From A to Z: Developing a Visual Vocabulary for Information Security Threat Visualisation

Eric Li[1,2], Jeroen Barendse[1(✉)], Frederic Brodbeck[1], and Axel Tanner[3]

[1] LUST, The Hague, Netherlands
{eric,jeroen,frederic}@lust.nl
[2] Princeton University, Princeton, NJ, USA
eyli@princeton.edu
[3] IBM Research – Zurich, Rüschlikon, Switzerland
axs@zurich.ibm.com

Abstract. Security visualisation is a very difficult problem due to its inherent need to represent complexity and to be flexible for a wide range of applications. As a result, many current approaches are not particularly effective. This paper presents several novel approaches for visualising information security threats which aim to create a flexible and effective basis for creating semantically rich threat visualisation diagrams. By presenting generalised approaches, these ideas can be applied to a wide variety of situations, as demonstrated in two specific visualisations: one for visualising attack trees, the other for visualising attack graphs. It concludes by discussing future work and introducing a novel exploration of attack models.

Keywords: Visualisation · Security · Model · Attack tree · Attack graph

1 Introduction

Understanding and assessing security threats has always been a key challenge to security practitioners, exacerbated by the emergence of the digital era and the increasingly intricate systems that make up the information security landscape. As a result, developing methods that distill vast amounts of data into consumable visualisations or diagrams that are both engaging and informative remains a critical problem. Many security models[1] can be thought of as a giant machine with tens, or even hundreds of levers and dials that must all be precisely calibrated in order to model each specific security scenario. This is particularly true of the models being developed for use in the TRE$_{S}$PASS project[2], which aims to

[1] A model is defined as “a simplified description, especially a mathematical one, of a system or process, to assist calculations and predictions” (Oxford English Dictionary). When discussing visualisation of said models, it is in regards to making this abstraction visible in some manner.

[2] Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security http://www.trespass-project.eu.

B. Kordy et al. (Eds.): GraMSec 2016, LNCS 9987, pp. 102–118, 2016.
DOI: 10.1007/978-3-319-46263-9_7

create socio-technical models for security and risk estimation and in which this research was performed. Because of the increasing complexity of security models, it is well worth spending the time to develop visualisations that complement such models.

Visualisation is defined as (i) the formation of mental visual images and (ii) the act or process of interpreting in visual terms or of putting into visible form.[3] As it relates to information security, both senses of the word carry equal weight. Visualisation is used not merely for aesthetics, but also to aid practitioners and end users in forming mental models by providing a visual aid for the data presented in a security model. Because the data presented in models such as Attack Trees [14] tends to be tedious and rather complex, it is important that a visualisation provides an abstraction that reduces the complexity while still maintaining sufficient semantic detail.

Ideally, the visualisation also creates a narrative of the data in a consumable or even actionable manner. It has the power to strongly influence a viewer's perception of a model and therefore requires careful consideration of all the dimensions presented (or not presented). However, many traditional visualisations that have been used to visualise security models tend to fall short of this goal, allowing visualisation of only two, sometimes three, parameters of this multidimensional data. While they do a good job illustrating the relationship between some aspects of their respective security models, attempts to visualise a complete model or any more aspects will not be as simple.

In the next section, we present an overview of current approaches for attack visualisation, explain our approach in Sect. 3, and provide examples of using this approach in Sects. 4 and 5, followed by conclusions and ongoing explorations in Sect. 6.

2 Survey of Current Approaches to Attack Visualisation

In the beginning of the TRE$_{\text{S}}$PASS project, one of the first steps was a survey of state-of-the-art information security risk visualisations [3].

In general, information security visualisations depend very much on the purpose (exploratory versus explanatory), topic (financial risks, environmental risks, computer security etc.), and target audience, with very different levels of abstraction in presenting the vast amount of data typically available for the systems under consideration. Therefore they range from dashboard-like presentations of overall system state for awareness in an operations centre, to tools for investigating very specific technical details, like packet flows across networks, for deep diving into available data. This breadth makes it difficult to survey the complete field. Some summarising reviews can be found in [6,13] and [8], for more general risk visualisations and in [12] for the more technical oriented visualisations in computer security.

Here we focus on a critical review of tools currently used by security practitioners such as Carnegie Mellon's OCTAVE [1] and Siemens' CRAMM [2].

[3] As defined by Merriam Webster.

These findings, which serve as background and motivation for a new approach, are summarised here.

Information security visualisations have traditionally been used to display the degree of impact, measure of risk, and value of assets. Tools similar to those mentioned previously use visualisations that map assets to threats and vulnerabilities, and often appear in dashboards. These visualisations cover a wide range of graphic outputs, including visual metaphors to convey certain portions of their security model. However, in most cases, visualisation approaches focus more strongly on functional implementation and interaction than on aesthetics and the narrative defined by the aesthetics. Many of these visualisations typically do not provide a narrative for the motivations of attackers and defenders and consequently require the users to perform their own analysis to draw meaningful conclusions.

