Modeling Changes in States of Computer Systems for Security

Vert, G., Harris, F., Nasser, S.

Dept. of Computer Science, University of Nevada-Reno, Reno, NV 89557

Summary
Traditional state modeling techniques have several limitations. One of these is the limited ability to model a large number of variables simultaneously. Another limitation is that understanding such models can often be difficult as they are often represented as a series of graphs with a large number of state transitions. In order to address the above issues, we propose a geometric approach to modeling state changes in computer systems and data. Geometric models lend themselves to mathematical combination and algebras and can encode large amounts of complex, interrelated data. Such data can be quantified, can be manipulated, and is something that human beings naturally process well. Visual state models can be utilized in a wide range of activities from monitoring system operation, security and data authentication. In this paper, we present a discussion of visualization model which can be useful in performing many of the activities used in security and operations of a computer system.

Key words:
Input here the part of 4-5 keywords.

1. Introduction

Operation and security of a computer system often requires processing large quantities of audit data, making it both computationally expensive and error prone [2]. To reduce computational requirements to a realistic level, administrative management of systems typically focus on a limited set of system and user attributes [3, 4].
Many types of sub systems in a computer can need monitoring among these are the network, processor and file systems. Very little work has been done in this area. One such system FABS provides a user interface and for monitoring anomalous accesses to a computers file system [5]. This system was conceived to deal with security threats and intrusion in a computer system. Visualization of system information can be a powerful mechanism for comprehending complex and varied data. As an example, in the summer of 2000, the National Safe Skies Alliance awarded a project to the Applied Visualization Center at the University of Tennessee to develop a 3D computer tool to assist the US Federal Aviation Administration security group, in evaluating new equipment and procedures to improve airport checkpoint security. To date, several detection models have been developed for a few airports [6]. This effort demonstrates the broad range of areas that visualization of system operation can be applied to. However, a dynamic and comprehensive view of system operation especially across a network of computers is hard to encapsulate with just a few variables. Further more, system operation data has traditionally been presented in text form, which, given the typical volume, is difficult for administrators to process. Limitations of traditional administration techniques are as much a function of the ability of a human to process large amounts of information simultaneously as they are limitations of the techniques themselves.

In order to improve the capability and performance of systems administration two things are needed:

- A model which can encode and represent complex computer systems data in a unified manner.
- An approach that presents information in such a way that relationships between components may be clearly seen.

To balance the need to examine large quantities of data with the difficulty of comprehending such quantities, we propose a geometric approach to modeling state changes in system data with a visual component. Geometric models lend themselves to the mathematical algebras of combination and prediction. Visually presented information can encode large amounts of complex, interrelated data, can be quantified, can be manipulated, and is something that human beings naturally process well. In this paper, we define a geometric model based on vector mathematics which appears to be useful in administration of computer systems, security operations and authentication of certain types of data.

2. Overview of Spicule Visualization

One important characteristic of a Spicule model is that it is not a static model. In other words, Spicule states change over time. Such a property is the basis for modeling state changes in a system. This allows for blending the visually intuitive with the mathematically quantifiable.

The concept of a Spicule was originally developed for intrusion detection [7] but has never been generalized to

Manuscript received January 5, 2007
Manuscript revised January 25, 2007

system state modeling. A spicule is an spheroid geometric primitive. One spicule represents the system model for one computer in a network of computers. Its volume is determined by the security fitness measure of the system which is utilized in the radius of the generated spheroid. A security fitness metric is a weighted sum of factors that contribute or detract from the vulnerability of a node. Systems with high security fitness metrics will thus have smaller spicules than those deemed at higher risk, essentially magnifying the most likely candidates for potential compromise. Because spheroids have volumes and radii, the security fitness is a mathematical property of a node that can be used in discovery of equations used to describe system state ergo stable or compromised configurations.

In addition to the central spheroid of the visualization, Spicules incorporate feature vectors, each modeling a facet of system activity. There are two types of feature vectors associated with a spicule. The first of this is referred to as a tracking vector and is utilized to represent system state data that can be mapped onto an interval ranging between 0% - 100%. As an example of a tracking one might consider CPU utilization, which can never exceed 100%.
A second type of vector is referred to as a fixed vectors. These vectors an not be mapped to a range but have the mathematical property of:

$T_v = [\ 0,,,\ \alpha\ ]$

Tracking and fixed vector behaviors differ. A tracking vector grows from the equator and is in a
plane parallel to the ground. As the feature value increases, the tracking vector travels along a track on the surface of the spicule to the vertical
which represents 100% of the feature's value. Because this vector is a normalized vector, a mathematical signature may be computed from the angles of the tracking vectors.

