Network Security Visualization

Keith Fligg and Genevieve Max

1 Introduction

Network security is the art and science of detecting, stopping, and defending against current and future network intrusion incidents. As attack tools get more sophisticated and more freely available, it becomes easy for even a novice to launch attacks. Attacks can have many different motives, including bringing network services down, stealing confidential information, and misusing computing resources.

All of the activity generated by an attacker leaves traces, which can be seen by inspecting network traffic. Log files are created in response to both malicious and innocuous activity, depending on the logging level. These logs and traces are what network security professionals have to work with in order to detect, defend against, and prevent security breaches.

The problem is the vast amount of network traffic data to be sifted through. Security tools have been developed to help security analysts process all this data. The aim of any security tool is to find patterns, determine if these patterns are anomalies and communicate the severity of the attack.

One of the most useful tools for network security workers are visualizations. Just as graphs and charts are useful to convey ideas about the economy, experimental results and your spending habits, they can also help make sense of the mass of data related to network operations. Humans are better at processing visual information since their sense of sight is more developed than any other sense. It is the same logic as the common saying "a picture is worth a thousand words." The aim of network security visualization is not to create new information, but to present the available information in an easy to digest manner.

Figure 1 shows a typical attack and how visualizations help the security analyst fix vulnerabilities much faster than reading through raw network data. However, creating just any image of data will not achieve the goals of network security visualization. There is a need to understand visualization theory just as much as network security in order to design and build an effective network security visualization tool.

2 The Psychology of Visualization

Network security visualization allows security analysts to take the multitude of text logs produced by a network and generate an image indicating what information these logs contain. Since such a large portion of our brain is dedicated to gathering information by looking at it, it makes sense that we would want to use this natural strength to our advantage when designing network security tools. A properly structured visualization allows us to take large amounts of information and process its meaning quickly.

However, simply transforming text logs into a picture will not achieve the goal of easily understanding network activity. The visualizations constructed must allow important information to be more visible than less important information. This concept is called *pre-attentive*, which is the

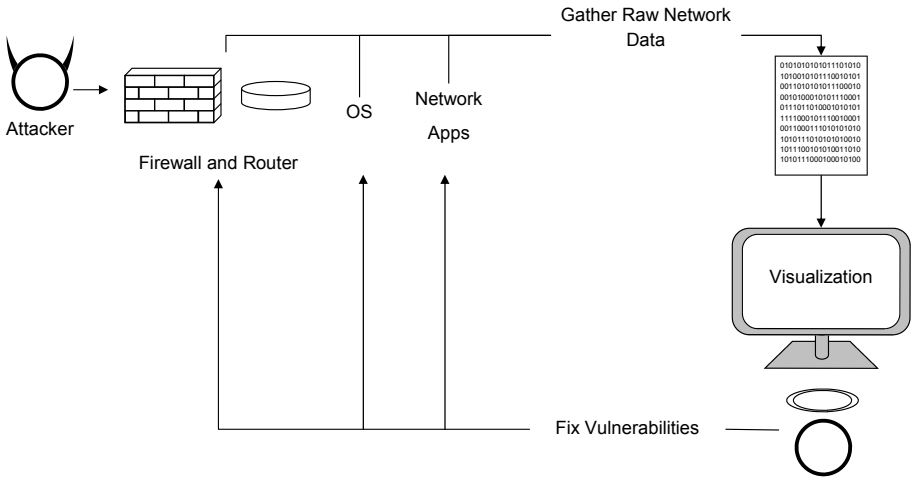


Figure 1: A typical attack on a network must be detected as quickly as possible. This speed is achieved using network security visualization.

characteristic seen when an image stands out from other images. Commonly, the circles, squares, lines and other shapes that make up an image are called *objects*. Objects that are pre-attentive are effortlessly distinguishable from other objects. Additionally, pre-attentive objects are easy to spot no matter how many irrelevant objects, *distractors*, there are in the same image. This concept is particularly important to network security visualization as we want an event, such as a security breach, to be immediately visible no matter how much normal activity is occurring in the background.

The amount of time it takes for the human brain to process pre-attentive objects versus non-pre-attentive objects is a factor in why pre-attentive objects have their stand out effect. A pre-attentive object will be processed in 10msec or less. A non-pre-attentive object will be processed in 40msec or more. This means pre-attentive objects are perceived by the brain before the human is even aware of what they are looking at.

