanded tree showing the 168 object groups with 554 port-, network-, protocol or group-objects.

Figure 2: Interaction with the Visualization

code highlighting, which is later also employed in the visualization.

The editor’s text area is fully editable. All operations and changes can be saved via the main menu to a new or to the already existing file. The editor also has several interaction options. By pressing the right mouse button on a selected word, a menu pops up with the options copy, paste, cut, undo and highlight. Since structural information is available from the context only the element corresponding to the text marking in the textual rule description of the editor is highlighted in the Sunburst visualization. The resulting highlighting in the visualization is similar to the result of the search feature as shown in Figure 2(a). However, in contrast to the contextual highlighting, the search feature highlights all possible occurrences of the keyword in the visualization. Further search options for the editor are accessible through the main menu; matched keywords in the text are then highlighted in red.

Since rules need to adhere to a strict syntax to be processable by the firewall, we created a *quick edit component* to support the network administrator in this task. After accessing it through the extended views menu, several drop-down boxes with predefined keywords guide the user though the creation of a new rule in the editable text box below. Upon submission, this new rule is added to the current cursor position in the firewall configuration file shown in the editor.

Underneath the editor, two *interactive tree view components* allow the structural exploration of the access control lists and object groups. This view is useful to group rules and objects based on their structural elements. Figure 1, for example, shows the next grouping level of all deny rules of the access control list “acl1” on the right side.

3.3 Rule and Object Group Visualization

The employed Sunburst [18] is a hierarchical visualization technique, which puts a root node as a circle in the middle of the visualization and then recursively maps the elements of each hierarchy level onto ring segments. In order to use this visualization for our dataset, we need to model the firewall rules and object groups in a hierarchical way. In our concrete case, the first level after the root node for the rules visualization consists of the names of the different access lists as shown in Figure 1. The second level contains the access privileges (“permit” or “deny”), the 3rd one the protocol (i.e., “tcp”, “ip”, “udp”, etc.), followed by the source and destination dimensions, which are usually set to values such as “any” or concrete object groups. Note that the remaining dimensions vary in values since these fields are optional for rules, which causes the spikes on the outside of the visualization. Usually these dimensions contain object groups that define grouping of ports or comparisons (e.g. “gt 1023” for destination port greater than 1023).

A hierarchy is formed from our data by grouping identical rule elements in a recursive manner according to the above described dimensional order. As a result, “acl1” in Figure 1 is split up into blocks of permit and deny rules in which each contains among others a block of TCP and IP protocol rules. This subdivision is repeated until the end of each rule is used.

We designed our Visual Firewall Editor in a way that it supports two basic tasks. The first task is the exploration of firewall rules and object groups, whereas the second task is to test certain hypotheses about a rule set. We will now describe the interaction features of our visualization component according to these opposing usage strategies. Note that a common analysis process often involves both strategies and the respective interaction methods can thus also be used for the opposing analysis task.

The depth level slider depicted on top of the visualization in Figure 1 defines the number of hierarchy levels to be shown in the visualization. When the depth level is set to maximum all nodes of the visualization are displayed and the corresponding trees on the right side are expanded. To make the best use of the display space, the auto zoom feature automatically scales the visualization with the selected depth according to the size of the frame. This ensures that the elements of the visualization are enlarged in the best possible way in order to make their labels readable and to enable cross-element comparisons of their sizes. When turning off the auto zoom, the user can zoom in and out of the displayed vector graphic. This interaction is triggered by holding down the right mouse button and moving the cursor up or down.

To improve the visibility of a node and the readability of its label, the width of a segment can be changed interactively using the mouse wheel. This interaction can also be used to decrease the used space of an element when it is irrelevant for the current analysis task and to give more more space for the remaining elements.

When the mouse cursor is set above a node in the visualization, a tooltip with is label is shown. By pressing the left mouse button on a node in the visualization, the corresponding line in the text editor containing the firewall configuration file will be focused in the editor window and highlighted using red background color. Furthermore, the corresponding hierarchy element in the access list or object group treeview on the right is opened and highlighted.

The second usage scenario of our tool is to test hypotheses about a certain firewall configuration. In this scenario, the network administrator already has a concrete idea about what he wants to find out about the firewall rule set. We created several interaction capabilities in the tool that support this goal-driven search for information:

First, every node can also be expanded manually to each depth level through a left mouse button click on the node without expanding the whole tree with all nodes. Figure 3 shows a node that has been expanded manually to the protocols and object groups that will be permitted. As already described above, this interaction also triggers highlighting and focusing in the text editor and the respective treeview.