Often, visualisations depict the information security attack surface as having only two parameters, requiring researchers to sometimes oversimplify a model to represent it. However, in doing so, crucial interrelationships between actors and elements of the system are not shown, and there is a risk of misrepresenting the data or portraying it in a way that causes the viewer to misinterpret it. As a result, it would be beneficial to explore methodologies of extending existing visualisations to support higher dimensionality, more depth and influencing connections. Visualisations must therefore be flexible, supporting visualisation of individual aspects of the model as well as the model in its entirety. This does not necessarily mean that practitioners should aim to visualise all available data, but to choose the most important aspects to convey the message, visualise this in the most relevant and convincing way, and/or allow viewing from different perspectives.

3 A Parameterised Approach

3.1 Defining a Security Language

Taking language as a metaphor, before writing sentences and paragraphs, one has to first define an alphabet. This alphabet is the domain from which everything else in the language is formed. The richer the words, the more eloquent the sentences that can be written. We are therefore in the process of learning how to 'speak' by developing a language and using the corresponding 'characters' to express the aspects of information security: transforming data into information, information into knowledge, and knowledge into insight. All of these elements themselves can be relatively simple and straightforward. But when combined in an open-ended manner, they can convey complex information.

More concretely, this means to take the whole of a model and break it down into its most elemental components. It is sensible to identify elements or parameters of a model that have an effect on its outcome. This allows the assignment of a certain hierarchy or ranking to the elements, depending on how influential they are, i.e., picking and choosing the particular aspects to focus on and visualise. For example, a typical step in the TREsPASS Attack Tree, part of its security model, has multiple parameters such as cost, time, probability, and difficulty. These all can affect the outcome of the risk analysis for the model and thus form the basis of the attack tree language.

3.2 A Visual Vocabulary

A visual vocabulary is composed of a set of symbols or graphics that function as building elements for describing larger, more complex visual entities. A strong visual language forms an important basis for representing security models as it provides a suitable mapping from the model's language to the visual vocabulary. This vocabulary should also be extensible, allowing one to highlight and visualise particular parts of interest, including uncertainty. Although uncertainty may seem to go against highlighting important aspects in a security model, both are critical for the understanding and evaluation of information, and should be developed in concert with the remainder of the vocabulary. The core of any visualisation is the selection and development of an effective visual vocabulary and a mapping, or legend, that supports it. Such visual vocabularies are often aided by the principles of *Gestalt* psychology.

Gestalt and Visual Thinking. The overall appearance and qualities, or *Gestalt*, of a visualisation are very important properties. *Gestalt* is a term from psychology defined as the 'unified whole'. Being aware of and implementing the principles of *Gestalt* theory in a visual language renders visualisations stronger and more informed. These theories of visual perception were first developed by a group of German psychologists [10, 11] in the 1920s and describe how people tend to organise visual elements into groups. Although there are certain faults with some Gestaltist assumptions [16], it is important to be aware of those principles in order to use and, at other times, also to creatively mis-use them:

Similarity. The principle of similarity states that things sharing visual characteristics such as shape, size, colour, texture, value or orientation will be seen as belonging together.

Continuation. The principle of continuity predicts the preference for continuous figures.

Closure. The principle of closure applies when viewers tend to see complete figures even if part of the information is missing.

Proximity. The principle of proximity or contiguity states that things which are closer together will be seen as belonging together.

Figure and Ground. The terms figure and ground explain how viewers use elements of the scene which are similar in appearance and shape and group them together as a whole. Similar elements are contrasted with dissimilar elements (ground) to give the impression of a whole.

Pre-attentive Variables and Layering. Pre-attentive variables operate mostly at a 'subconscious' level; people recognise trees, tables, and maps, and immediately process the underlying data according to the first impressions

> gained without any conscious analysis of actual data. Encoding via pre-attentive factors relates to the general graphic design concept of 'layering'. When looking at well-designed graphics of any sort, different classes of information are perceived on the page. Pre-attentive factors like colour cause visuals to perceptually 'pop out,' and any sense of similarity causes them to be seen as connected to one another, as if each were on a transparent layer over the base graphic. This is an extremely effective way of segmenting data, where each layer is simpler than the whole graphic, and the viewer can study each layer in turn, while relationships among the whole are preserved, emphasised, and therefore are brought seamlessly to the analyst's attention.

Pre-attentive variables, combined with certain cultural habits (the colour red indicates stop or dangerous), can already lead to a basic understanding of a visualisation by viewers. Therefore it is important that these pre-attentive variables and habits correspond to rather than contradict the mapping chosen. To be able to fully understand a visual vocabulary developed, every visualisation needs a legend (which is in fact derived from the latin word *legenda*, meaning "the things that need to be read"). A legend or key is often a box in the corner of a map or visualisation, and is essential for understanding a visualisation. Having an effective legend is crucial because it requires the designer to establish clear goals in order to provide a clear mapping from the language of the model to the visual vocabulary. It defines which aspects of the model are represented and how they appear visually. As the legend is actually the most important element on a visualisation, it should be the starting point for each project. This means that the legend need not be a box in the corner, but can be a part of the information presented and as integrated as possible.