2.1 Pre-Attentive Objects

Pre-attentive objects can be categorized by their:

- Color
- Position
- Form
- Motion

Color is pre-attentive as seen in Figure 2(a). The dark circle is quickly seen, even though there are several lighter circles in the image. Position refers to the physical location of objects in an image. Figure 2(b) illustrates how position is pre-attentive, in that the circle that is offset from the others can be quickly spotted. Form describes a physical difference between an object of interest and all other objects. Some examples are shape, size, orientation and enclosure. Figure 2(c-f) shows how these can be used. Motion refers to an object moving or blinking on the screen, as opposed to being stationary.

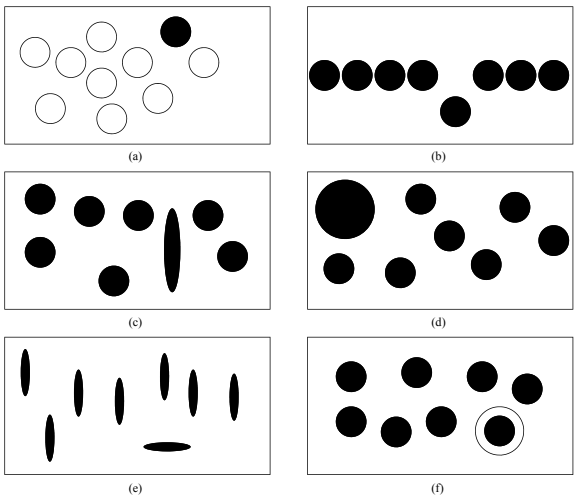


Figure 2: Pre-attentive objects are distinguishable from the objects surrounding them. (a) Color (b) Position (c) Form – Shape (d) Form – Size (e) Form – Orientation (f) Form – Enclosure

2.2 Visualization Techniques

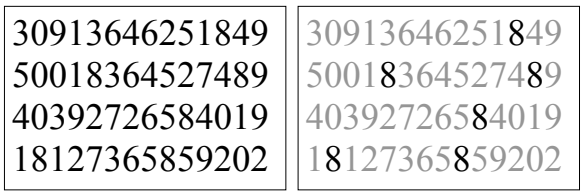
A challenge in network security visualization is that there is no single right way to represent network logs because these logs are abstract data. As an example, other science disciplines can use visualizations to represent molecules or geographic features. Such visualizations could be based on physical measurements and data. In network security, data is abstract and does not correspond to physical data. Despite this challenge, there are techniques that can be used to develop an effective visualization of network security data.

Some of the features sought after in visualizations displaying large amounts of data, described in subsequent paragraphs, are:

- Remove Serial Parsing
- Minimize the Number of Types of Objects
- Minimize Non-data Ink/Pixels
- Determine Root Cause
- Provide Interactive Display
- Allow Data Comparisons

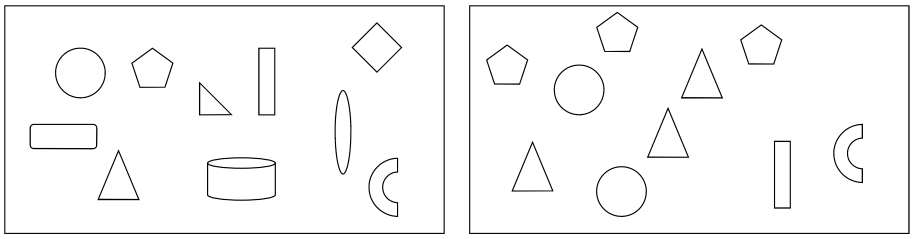
A visualization must allow the user to understand the image without using serial parsing, or visual scanning. This follows the idea of pre-attentive objects. The user of the system should not have to visually scan an image to find what is important, or there would be no improvement over visually scanning log files. Instead, a visualization must make the important information stand out. A popular way to illustrate this is shown in Figure 3. In Figure 3(a) it is difficult to find all the 8s in the table. In Figure 3(b), the pre-attentive category of color is used to easily detect where the 8s are.