Second, there is also a possibility to search for a specific keyword, which is highlighted in the visualization to obtain a closer view of a known element and its connectivity with other access lists. Figure 2(a) shows the use of the search option with the keyword domain. It can be seen that "domain" is included in four access lists. The number of matches is also displayed on the left side of the search field.

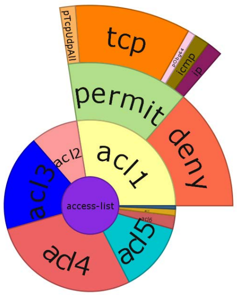


Figure 3: Manual node expansion

Third, the user-defined visualization provides the ability for the user to visualize the relationship of arbitrary rule elements in a user-defined priority. For example, the user can visualize the access control list, source objects, protocol and access privileges in descending order. Figure 4(a) shows that nearly all rules for TCP protocol permit access. Another insightful visualization could be a comparison of the permit and deny rules for each access control list as shown in Figure 4(b). All of the access control lists contain "permit" rules, whereas only 6 out of 8 contain deny blocks.

Note that both the above described visualization and interactions can not only be applied to firewall rules, but can be used in exactly the same way on object groups of a firewall configuration. In our tool, this is done by changing the root node from access list to object-group. The result of this change is shown in Figure 2(b) . It can be seen that there are two types of object groups with a large number of entries ("network" and "service") and two types with only a few entries. If the visualization is expanded to maximum depth level, all elements of the visualization are shown. The fine-granular elements at the outer rings define concrete IP addresses for host objects and port number for service objects.

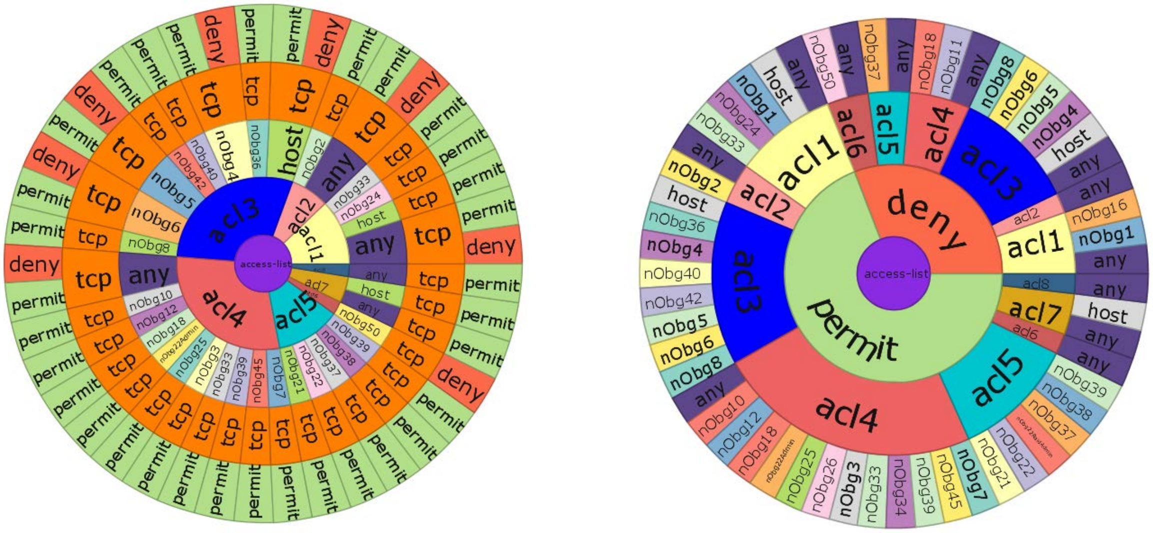
4. CASE STUDY

This section describes two relevant use cases within the analysis of a firewall configuration file. The configuration file consisted of 160 rules, which were grouped into 8 access control lists and 168 object groups. In total, this configuration file consisted of approximately 1150 lines of code.

4.1 Analysis of Rule Usage

Another issue of firewall configuration is the matching of traffic for the access control entries. For this reason a hit count value can be added to each access control entry. This value documents how often a certain rule was matched and is thus commonly used for troubleshooting connectivity and security of a firewall configuration. Since it is tedious to compare these hitcount values and to manually aggregate them on access control lists, we configured our Visual Firewall Editor in such a way that it can read and aggregate these statistical usage values. Figure 5(a) shows the resulting visualization of access control lists with hit count values. It can be seen that the access control lists "acl1" and "acl4" are very large, which means that the the sum of all hit count values for those access lists is very high.