3.3 Approaches for Developing Visualisations

Parameterisation of Visual Elements. Every type of visualisation or graph will contain graphic elements whose rendering is modified by variables. A circle, for example, has several variables, such as position, radius, fill and opacity, that affect the final visual outcome. One can think of these variables as knobs that can be adjusted depending on a certain input. This way of thinking allows the creation of nearly unlimited combinations of visual elements.

For example, directed graphs[4] are used heavily in security models because of their ability to represent complex network relationships between entities. These graphs consist of edges and vertices, visually often represented by the two essential elements of a line and a circle. Feedback from early explorations found that it is most visually effective to parameterise at most three variables simultaneously. A line can be defined, for example, by its thickness, colour, and opacity. By applying a mapping from a derived security vocabulary (e.g. difficulty, time, and

[4] Here we discuss graphs in the mathematical not the visual, context. So a graph in this context is defined as a network of vertices or nodes connected by (directed or undirected) edges.

probability) to the visual vocabulary, it is possible to begin building the framework for visualisation. Thickness can be mapped to indicate difficulty, colour to indicate time, and opacity to indicate probability. Note that certain properties are more suitable for mapping to certain visual parameters. It does not, for example, make sense to map time to opacity because a lower opacity implies a weak link, whereas time only indicates the length to complete. More examples can be found in [5]. All of these parameters combined lead to a very rich, yet simple visualisation that leaves no visual 'stone' unturned. One can apply a similar approach to visualise information in a node.

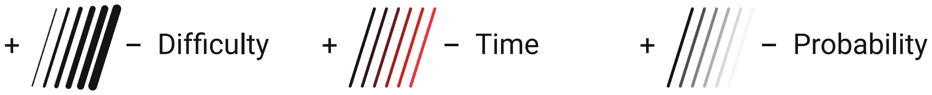


Fig. 1. Legend for attack trees; difficulty is indicated as stroke width; time is indicated as stroke colour; probability as stroke opacity. (Color figure online)

Stacking Visual Elements. There are often cases where the number of parameters that need to be visualised far outnumber the sensible variants to a particular visual element. In other cases, it may not be known what the total number of parameters is (as they may shift depending on user input). An example might be an entity, such as an attacker profile, for which the user has the freedom to pick and choose the parameters to assign. In such cases, it makes sense to develop a language that is extensible. An approach in which visual elements are stacked provides a solution to the problem, as it can be applied to a wide range of parameters.

First, it is important to establish a unified legend, where, for example, line thickness and colour are used to indicate levels of risk. Quite often parameters in security models can be mapped to a scale that ranges from low to high risk. This means that a visual element becomes a generalised module for visualising, allowing for adaptability and re-usability. Attacker profiles are a good example because the number of parameters change depending on the situation. Intel provides a good set of baseline attacker profiles in [5]. But there are cases where perhaps some parameters may not matter. As a result, it is necessary to create a visual system that allows for this. By using a unified legend, as described above, where thickness and colour can represent threat level, it is possible to represent an attacker profile as a set of stacked circles, in which each parameter is one of the circles (Fig. 2). This technique allows extensibility if say, later on, a situation calls for an additional parameter by providing the ability to stack an additional circle. Again, it is important to pay attention to visual hierarchy as parameters that are closer to the outside of a circle are weighted as visually more important. This can be compensated by arranging the parameters in order of importance, or preference, from inside to outside.

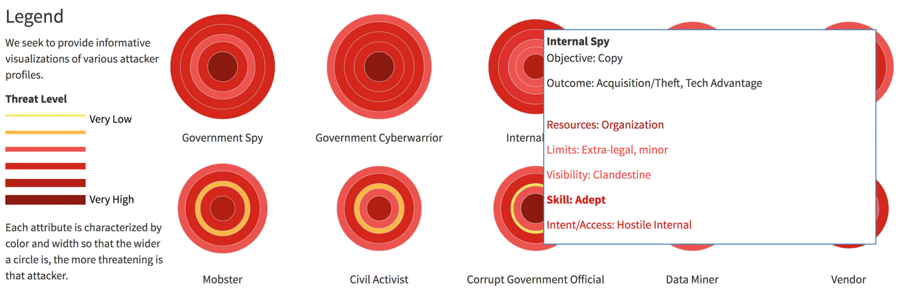

Legend
We seek to provide informative visualizations of various attacker profiles.
Threat Level
Very Low
Very High
Each attribute is characterized by color and width so that the wider a circle is, the more threatening is that attacker.
Government Spy
Government Cyberwarrior
Internal
Internal Spy
Objective: Copy
Outcome: Acquisition/Theft, Tech Advantage
Resources: Organization
Limits: Extra-legal, minor
Visibility: Clandestine
Skill: Adept
Intent/Access: Hostile Internal
Mobster
Civil Activist
Corrupt Government Official
Data Miner
Vendor

Fig. 2. Legend for stacking circles, and resulting visualisation of attacker profiles. Demo at: http://lustlab.net/dev/trespass/visualizations/profiles/. (Color figure online)

Semantic Zooming. Often, security visualisations will either be too simple in their attempts to abstract a model, or too complex and confusing by trying to show all the data. An approach to solve this issue is the idea of semantic zooming, which applies meaning to different zoom levels. Semantic zooming is an approach that displays different, semantically relevant, information at different levels of zoom on a visualisation. More abstract representations can be used at a macro view, whereas all the complex intricacies can be shown in a micro view. This is similar to the appearance of additional details when zooming into an online map.