Fixed vectors in contrast are located at the equator of the spicule, coplanar to the ground and they grow in magnitude parallel to the ground. A fixed vector feature might be the representation of number of child processes spawned for a program. Once can see that there is no upper bound of 100% on this type of value and therefore fixed vectors can grow in magnitude infinitely. As such fixed vectors provide magnitude of feature value information. The three components of the spicule, security fitness, fixed vectors and tracking vectors hybridize current approaches to visualization of state information and changes in state information for a system. Application to determination of state changes can be done in the following fashion. The angles and magnitudes of any vector can be established for a normal state system and therefore thresholds, can be established. Secondly a set of tracking vectors with characteristic angles or fixed vectors with magnitudes for either normal or abnormal systems can be determined, providing a geometric signature. Comparing a system's model with this set of signatures lends itself to implementation of traditional state transition techniques.

In addition to the concept of classes of vectors representing state information in about a computer system, the spicule has the ability to model an infinite number of state variable at any given moment in time. This is a drastic improvement of current state transition models that often model a transition based on the change in a single variables value. For instance if we model vector locations on the spicule to be discrete integer values, and the spicule is composed of only tracking vectors, the number of variable we can model is 360 and the number of states for the 360 variables is given as:

$360 * 180 \rightarrow 64800$ state locations

If we model tracking vectors on the real number system R the number of states and variables that can be model simultaneously becomes:

$R * R \rightarrow \alpha$

The extremely large modeling capacity of Spicule dictates that it have a very rapid method of communication with humans and thus we use a visualization because of the large amount of information content that can be conveyed to a user looking at a visualized state model. Because spicule is based on vector mathematics, a given spicule can be operated upon with vector algebra to eliminate unchanged information and display only information that has had state changes. The application of these properties ranges from intrusion detection, data authentication, system administration to creation of visual taxonomies of the characteristics of an attack on a computer system. A representation of a Spicule model can be seen in Figure 1.

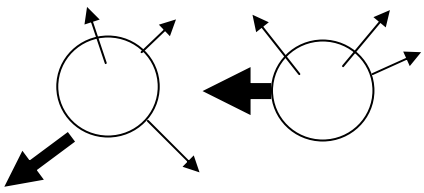

Figure 1: Polar and side view of Spicule

Where the thick arrows are fixed vectors that grow in magnitude and thickness and the thin arrows are vectors that track from 0 degrees to 90 degrees on the spheroid.

As mentioned previously Spicules are expected to change states over time. Therefore we introduce the notation At_i which represents a Spicule state model for node A at time t_i. The state of the spicule at t_0, As such we can have the notation of a set of state changes for a system represented as:

$A = \{ At_i \ldots Atj\}$

Additionally regions on a Spicule can be partition into categories of related information. As an example consider a spicle with four information classifications are regions in the upper hemisphere classified as system resources vectors, user processes vectors, system files vectors and system processes vectors. These quadrants may vary in number and type depending upon the information being represented. The appropriate set of feature vectors is dependent on those system characteristics which are deemed most useful in modeling to a user.

The process for updating vector state information should be and automated. This means employing some sort of auditing tools to continuously collect data and send it to the Spicule's visualization software where it can be dynamically rendered as a member of set A. As an example, the process of state update and visualization might have the following steps:

1. The feature vectors precess towards the vertical visualized as a percentage of maximum value to 100%. For example, if CPU usage was 50% at time t1, the vector representing CPU usage would be visualized at 45 degrees along a track to the north pole of the spicule,

Vector around the 360 degrees of the spheroid provide characteristics angles that can be measured for particular data feature vectors. Characteristic angles provide a geometric `signature' for ongoing activity.

2. Vectors can combine based on membership to in the same category. The recombination property allows for geometric signature generation. For example, the System Resources category might have the following members:

CPU usage	4% change/sec
CPU throughput	5% change/sec
port 25 traffic	1% change/sec
number writes	.05/sec
number reads	2% change/sec

Recombination we might combine the three vectors with the highest rate of change over time or it might combine (add) all vectors in the category. The resultant vector is shown in Figure 2.