When designing a visualization, attention must be paid to keep the number of different types of objects to a minimum. Although color and shape are pre-attentive, overuse of these can have the



(a) (b)

Figure 3: Reduce serial parsing by highlighting information the user needs to see immediately, such as the 8s



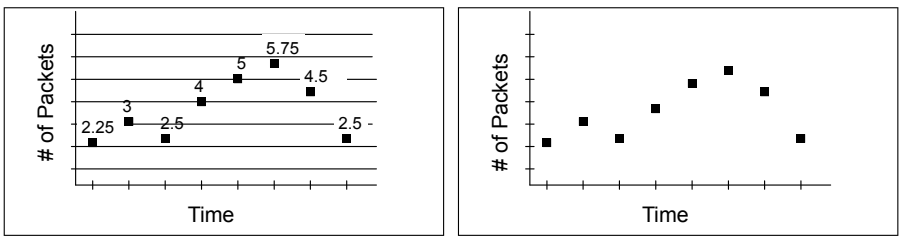
(a) (b)

Figure 4: Minimize the number of objects, such as the number of different shapes

opposite effect. The number of shapes should be kept to a minimum to avoid important information from being overlooked. A general good number to use is five to eight shapes per visualization. In Figure 4(a), the image shows how too many shapes detract from what is important. Figure 4(b) highlights two possible anomalies, the rectangle and the arc.

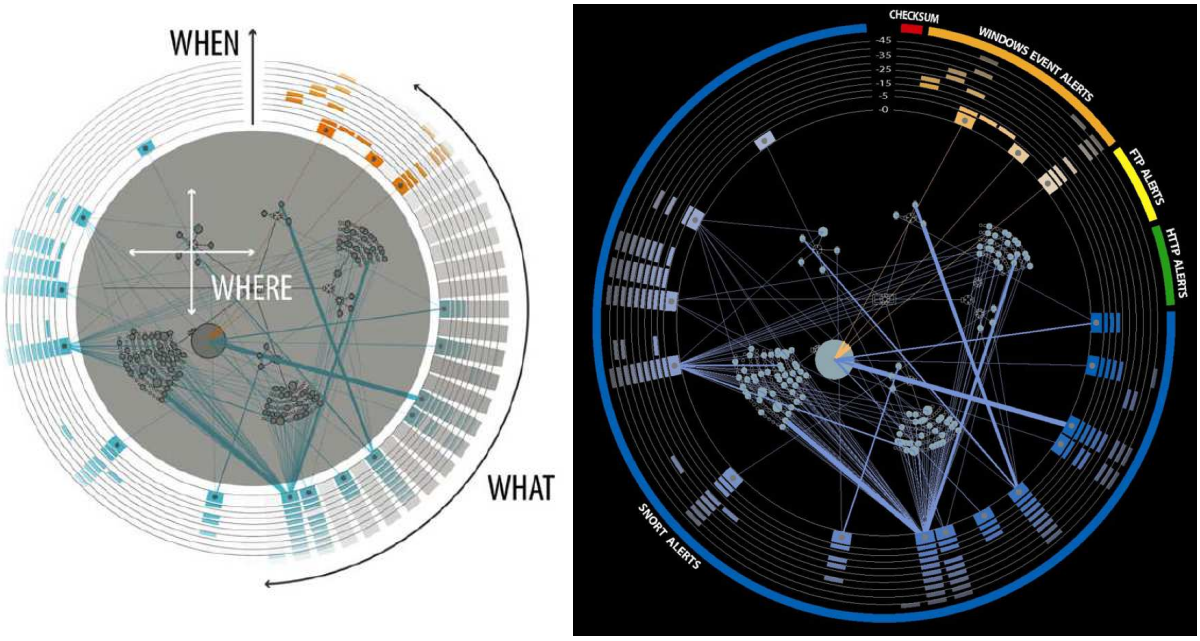
There are only so many pixels on any screen, and whatever visualization is produced must make the best use of what resources are available. For this reason, the amount on non-data ink, or pixels, must be minimized. An example of non-data ink on a chart are the two lines drawing the x and y axes. While these are necessary, we may not need the tick marks at each interval, so they should be removed. As seen in Figure 5(a), the numeric labels on each data point and the horizontal grid lines on the plot are examples of non-data ink that can be removed without affecting the main message the graph is conveying. Figure 5(b) shows how this graph can be made easier to read.