This approach complements stacking visual elements quite well as it allows elements to be seen only when such detail is required. For example, when viewing attacker profiles from a macro view, it is only important to show the total perceived threat that an attacker has. As a result, the attacker can be represented by a single circle whose radius is the sum of the stacked circles. Only when zoomed into a more detailed view, where a viewer might want to inspect how parameters differ between attackers, does the visualisation reveal the individual stacked elements. By displaying this detail only when necessary, it is possible to create visualisations that can be relatively simple without any loss of information while still allowing the data to be understood in detail through taking a closer look.

Multiple Views. Every visualisation foregrounds certain aspects of the data it is representing, while backgrounding other aspects. As not all viewers are interested in the same aspects, multiple views offer a good way to cater to this. In addition to multiple views on the same data, it is also possible that a certain model can be analysed by various tools. Each of these tools provides its own outcomes and therefore needs its own visualisation, because each view tackles a certain aspect of the security model. Viewed as a whole, often in a dashboard-like view, they paint the entire picture of the visualisation.

Contextual Awareness and Highlighting. A key aspect in security visualisation is the ability to highlight key points of vulnerability. This tends to be much more effective than just textual output, as it also gives viewers the ability to contextualise potential points of interest in the model. Oftentimes there are analytical tools that can provide insights such as the weakest or cheapest path in an attack tree. It may also make sense to highlight certain connections based on user interaction. It is also important to consider how and when certain elements will be highlighted when developing a vocabulary. Depending on what needs to be highlighted, certain approaches may be better than others. Contextual awareness also allows a fine-grained representation of information without overwhelming the viewer.

As mentioned, visualising uncertainty is often just as important as the data itself, as it allows viewers to understand the accuracy or the fuzziness of certain factors of the model. Existing work in visualising uncertainty can be found in [7, 9]. Possibilities to visualise uncertainty include transformations such as blurring visual entities, or introducing a way of displaying multiple possible predictions of a model, similarly to how a user might choose to highlight a certain path or visual element.

4 Application to Attack Trees in the TRE$_S$PASS Project

Attack trees, as developed and used in the TRE$_S$PASS consortium, are a tool to capture all possible attacks to reach a specific goal, as described in the root node. To build such an attack tree, experts typically gather and, starting from the goal node, try to enumerate possible ways of attack to reach this goal. Each subnode can be iteratively refined as far as it seems fit. Individual intermediate nodes can thereby be either conjunctive or disjunctive. A conjunctive node requires that all of its children be fulfilled in order to proceed up the tree, whereas disjunctive nodes only require one child to be satisfied in order to proceed. A complete path of actions consists of any number of leaf and intermediate nodes leading to the root node. Within the project, each leaf node is considered to contain four parameters: difficulty, minimum cost, probability of success, and minimum time required to complete. Depending on the effort put into the creation, these attack trees can be very complex, comprising hundreds or thousands of nodes, especially when they are generated programmatically from an underlying model as it is the aim of the project. When attempting to visualise these trees, such visualisations quickly become complex and unreadable.

In their traditional form, attack trees present a wide variety of important and relevant information, but are not easily visualised, oftentimes shown as an arrangement of text in a directed graph. From a visualisation perspective, attack trees have several flaws; the tree structure gets very wide rapidly, repeating lots of elements to eventually become effectively unreadable even in a medium allowing arbitrary zooming. Also, because attack trees consist of conjunctive and disjunctive nodes, it needs to become visually clear that in the case of conjunctive nodes, all steps need to be fulfilled in order to proceed. We can counteract the complexity by improving the way the tree is laid out and labelled, as well as by testing alternative layouts that result in more compact trees, while maintaining

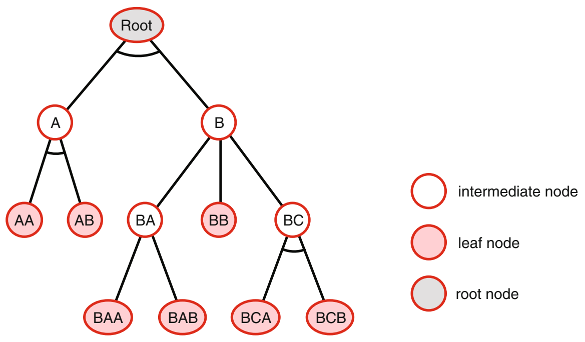


Fig. 3. Example of a typical structure of an attack tree.

readability. Next to that, exploring interactivity allowing the user to zoom and pan, and to collapse sub-trees at any level, makes it easier to concentrate only on certain parts of the tree (Fig. 3).