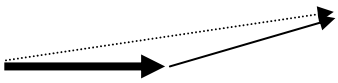

Figure 2. Recombined feature vector

Vector recombination provides additional characteristic signatures of system operation. In conjunction with the precession of vectors to the vertical prime axis and 360 potential features that can be monitored and visualized to measure for changes in normal and abnormal system operation. In the example shown in figure 3 we see:

- vector recombination by category
- precession by activity to the vertical which represents maximum activity per feature vector.
- development of a set of characteristic angles for $System_N$ spicule $\{ o', o'', ...\}$

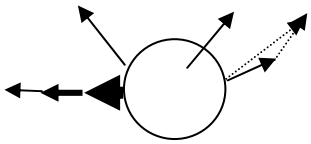

Figure 3: Spicule and vector recombination at t_n

Collectively, all the geometric information in the spicule for $System_N$ provides a model of system N. These configurations of feature vectors correspond to a series of mathematical equations describing each vector of the form:

$V_n = \{ id, mag, angle, circdeg, vertdeg\}$

where:

id - identifier of the vector
mag - magnitude of the vector
angle - angle vector points off coplane
circdeg – degree location around circumference

of vector
vertdeg - vertical degree location of base of
tracking vector tail on Spicule's surface

The equations can be subtracted or added to the same vectors equation at time t_n to measure changes in state of the vector being modeled. This collectively becomes a model of state change in system operation.

3. Spicules Additive Properties

Spicule borrows the concept of addition objects together to create new spatial object from geography [1]. Because a Spicule is a graphical object, it can also be added to another object to create a composite. What this means in practical terms is that two Spicules representing different states in a system can be added together to create a new composite state model. This property can be utilized to predict a new state based on a current state and some anticipated changes in the systems state variable data.

The implications of Spicule tessellation are powerful. The first of these benefits is that spicules for a system can be created from building block Spicules. These can be very accurate representations of the system. Then an administrator can pose the question of given certain changes in a system that may represent for instance an attack on the system, what will the system look like if it has been attacked. As an example, if a model of a bacteria attack is desired, one of the markers of such an attack is the rapid creation of child processes all with the same parent process id. A spicule created simply of a feature vectors that models user A's process activity thus is able to detect if user A mounts a bacterial attack using a small spicule but accurate representation of changes in system process states.

Such a model might have a single feature vector and is therefore highly accurate. If user A mounts the attack, his single feature vector will move. If one has such a model for user B, using tessellation to add the two spicules produces a composite spicule that is highly accurate at detecting bacterial attacks for users A and B. In theory if A and B are the only users on system X, what has now been constructed via the concept of tessellation a model that is 100% accurate at detecting bacterial attacks.

By iterative tessellating and composing of spicules within a particular category, e.g., bacterial attacks, one can create a highly accurate Spicule for visualizing and identifying that particular type of attack. The ability to build tessellated Spicules also allows one to create a model of system operation and a characteristic taxonomy for various categories of attacks. The tessellation property of spicules also implies the ability to predict attacks and to determine what attacks will look like on systems that have never been attacked. For example, if system N has had a bacterial attack, the attack will have occurred during a certain temporal frame. In this case the pre-attack configuration was at Nt_m and post-attack configuration was at Nt_n. Using a subtractive property of the Spicule. that (Nt_n - Nt_m) produces a spicule that reflects only the essence of a state change, the essence of an attack in the form of a Spicule can be visualized and identified.

This is referred to as the attack form of the spicule. The attack form spicule only contains the additive vectors that created the post attack form. Thus, it is portable and normalized. To predict attacks on target systems, the attack form can be tessellated to the spicule of an target system. The result is a view of that system as it will look after it has been attacked, a predictive spicule. If one knows how their system will look after an attack that has never experienced before, then they can monitor for changes in you're the current system state Spicule that tend to appear how a system might look after a given attack.

4. State Modeling In A Network Using Spicules

The spicule was conceived to model a local systems operation, but with the advent of the Internet, the equally important aspect of modeling network state changes can be addressed through the use of Spicule.

When one considers the relationship of systems to each other, it is clear that relationships have a geometric component to them, These relationships can be expressed in logical terms. which means that computers have a logical connection or relationship to other computers. Such a relation can be found when several computers connect to the same DNS server for address resolution and lookup. This relationship can be expressed as a function of which computers a given system predominately shares information with. To share information, a computer must therefore have a ``knows" relationship with other computers. Such a relationship can be described by an email address, client server relationship or an IP address etc.