While a visualization may show that a security breach has occurred, it is even more important that the root cause, or why the breach happened, can be determined from the visualization. This



(a) (b)

Figure 5: Minimize the amount of non-data ink, such as excess lines in a graph



(a) **(b)**

Figure 6: Example of a network security visualization tool applying visualization concepts. (a) This visualization shows how an image can anwser the questions *where*, *when* and *what* about an attack. (b) This is the same visualization, but showing the actual tool usage, including what types of alerts are being analyzed. Image(a) from: [6], image (b) from: [7]

may require a visualization to be interactive, or allow the ability to overlay multiple data sources.

A network security visualization must not simply be an image, it must allow the user to further explore and understand the data they are looking at. Creating an interactive display achieves this goal.

It is not enough to know that an anomaly has been detected in a network. It is also important to show the amount of deviation from normal system behavior. To achieve this, a visualization should allow comparisons between different sets of data.

2.3 Application to Network Security Visualization Tools

A network security tool must answer three major questions:

- *Where* in the network is the attack happening?
- *When* is the attack happening?
- *What* type of attack is happening?

Figure 6(a) illustrates how visualization techniques discussed in the previous sections can be applied to answer these three questions. The center portion answers the question *where*. The circles represent different nodes in the network, and the network topology is shown through the links between the circles. The size of the circle increases when a node experiences more alerts. The pre-attentive idea of form based on size is used to indicate security issues.

Each column in the outer circle corresponds to a different type of attack, and answers the question *what*. Figure 6(b) shows actual usage of the tool with the types of alerts being looked for.

The circles moving outward represent time, with the circles closer to the network topology being more current, and past activity moving outward. This part of the visualization answers the question *when*.

3 Data Sources

In order to generate visualizations, data is needed. One source of data are log files generated by firewalls, web servers, routers and other tools that utilize the network. Other sources of log files may be virus/malware detection systems, email systems, and the operating system itself. In addition to log files, network packets themselves may be inspected and used to create visualizations. Finally, Intrusion Detection Systems can aggregate log files and network packets and perform actions according to user-defined rules. All of the above are fodder for visualization generation.

3.1 System Log Files

Almost every daemon or service that runs on a computer is capable of generating information that pertains to its operation. Some simply write log files directly to their own logging subdirectory, like Tomcat, while others utilize system-wide logging services, such as syslog, which stores logs in a dedicated operating system directory, e.g., /var/log. Additionally, the log level of most software is configurable and generally range from extremely verbose debugging output to very terse logs of exceptional conditions and critical errors.

In almost all cases, the log entries are time stamped so that there is a chronological order to the log entries. This is important because without a clear ordering, subsequent entires may not make sense, because of the lack of context.

Similarly, most log files on UNIX and UNIX-like systems are text-based. This means that log files can become quite large – especially at greater levels of verbosity. There are automatic mechanisms to rollover and compress log files, but this is only a stopgap measure, since most systems do not have infinite disk space. Therefore, there is usually a relatively small window of time during which log inspection may occur.

3.2 Packet Data

The protocol of the Internet, and thus what nearly all network security is concerned with, is IP, or Internet Protocol. Another is ICMP, or Internet Control Message Protocol, upon which the 'ping' utility is built. IP routes datagrams between machines on the Internet and each machine is assigned an IP address.

The datagrams that are transported with IP include TCP (Transport Control Protocol) and UDP (User Datagram Protocol) among others. These are part of what is considered the transport layer. Applications like DHCP, HTTP, FTP, DNS, etc. run on top of this layer and are part of the application layer.

Of primary concern to network security on the packet level are TCP and UDP. Application layer protocols are generally dealt with as log files, whereas the TCP and UDP packets can be taken apart and analyzed separately. The primary advantage of looking at packets is that they contain the 'ground truth'. Log files may be manipulated by intruders and malicious users, so called insider threats, but the packet data can be retrieved directly from the networking stack that runs in kernel space, and are therefore much harder to manipulate.