The key components and their respective properties are the following:

Node	Edge
Type of node (leaf, intermediate, root)	Parent/Child nodes
Conjunctive or disjunctive node (if intermediate or root node)	
Label	
Minimum cost to complete node	
Probability of success	
Difficulty	
Minimum time required to complete	

A visual language can be developed based on this. It is important to keep in mind that attack trees can vary greatly in size, as their construction is largely dependent on the scenario and environment that they are trying to model. As a result, the language should be scaleable to any size tree. In initial explorations, feedback revealed that when representing the graph with directed graphs, edges carry much more weight and information visually. Subsequently, most of the parameters were mapped to the edge leading to parent nodes. This allows focus to be placed much more on the path rather than each individual step. The resulting legend was developed by parameterising the visual properties of a line (Fig. 1) and creating a mapping to the attack tree vocabulary (Fig. 4).

4.1 Multiple Views

Visualising an attack tree in a tree structure may do a good job for displaying how the nodes are connected, but it does a poor job for examining frequency.

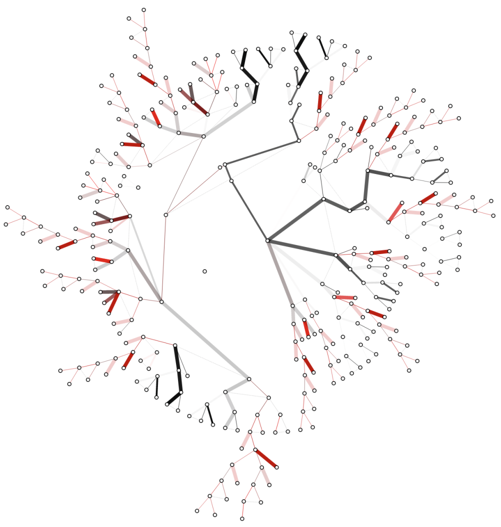


Fig. 4. Attack tree visualised in radial form where each node corresponds to an attack step. The root node, i.e., the goal, is placed in the center. The edges are coloured according to three parameters: difficulty is indicated as stroke width, time as stroke colour, and probability as stroke opacity. (Color figure online)

Therefore, it makes sense to split this into two visualisations: an attack tree visualisation structured as a tree, as well as a tree map visualisation that focuses just on the relative frequency of each node (Fig. 5). The frequency of the node determines the size of each box, while the colour depicts the relative difficulty of each node. A hover over each box in the tree map shows its label and highlights the nodes in the attack tree, allowing a user to understand the visualisation. Together, they paint a more complete picture. We consider the tree map as part visualisation and part legend.

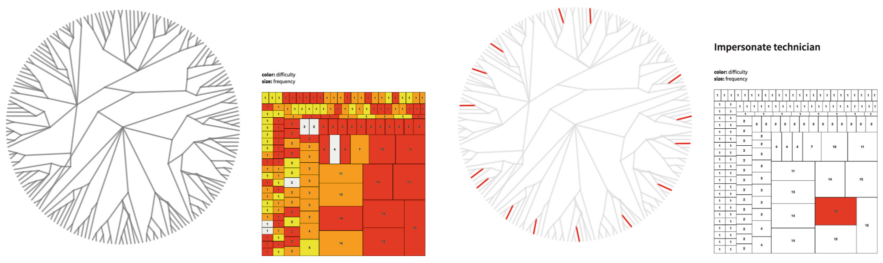


Fig. 5. Tree map visualisation that shows frequency of an attack step. (Color figure online)

4.2 Attack Tree Linearisation

More simple elements are better than fewer, complex elements. A tree works well in situations where the structure is fairly simple and small. However, the attack trees that are used in TREsPASS are already more complex than can comfortably be fit on a screen. Working with and studying attack trees from a visualisation point of view, one can question the role of intermediate nodes. Other than being a labelled container for their child nodes, they are not actually steps along the attack path but nevertheless occupy a large part of the attack tree. We can visually simplify attack trees by turning them into linear sequences of their required children. This will result in more paths, but each path will be easier to follow. The simplification and conversion to straight paths benefit readability from a visualisation standpoint. One path now shows a user the steps that need to be taken in a straight and easy to follow line (although it does not usually imply a temporal or causal sequence).

4.3 Stacking Visual Elements

Another legend was also developed in the case that additional parameters to each step may be needed. By mapping different visual elements (thickness and colour to threat level) of a line to a scale of threat, it is possible to modularise this element and stack it to any number of parameters.

Visually, this becomes just as effective as the original legend because a step in which all parameters have a high perceived threat level will stand out much more strongly than a step with a low perceived threat level. When combined to form a path, as in Fig. 6, this legend is very informative on which steps and connections are areas of vulnerability.