Since there is a large variance in the hitcount values, we use logarithmic scaling to determine the width of the rule segments. While absolute comparisons are not possible any more, this avoids the effect that one or a few standard rule cover the whole visualization space. The segment of the access control list "acl1" in Figure 5(a) is very large because of high hitcount values whereas the next access control list "acl2" almost disappears due to a small number of hitcount values. It can also be seen that the ring segment of the deny rules of "acl1" compared to the ring segment of the permit rules have almost the same size. This means that the traffic of the permit and deny rules is nearly the same. In case of the access control list "acl4" it can be seen that its permit rules compared to its deny rules have a larger circle segment. The hit count visualization in general gives a good overview what access control lists apply for high traffic and vice versa.



(a) Access list, source object, protocol, access privilege

(b) Proportions of permit and deny rules with access lists and source objects

Figure 4: Hypothesis driven analysis with user-configured visualizations. The rule hierarchy is specified from inside to outside using a subset of the rule fields (i.e. the access privileges) and/or values (e.g. tcp).

So the user can individually decide what access control list should be inspected in detail.
Furthermore, we provide the option to display unused rules in a firewall configuration, which all have a hitcount value of 0. Figure 5(b) shows that "acl3" and "acl5" have a very large number of access control entries with a hit count value equal to 0. Access control list "acl4" is nearly half the size in contrast to the visualization of used rules in Figure 5(a). In general, the fact that a rule has no hits does not necessarily mean that it is not useful from a security point of view. However, some of these rules can be considered as historic artifacts, which are either not necessary any more (especially permit rules that were not used for several months) or are not executed due to other executed rules that now fill their place.

4.2 Understanding Nested Object Groups

In a CISCO firewall configuration, object groups provide the feature of including other object groups in their definition. While this gives more flexibility to the network administrators, it also increases the complexity of a firewall configuration. Our firewall visualization can be used to explore such dependencies within object groups. Figure 6 shows the visualization of all nested object groups of the configuration file with a detailed view on the object group nObg22Admin. The object group nObg22Admin has several network-objects and as a nested object group the nObg3 group object. In this case all details of the network-objects of the nested object group are displayed for inspection.

5. DISCUSSION

The purpose of this section is to discuss on the one hand the implicit and explicit design choices that we made in our tool and on the other side the feedback that we obtained from firewall administration experts.

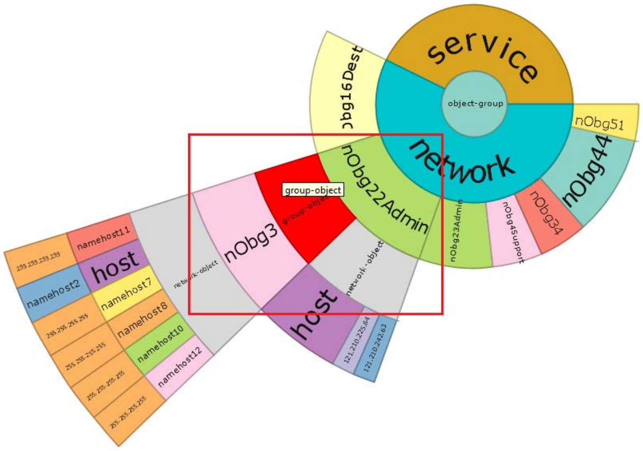
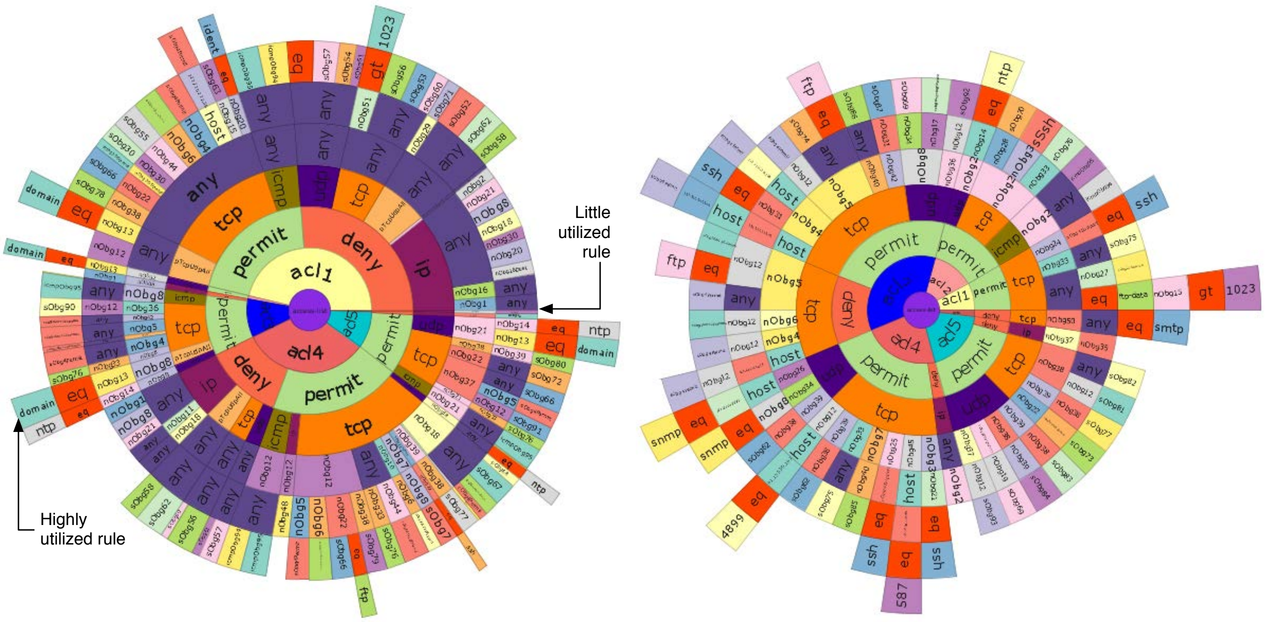