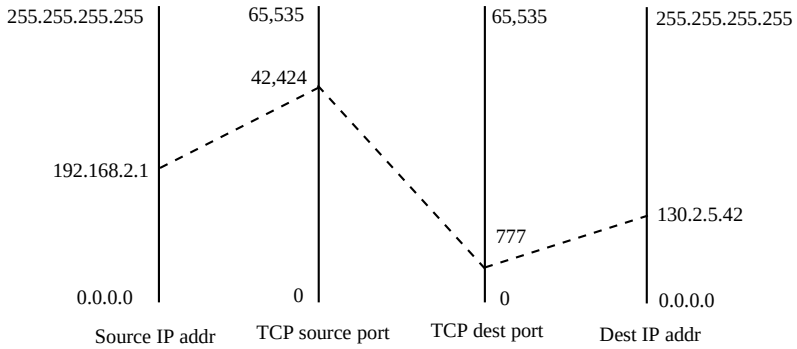


Figure 7: Parallel coordinate plot for a TCP packet from 192.168.1.1:42424 to 130.2.5.42:777.

3.3 Intrusion Detection Systems

Another source of visualization data are Intrusion Detection Systems, or IDSs. These are policy-based security systems that monitor log files and/or network packets looking for conditions that may indicate a security breach. They themselves may write log files, send alerts to system and network administrators, and/or write to the console. Some IDS systems, known as Intrusion Prevention Systems may even modify firewall rules in response to events.

One of the issues with IDSs are the rules that they use. On one hand, rules written too strictly may generate false positives. One the other hand, rules can be too loose and allow attacks to occur undetected, a so called false negative. Balancing the strictness of rules can be difficult in itself, and may be made worse by network users that are doing 'interesting' things on the network that cause rule relaxation to reduce false positives.

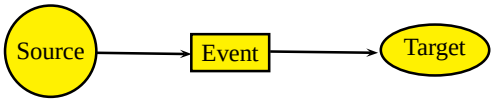
4 Example Visualizations

Visualizations come in many shapes, sizes and colors, and most are based on traditional methods of data visualization. Different types of data may be more suitable for a different type of visualization. Additionally, as we try to cram more dimensions (dimensions of data, not necessarily spatial) into the visualization, new and different techniques are necessary to make the resultant visualization useful and not just a cluttered mess of colored pixels.

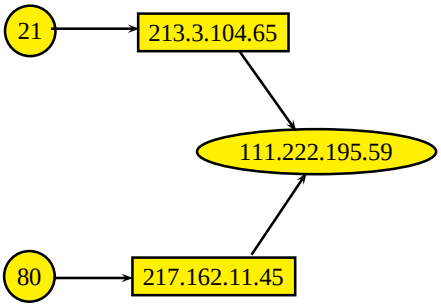
One of the most useful visualizations to investigate data packets is the *parallel coordinate plot* illustrated in Figure 7. This type of plot makes it easy to see how traffic is flowing between machines. The number of axes used is only limited by screen width, but a practical limit is around 20. Figure 9 shows an example with 19 axes, showing one evening of Internet traffic to a single computer placed directly on the Internet, and not responding to any traffic. While the graph is busy, it shows overall trends in packet characteristics, and serves as a good starting point from which deeper evaluation may take place. From here packet length, protocols, payloads, and traffic patterns may be investigated by filtering the input, changing axis assignments or zooming in on certain spans, for example ports between 0 and 1024.

Another useful visualization is the *link graph*, depicted in Figure 8. A link graph is a directed graph who's basic structure is shown in Figure 8(a). The circular 'source' vertex points to the square 'event' vertex, which in turn points to the 'target' vertex. The link graph is another way to visualize traffic in a network, and is usually used to display firewall logs. It is discussed in more detail in Section 5.

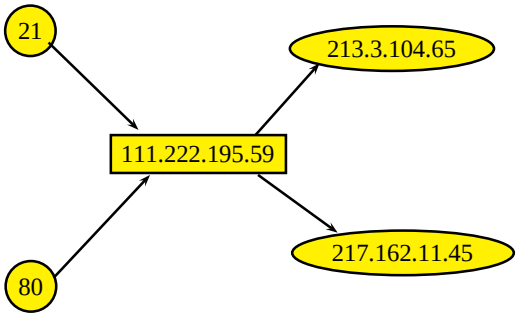
In many cases, different views into the same data are needed. One solution to this problem is to



(a) Link graph nomenclature.



(b) Destination port, source address, and destination address.



(c) Destination port, destination address, and source address.

Figure 8: Example link graphs.