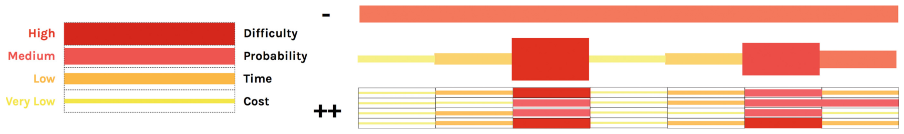


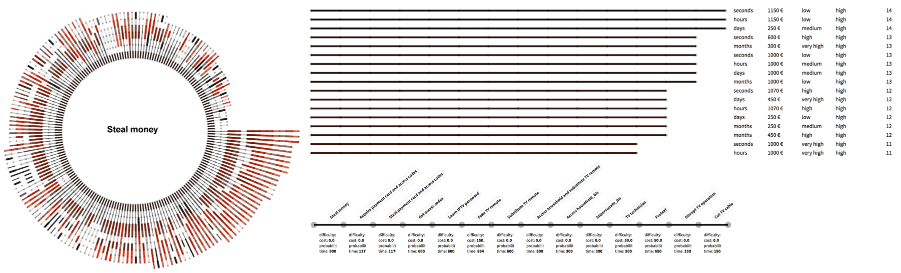
Fig. 6. Tree map plus radial attack tree visualisation, where the size in the tree map shows the frequency of an attack step and the colour indicates difficulty. Both views are linked in highlighting when selecting elements. (Color figure online)

4.4 Semantic Zooming

When applying the stackable legend developed in Sect. 4.3 to massive attack trees, the overall visual effect of the visualisation becomes confusing and harder to read. Viewers are not as able to follow paths as easily as before. However, semantic zooming can be applied in multiple ways. Because the stacked lines are only necessary at a very detailed level, it is perfectly fine to show the average threat level at a macro view with other paths or the entire tree (Fig. 6).

Only when zooming in to view specific paths will the individual stacked lines be revealed to the viewer. This eliminates the original complexity at a macro level while still allowing specificity at a micro level.

This can be combined with a rearrangement of the linearised attack trees to present the paths in a more understandable manner. By using a radial view for the linearised attack trees at a macro view and transitioning to a table, in which information about the total path can be displayed alongside each path upon user interaction, it can be possible to sort and analyse paths in a way that might otherwise be unwieldy at the macro level (Fig. 7). A viewer can then zoom in even closer to see an individual path and its stacked line components, as well as any intermediate labels that might not have been shown before.



seconds	1150 €	low	high	14
hours	1150 €	low	high	14
days	250 €	medium	high	14
seconds	600 €	high	high	13
months	300 €	very high	high	13
seconds	1000 €	low	high	13
hours	1000 €	medium	high	13
days	1000 €	medium	high	13
months	1000 €	low	high	13
seconds	1070 €	high	high	12
days	450 €	very high	high	12
hours	1070 €	high	high	12
days	250 €	low	high	12
months	250 €	medium	high	12
months	450 €	high	high	12
seconds	1000 €	very high	high	11
hours	1000 €	very high	high	11

Fig. 7. Semantic zooming applied to linearised attack paths. From left to right: an attack tree in radial and straight path form, and the same paths as a table allowing detailed inspection

5 Application to Attack Graphs

Attack graphs are a common tool used by security researchers to organise information on all possible attack paths within a certain space. Although they generally are adapted for custom use, the general idea is the same: there is a directed graph with a starting point and an end point (the goal), as well as nodes that function as attack steps or entities (Fig. 8) (as used in [4]). The edges are possible paths from one entity to another. These nodes and edges carry with them several parameters, such as probability, cost, and incident count.

For the remainder of this section, however, the attack graph referred to is the one defined by Verizon in their annual Data Breach Investigations Report (DBIR)[5] [15] as a test case to see whether we could also apply the visualisation principles laid forward in the preceeding chapters to graphs other than attack trees. For 2016, there are seven action groups, with multiple sub-actions, as well as three attribute groups, again with multiple sub-attributes. Within the graph itself, actions lead to other actions or to compromised attributes. Compromised attributes will lead either to the end of the breach or to another action by the attacker.

[5] Data link: https://github.com/vz-risk/VERISAG/tree/v2/static.

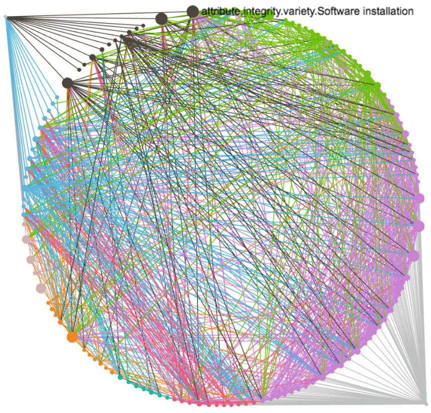


Fig. 8. Example attack graph as used in the online tool *2016 DBIR Attack Surface Analysis* [4].

5.1 Visualising Attack Graphs

There are several goals that the visualisation of the attack graph aims to achieve: (i) displaying and differentiating actions and attributes, (ii) displaying relative threat of nodes and edges, (iii) displaying paths, and (iv) displaying a comparison between different versions of the graph (either through mitigations or comparison with previous years' data). The principal flaw of traditional attack graph visualisations is that they attempt to visualise all nodes and connections at once. In cases such as the DBIR, this grows very complex and as a result, it becomes hard to perform even simple tasks, such as determining the relative importance of a node or discovering which nodes are connected. In fact, the version presented at [4] mostly serves to illustrate how complex the attack space is.