Patterns of computer associations occur along these ``KNOWS" relationships. Therefore it is appropriate to design models that incorporate this fact when attempting to do model state changes in a set of related computers. The spicule can not do this by itself because its geometry is relative to itself. To solve this problem, a Spicule for a

given node in the network can be collapsed into what is called a singularity and incorporated in a singularity model. This singularity is located in 3D space by its IP addresses on the x, and z access and by a mathematical quantity derived from the spicules vector angles on the y axis. This quantity is initially thought to represent system activity as a whole.

This singularity model is used in conjunction with system Spicules to visualize state changes in a system of computers Within the Spicules singularity is a world of activity and quantifiable information. The singularity model then captures the relationship among other spicules via the ``KNOWS" relationship. This model is shown in

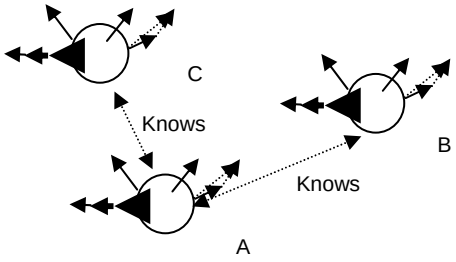

Figure 4. Sample singularity model

In the above example, for a unit of time, system A KNOWS system B, and has a lot of communication between. System A has a lot of activity, modeled by its spicule singularity and therefore may be a server. There is also a KNOWS relationship from A to C. The KNOWS relation implies that state changes in one Spicule affect state changes in another Spicule. Such a relationship can used to model for example the spread of a worm across the internet or a distributed database system where work is migrated from one system to another.

The metrics that might be used to detect model state change and its effect on other systems can be varied depending on what is being modeled. Among some possible metrics that might be part of such a model are:

- the number KNOWS relationships between nodes
- the ratio of number KNOWS relationships between node n, n+1 / a strength metric of KNOWS relationships

As part of the research into the singularity model future research will work on the development of a series of equations that will be relevant to modeling relationships among Spicule singularities.

5. Future Directions

This initial research presents some initial theory and conceptual development of a system to model state changes in a computer system. It also presents some initial work on the extension of such a model to consider the relationship among Spicule systems located on distributed computers. Some of the benefits of such a model are:

the ability to model n dimensions of state data simultaneously

the ability to comprehend data rapidly in a visually intuitive fashion.

The application of such a model ranges from system administration to authentication of data with spatial components to administration of system and the construction of visual taxonomies of constituent Spicule models.

Much work remains to be done to validate and further test this model. We need to develop a better understanding of how to apply Spicule to some of the potential areas mentioned previously. Additionally, further work needs to be done on developing metrics that model relations among spicules. Finally data in a computer system is typically ambiguous. For example it may sometimes mean one thing and at other times mean another. Future work will concentrate on the application of fuzzy logic to spicule and what the semantics of such an application might mean.

References

[1] Chrisman, N., Exploring Geographic Information Systems John Wiley and Sons, 1997.
[2] Heberlein, L., Dias G., Levitt, K., Mukherjee, B., Wood, J., Wolber, D., A network security monitor Proceedings of the Symposium on Security and Privacy pages 296-304, May 1990.
[3] Ho, Y., Partial order state transition analysis for an intrusion detection system, Master s thesis University of Idaho, 1997.
[4] Valdes A., Anderson D., Statistical methods for computer usage anomaly detection using NIDES Conference on Rough Sets and Soft Computing November, 1994.
[5] Stanton, P.T. Yurcik, W. Brumbaugh, L., FABS: file and block surveillance system for determining anomalous disk accesses, Systems, Man and Cybernetics (SMC) Information Assurance Workshop, 2005. Proceedings from the Sixth Annual IEEE, pp 207-214, June 2005.
[6] Koch, D.B.; 3D visualization to support airport security operations, Aerospace and Electronic Systems Magazine, IEEE, Volume 19, Issue 6, Page(s):23 – 28, June 2004
[7] Vert, G., Frincke, D.A., McConnell, J. A visual mathematical model for intrusion detection, 21st NISSC Conference Proceedings, Crystal City, Virginia, 1998.