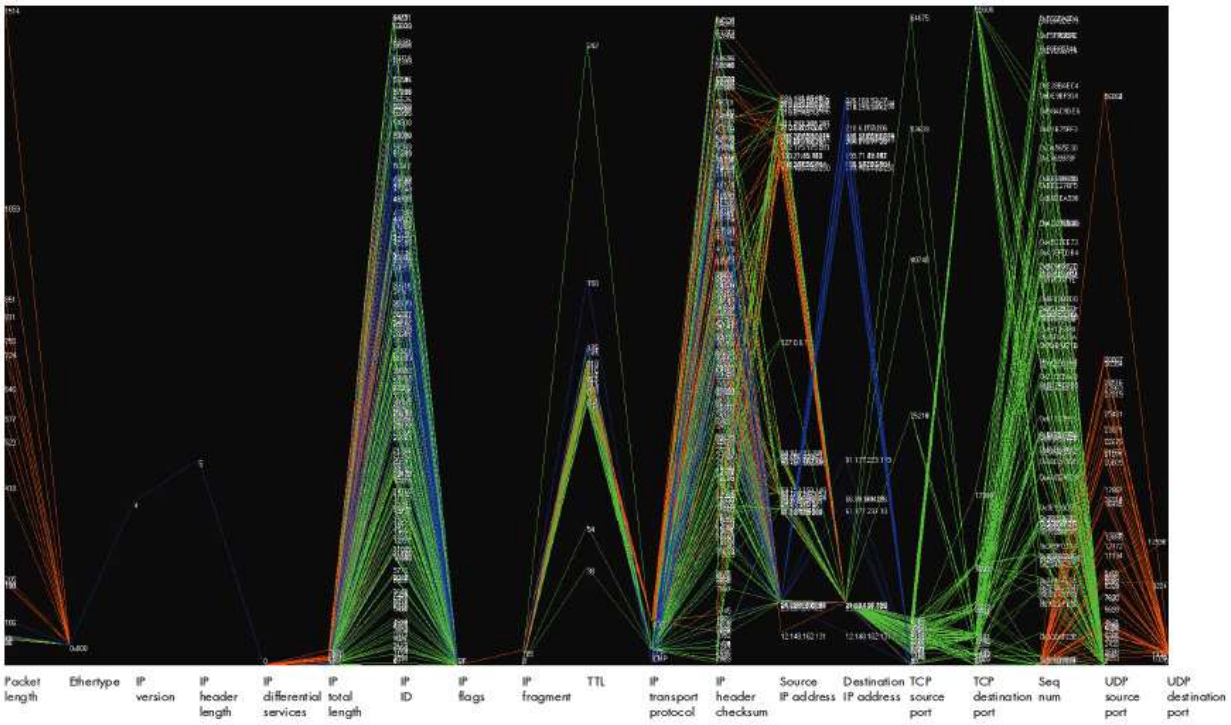


Figure 9: An example parallel coordinate plot from [4].

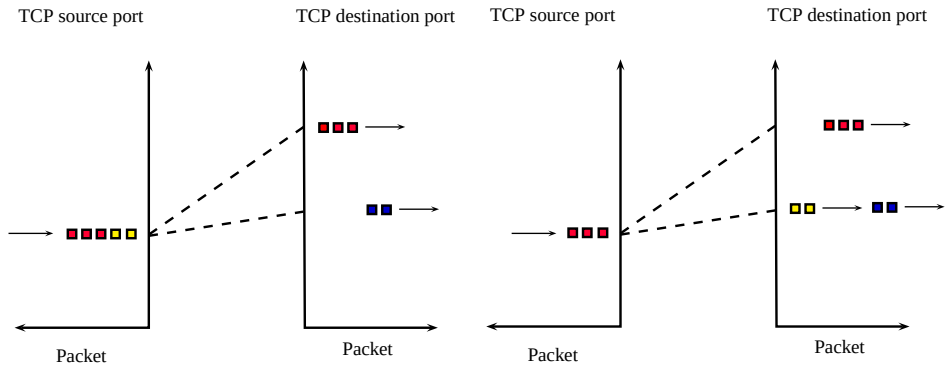


Figure 10: Packets are animated in this parallel coordinate plot. The left side shows five packets arriving, with five having already arrived. The right side is after two more packets have arrived.