As the principal structure is a directed graph, it is possible to immediately identify the nodes and edges as elements of its vocabulary. Digging deeper, the following characteristics make up these elements:

Node	Edge
Type of node (action, attribute, start, end)	Type of edge (edge to attribute, action, or end)
Relative frequency of node (occurrences within incidents)	Relative frequency of edge (occurrences within incidents)
Sub-type of node (one of 5 actions or one of 3 attributes)	

The visual language begins with the same traditional elements of the directed graph: nodes and directed edges. Traditionally, these graphs are visually composed of circles that represent the nodes, and paths with arrows indicating direction.

To begin building a visual vocabulary, each of the elements is parameterised. Assigning radius and fill colour of the circle to represent frequency of incidents creates an aesthetically informative visualisation of the node. Textual treatment and visual treatment can then be applied to each circle to indicate the type of node. Rather than by drawing an arrow as a path, direction on edges can be shown by decreasing the stroke width of a path. This width can also be parameterised, as well as the opacity of the edge, to show how frequently that edge occurred within the incident space, resulting in the legend in Fig. 9. From this legend the visualisation[6] was developed based on the outlined approach.

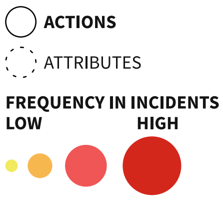


Fig. 9. Legend for attack graph visualisation. (Color figure online)

Arc Diagram. Arc diagrams were chosen because of their ability to clearly display multivariate data, even in complex situations. Building on the work of Wattenberg [17], these graphs provide an easy way to visualise connections by placing the nodes on the same line (Fig. 10). They also allow easy comparison of nodes and edges, and give viewers a more linear ability to visualise this data. The two-sided nature of this graph allows one to visualise edges going to attributes on one side, and edges going to actions on the other, providing a clear visual delineation between the two types of edges. When a comparison of different versions of the graph should be shown, all edges can be moved to one side and two versions of the graph can be shown simultaneously.

Semantic Zooming. Although visualisation of all 119 nodes in one view provides a good overview, this level of complexity is very hard to follow and unactionable. To improve this, semantic zooming can be applied. The initial view now becomes the eight macro categories of actions and attributes. This presents a good overall sense of how certain attacks paths might be structured, as well as the relative frequency of certain action or attribute categories within the attack space. If viewers want to learn more about the composition and frequency of each attack or attribute, they can click on it for a view containing all subgroups (Fig. 11a). We choose to contain these sub-nodes within the overall node to visually show the hierarchy of nodes. At this level, viewers can see the count of each node and determine which specific sub-node presents a greater threat. This also applies to comparing graphs (Fig. 11b).

[6] The interactive version of the DBIR Attack Graph can be found at http://lustlab.net/dev/vzw/index.html.

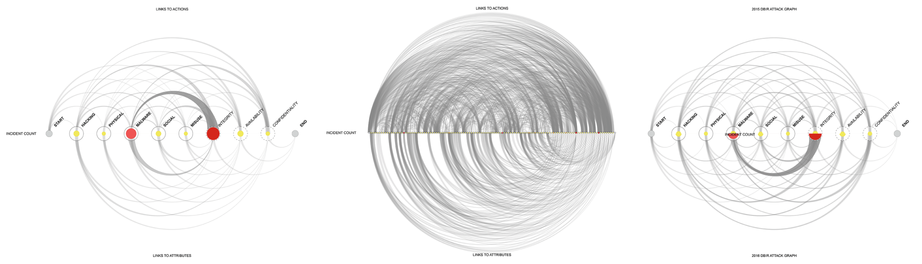


Fig. 10. Different visualisation views afforded by using an arc diagram. From left to right: macro-view of 2016 DBIR Data, all nodes of the 2016 DBIR, and comparison between 2015 and 2016 DBIR.

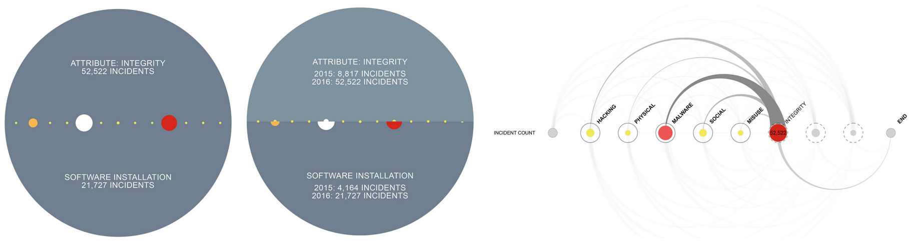


Fig. 11. From left to right: (a) Micro-view of 2016 DBIR data with highlighting, (b) Micro-view of 2015/2016 DBIR data with highlighting, (c) Highlighting a specific node and its target edges in the 2016 DBIR. (Color figure online)

Contextual Awareness and Highlighting. A level of interactivity is also built into the visualisation that allows the viewer to highlight certain aspects of the graph depending on the current zoom level. At the macro-view, hovering over a node reveals the incident count of that node within the attack space, and other child nodes (Fig. 11c). This allows viewers to pay attention to the context in which they are highlighting the node. All other nodes are greyed out for clarity, which further aids focusing on the relevant information. In the micro-view, when a user hovers over a node in the graph, the incident count at the bottom of the circle updates to also display information specific to the currently highlighted node (Fig. 11a). Visual feedback is also given by turning the highlighted node white to match the text colour. This contextual highlighting provides information to the user only when requested and presents the wealth of information in the graph in a non-overwhelming manner.