Figure 6: Nested object groups

5.1 Design Choices

In the process of designing a visualization tool to support network administrators in their task to understand and maintain a firewall configuration, we considered many options. After initial tries, node link diagrams (also known as graphs) were discarded due to emerging ambiguities when linking several rule elements or object group elements with each other. Furthermore, since the information space was so large, we opted for a compromise between showing aggregations and showing the full level of details, which eventually converged to the choice of using an adaptation of the Sunburst visualization linked with classical treeviews and a text editor.

The usage of color for establishing a linkage between the textual elements in the text editor on white background



(a) Access control lists with hitcount values

(b) All unused rules (hitcount value = 0)

Figure 5: Expanded tree for access control lists with and without logarithmically scaling the rule segments' width according to rule hits. Note the expansion of "acl1" in (a) due to high usage of its rules and the many unused rules in the blue access list "acl3" in (b).

and the solid elements of the Sunburst visualization required many iterations. Since the distinction of color for categorical values is very limited, we were forced to reuse some colors, which introduced ambiguities. In the end we opted for fixed colors for the most common keywords (e.g. "TCP", "any", etc.). Especially for the access privileges we used familiar semantics by assigning green to "permit" and red to "deny". Less common keywords and names are assigned repeating colors. Both search and interaction features can then be used to resolve ambiguities. Unfortunately, these color design choices make the system sub-optimal to use for people suffering from color blindness.

Another critical design decision was not to use blank elements to compensate for non-existing elements in rules. While this lead to spikes on the outside of the circle, we found it visually more appealing than holes in the inside of the visualization.

5.2 User Feedback

Feedback from a group of five firewall administrators in our IT department was very encouraging. Some stated that an overview was possible for the first time. They liked the fact that finding things in a configuration file was supported both through clicking on elements in the visualization and keyword search. Especially the two-way linkage between the visualization and editing through color linkage and interaction was considered a major benefit. One person also stated that changing the firewall configuration file in such an interactive way would make work easier, in particular when searching for the right place to amend a new rule. Furthermore, they wanted to use this tool in order to get rid of historically grown rules, which were not necessary any more.

Unfortunately, due to the limited human resources in this research project, we were not able to provide the necessary support for a long-term deployment of our prototype. However, many feature requests by the users such as the support for other firewall configuration formats, comparison capabilities for different versions of a configuration, extensions towards large log files (typically several hundreds MegaBytes), packet testing features, and a translation of user policy settings to access control lists showed us not only the users' engagement into our project, but also the need for further developments in this direction.

6. CONCLUSION

In this paper we presented a research prototype for the analysis of complex firewall configurations. The main focus of this work was to break down the complexity of many rules in access control lists and complex object groups through a Sunburst visualization. In order to use this visualization we abstracted the firewall configuration data by deriving a hierarchy based on common attribute values of object groups and rules. A strong focus of our study was to define the contextual interaction capabilities between the text editor containing the firewall configuration file, the Sunburst visualization showing an abstracted representation of the data and two treeviews listing the hierarchically grouped data.

The presentation of our prototype to a group of five firewall configuration experts revealed not only the value that this tool provides to simplify common steps in their work, but also showed the need for further developments in the domain of firewall visualization.

In general, we believe that our visualization is quite scalable in the number of rules and object groups it can display. Only labeling became a major problem in our case, which was resolved through zooming features. Nevertheless, we realized that there exist larger configuration files than the one used in this work. Dealing with such files would require additional filtering criteria, an aspect that might be considered in future work. Further aspects to be considered in future work are taking into account the order of rule processing, rule testing, such as a generalized search for IP addresses within prefixes of rules and objects, and a temporal visualization for historic rule usage.