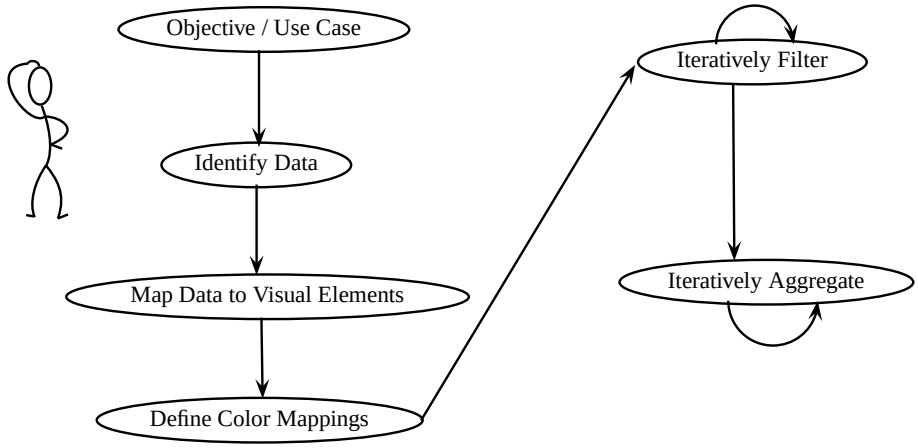


Figure 11: Typical Visualization Creation Workflow

provide several different windows with different views. For example, a bar chart showing the packet count per target address; a parallel coordinate plot showing attacker IP address and destination port in another window; and a pie chart showing the business role of target machines in yet a third window. All of these may be controlled by a sliding window on a timeline showing packet count per unit of time. Using multiple views, especially when they all automatically update to reflect the current dataset, can help to gain insight into the data that may not be available by just looking at one type of graph.

Animation can be useful in network security. For example, adding packet animation to the parallel coordinate plot can be very useful. This alleviates a serious problem with the parallel coordinate plot: if many packets have the same attributes, there will be only one plot for all of the packets. As shown in Figure 10, animating the packets by arrival time significantly clarifies the graph.

5 Workflow

The typical workflow used to create visualizations can be seen in Figure 11. The first step is to clearly state the purpose of the visualization. This important, often ignored step is crucial to ensuring that subsequent steps are properly aligned. Skipping this step may lead to some interesting discoveries, but most of us are very much event driven, and unless you start with a clear use case,

time will be wasted. Finally, it is at this stage that the type of visualization to use will be decided.

The next step is to identify the data fields that will be visualized. This may be source and destination IP addresses or ports if the data are network packets. Netflows may simply be the IP addresses. Timestamp data may be necessary if looking at data over time, or the change of a quantity over time. Log files may include user information, filesystem information, etc. in addition to networking data.

Once the data has been identified, one needs to map the data to the individual aspects of the visualization being created. This step is important, since choosing the incorrect mappings can obfuscate what is happening with the data, as shown in Figure 8(b) and (c). Figures 8(b) uses the structure of destination port → source address → destination address, and clearly shows two connections to 111.222.195.59. These are the connections to port 21 from 213.3.104.65 and another to port 80 from 217.162.11.45.

The same information visualized as destination port → destination address → source address is shown in Figure 8(c). The problem with this graph is that it is impossible to tell which host initiated the connection to port 21 and which to port 80. Perhaps both hosts made connections to each port? It's not possible to tell. This is an excellent example of how choosing the correct visual element can make or break a visualization.

After the data is mapped to the elements of the visualization, color mappings may be chosen. The use of color should enhance the immediate understanding of the data. Sometimes color is not necessary, and can be harmful if it is abused, as discussed in Section 2.2.

Once data is mapped to colors and visual elements it's possible to draw the visualization. However it may not immediately show what's important, but hopefully some patterns appear. The next step is to filter the data to make the interesting patterns clearer. Filtering may include removing events that might be firewall or server misconfigurations, like DNS not being setup correctly. This is an iterative process because as data are removed, the picture will become clearer (or not) and you will need to either filter more data, or put some previously filtered data back.

Finally, another data reduction method that is similar to filtering is aggregating. An example of aggregating data is to lump together all incoming traffic by the first IP octet, e.g., 194.xxx.xxx.xxx. A variation is to only identify internal hosts by the last octet. This clears up the graph, while still leaving the pertinent information in the visualization. The difference between filtering and aggregation is that in the former case, the filtered data is completely removed from the visualization, whereas in the latter case, the data still exists; however it's combined with similar data, thus reducing the pixel count of the visualization.