6 Conclusion and Future Work

Security-related visualisations have always been a particular challenge, due in part to the complex, multi-dimensional, and wide ranging nature of the field. Not only do models tend to be complex in order to capture the intricacies of a particular scenario, but their usage varies from highly specialised technical people requiring very specific visualisation to security researchers attempting to create layman infographics that aim to visualise an overall model, resulting in visualisations that are not quite effective because they try to do too much or too little. As designers begin to move into the era with interactive visualisations, we should consider how all of these differing aspects will play a role in our approach to security visualisation.

This paper presented an approach for developing effective visualisations of complex security models. By first outlining goals for visualisation and then breaking down a model and defining a visual vocabulary, it is possible to create a framework that allows nearly endless flexibility. From here, a visualisation can be built by applying some combination of the approaches outlined. These approaches offer ways of thinking about visualising complex data that are general enough to be applied to almost any type of visualisation situation, including those not related to security. We use attack trees from the TRE$_S$PASS project as well as attack graphs from Verizon's DBIR report as case studies to see how this parameterised approach can be used to create effective visualisations.

Early versions of this visualisation approach have been presented to two security practitioner feedback panels and the advisory board of the TRE$_S$PASS project. Based on their feedback, visualisations were adjusted and developed further. For instance, the stacking of visual elements is a direct result of feedback given, as many practitioners had a hard time combining more than two parameters in one visualisation.

Currently we are implementing features in the code so that external data can be loaded easily and can be applied to situations in which the input data is dynamic and changing frequently. In that way, this visualisation approach can cover an extensive range of media, from print to interactive and to time-based. We can also then see if the concepts and vocabulary developed are applicable to non-security-related sectors. The final code and designs will be made available as open source so that third parties can use and extend this work.

Acknowledgements. The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007–2013) under grant agreement ICT-318003 (TRE$_S$PASS). This publication reflects only the authors' views, and the European Union is not liable for any use that may be made of the information contained herein.

References

1. Alberts, C.J., Dorofee, A.: Managing Information Security Risks: The Octave Approach. Addison-Wesley Longman Publishing Co., Inc., Boston (2002)
2. Barber, B., Davey, J.: The use of the CCTA risk analysis and management methodology (CRAMM) in health information systems. Medinfo **92**, 1589–1593 (1992)
3. Barendse, J., Bleikertz, S., Brodbeck, F., Coles-Kemp, L., Heath, C., Hall, P., Kordy, B., Tanner, A.: TRE$_S$PASS Deliverable 4.1.1: initial requirements for visualisation processes and tools . Internal deliverable of the TRE$_S$PASS project. (2013)
4. Bassett, G.: Verizon Enterprise Solutions: DBIR Attack Graph Analysis, June 2015. http://dbir-attack-graph.infos.ec/

5. Bertin, J.: Sémiologie Graphique. Gauthier-Villars, Paris (1967)
6. Eppler, M.J., Aeschimann, M.: Envisioning risk: a systematic framework for risk visualization in risk management and communication (2008). http://www.knowledge-communication.org/pdf/envisioning-risk.pdf
7. Harris, R.L.: Information Graphics: A Comprehensive Illustrated Reference. Oxford University Press Inc., New York (1999)
8. Husdal, J.: Can it be really that dangerous? Issues in visualization of risk and vulnerability (2001). http://www.husdal.com/2001/10/31/can-it-really-be-that-dangerous-issues-in-visualization-of-risk-and-vulnerability
9. Kirk, A.: References for visualising uncertainty. http://www.visualisingdata.com/2015/02/references-visualising-uncertainty/
10. Koffka, K.: Principles of Gestalt Psychology. Harcourt, Brace and Company, New York (1935)
11. Koffka, K.: Perception: an introduction to the Gestalt-theorie. Psychol. Bull. **19**(10), 531–585 (1922)
12. Marty, R.: Applied Security Visualization, 1st edn. Addison-Wesley Professional, Boston (2008)
13. Roth, F., Eidgenössische Technische Hochschule (Zürich), Crisis and Risk Network, Schweiz. Bundesamt für Bevölkerungsschutz, Suisse. Office Fédéral de la Protection de la Population: Visualizing risk: the use of graphical elements in risk analysis and communications. 3RG report, Eidgenössische Technische Hochschule Zürich, Center for Security Studies CSS (2012). http://e-collection.library.ethz.ch/view/eth:6286
14. Schneier, B.: Attack trees: modeling security threats. Dr. Dobb's J. Softw. Tools 24(12), 21–29 (1999). https://www.schneier.com/cryptography/archives/1999/12/attack_trees.html
15. Verizon Enterprise Solutions: 2016 Data Breach Investigations Report. Technical report, Verizon (2016). http://www.verizonenterprise.com/verizon-insights-lab/dbir/
16. Ware, C.: Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc., San Francisco (2000)
17. Wattenberg, M.: Arc diagrams: visualizing structure in strings. In: IEEE Symposium on Information Visualization, 2002, pp. 110–116. IEEE. (2002)