7. ACKNOWLEDGMENTS

We thank Andreas Merkel and Marcel Waldvogel from the IT service center of the University of Konstanz for their continuous support during this project. Furthermore, we are very grateful to Stephan Pietzko who shared many insights into this complex topic with us.

8. REFERENCES

[1] R. Becker, S. Eick, and A. Wilks. Visualizing network data. *Visualization and Computer Graphics, IEEE Transactions on*, 1(1):16–28, 2002.

[2] E. Bertini, P. Hertzog, and D. Lalanne. SpiralView: towards security policies assessment through visual correlation of network resources with evolution of alarms. In *Visual Analytics Science and Technology, 2007. VAST 2007. IEEE Symposium on*, pages 139–146. IEEE, 2007.

[3] B. Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the internet. In *Proc. USENIX Annual Technical Conference*, pages 1–12. Citeseer, 2000.

[4] W. Cheswick, S. Bellovin, and A. Rubin. *Firewalls and Internet security: repelling the wily hacker.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1st edition, 1994.

[5] CISCO. Cisco systems. object groups for acls, September 2010. http://www.cisco.com/en/US/docs/ios/sec_data_plane/configuration/guide/sec_object_group_acl.html.

[6] C. Collins. Docuburst: Document content visualization using language structure. In *Proceedings of IEEE Symposium on Information Visualization, Poster Session. Baltimore.* Citeseer, 2006.

[7] S. Foresti, J. Agutter, Y. Livnat, S. Moon, and R. Erbacher. Visual correlation of network alerts. *IEEE Computer Graphics and Applications*, pages 48–59, 2006.

[8] J. Goodall, W. Lutters, P. Rheingans, and A. Komlodi. Preserving the big picture: Visual network traffic analysis with tnv. In *Visualization for Computer Security, 2005.(VizSEC 05). IEEE Workshop on*, pages 47–54. IEEE, 2005.

[9] J. Heer, S. Card, and J. Landay. Prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430. ACM, 2005.

[10] D. A. Keim, F. Mansmann, J. Schneidewind, and T. Schreck. Monitoring network traffic with radial traffic analyzer. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST '06)*, 2006.

[11] D. A. Keim, J. Schneidewind, and M. Sips. Fp-viz: Visual frequent pattern mining. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis '05), Poster Paper*, 2005.

[12] H. Koike and K. Ohno. SnortView: visualization system of snort logs. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 143–147. ACM, 2004.

[13] C. Lee, J. Trost, N. Gibbs, R. Beyah, and J. Copeland. Visual firewall: real-time network security monitor. In *Visualization for Computer Security, 2005.(VizSEC 05). IEEE Workshop on*, pages 129–136. IEEE, 2005.

[14] F. Mansmann, D. A. Keim, S. C. North, B. Rexroad, and D. Sheleheda. Visual Analysis of Network Traffic for Resource Planning, Interactive Monitoring, and Interpretation of Security Threats. *IEEE Transactions on Visualization and Computer Graphics*, 13(6), 2007.

[15] J. McPherson, K. Ma, P. Krystosk, T. Bartoletti, and M. Christensen. Portvis: a tool for port-based detection of security events. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 73–81. ACM, 2004.

[16] S. Morrissey and G. Grinstein. Visualizing firewall configurations using created voids. In *6th International Workshop on Visualization for Cyber Security (VizSec)*, 2009.

[17] H. Shiravi, A. Shiravi, and A. Ghorbani. A survey of visualization systems for network security. *Visualization and Computer Graphics, IEEE Transactions on*, 99(RapidPosts), 2011.

[18] J. Stasko, R. Catrambone, M. Guzdial, and K. McDonald. An evaluation of space-filling information visualizations for depicting hierarchical structures. *International Journal of Human-Computer Studies*, 53(5):663–694, 2000.

[19] T. Tran, E. Al-Shaer, and R. Boutaba. PolicyVis: firewall security policy visualization and inspection. In *Proceedings of the 21st conference on Large Installation System Administration Conference*, pages 1–16. USENIX Association, 2007.

[20] P. Wong, P. Whitney, and J. Thomas. Visualizing association rules for text mining. In *InfoVis*, page 120. Published by the IEEE Computer Society, 1999.

[21] L. Xiao, J. Gerth, and P. Hanrahan. Enhancing visual analysis of network traffic using a knowledge representation. In *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, pages 107–114. IEEE, 2006.