6 Conclusion

Network security visualization combines both the fields of visualization theory and network security practices to help ease finding attacks on network systems. These tools, when properly designed, allow details about the attack's progress to be determined quickly. By gathering the right data and visualizing it logically, situational awareness is increased, and attacks can be easily communicated to the people who need this information and the attack can be addressed accordingly.

References

[1] Robert Ball, Glenn A. Fink, and Chris North. Home-centric visualization of network traffic for security administration. In *In VizSEC/DMSEC 04: Proceedings of the 2004 ACM workshop on Visualization and*, pages 55–64. ACM Press, 2004.

[2] Ryan Blue, Cody Dunne, Adam Fuchs, Kyle King, and Aaron Schulman. Visualizing real-time network resource usage. In *Proceedings of the 5th international workshop on Visualization for Computer Security*, VizSec '08, pages 119–135, Berlin, Heidelberg, 2008. Springer-Verlag.

[3] Bill Cheswick, Hal Burch, and Steve Branigan. Mapping and visualizing the internet. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '00, pages 1–1, Berkeley, CA, USA, 2000. USENIX Association.

[4] Greg Conti. *Security Data Visualization: Graphical Techniques for Network Analysis*. No Starch Press, 2007.

[5] Anita D. D'Amico and K. Whitley. The real work of computer network defense analysts. In Goodall et al. [8], pages 19–37.

[6] Stefano Foresti, Jim Agutter, Yarden Livnat, Shaun Moon, and Robert Erbacher. Visual correlation of network alerts. In *IEEE Computer Graphics and Applications*, pages 48–59. IEEE, 2006.

[7] J. R. Goodall. Introduction to visualization for computer security. In John R. Goodall, Gregory Conti, and Kwan-Liu Ma, editors, *VizSEC 2007*, Mathematics and Visualization, pages 1–17. Springer Berlin Heidelberg, 2008. 10.1007/978-3-540-78243-8_1.

[8] John R. Goodall, Gregory J. Conti, and Kwan-Liu Ma, editors. *VizSEC 2007, Proceedings of the Workshop on Visualization for Computer Security, Sacramento, California, USA, October 29, 2007*, Mathematics and Visualization. Springer, 2008.

[9] Ivan Herman, Guy Melançon, and M. Scott Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6:24–43, January 2000.

[10] Noah Iliinsky Julie Steele. *Beautiful Visualization*. O'Reilly Media, Inc., 2010.

[11] Noah Iliinsky Julie Steele. *Designing Data Visualizations*. O'Reilly Media, Inc., 2011.

[12] A. Komlodi, P. Rheingans, Utkarsha Ayachit, J.R. Goodall, and Amit Joshi. A user-centered look at glyph-based security visualization. In *Visualization for Computer Security, 2005. (VizSEC 05). IEEE Workshop on*, pages 21 – 28, oct. 2005.

[13] Kiran Lakkaraju, William Yurcik, and Adam J. Lee. Nvisionip: netflow visualizations of system state for security situational awareness. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, VizSEC/DMSEC '04, pages 65–72, New York, NY, USA, 2004. ACM.

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

[15] Yarden Livnat, Jim Agutter, Shaun Moon, Robert F. Erbacher, and Stefano Foresti. A visualization paradigm for network intrusion detection. In *In Proceedings of the 2005 IEEE Workshop on Information Assurance And Security*, pages 92–99. IEEE, 2005.

[16] Raffael Marty. *Applied Security Visualization*. Addison-Wesley Professional, 2008.

[17] Jonathan McPherson, Kwan-Liu Ma, Paul Krystosk, Tony Bartoletti, and Marvin 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*, VizSEC/DMSEC '04, pages 73–81, New York, NY, USA, 2004. ACM.

[18] Toby Segaran. *Programming Collective Intelligence*. O'Reilly Media, Inc., 2007.

[19] Colin Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, 2004.

[20] Christopher D. Wickens, Diane L. Sandry, and Michael Vidulich. Compatibility and resource competition between modalities of input, central processing, and output. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 25(2):227–248, 1983.