Proceedings of the

2nd Cyberspace Research Workshop

June 15, 2009

Shreveport, Louisiana

Editors

Jean Gourd
Louisiana Tech University

Vir V. Phoha
Louisiana Tech University

S. S. Iyengar
Louisiana State University

The Cyberspace Research Workshop is hosted by the Center for Secure Cyberspace (CSC), a collaboration between Louisiana Tech and Louisiana State Universities. Funding for the CSC is made possible by a grant from the Louisiana Board of Regents Support Fund.
LEQSF(2007-12)-ENH-PKSFI-PRS-03.

GENERAL CHAIR

Les Guice

Conference Chairs

Stan Napper
Asok Ray

Publication & Proceedings Chair

Jean Gourd

Publicity Chair for Announcement and Web

Christian Duncan

Local Arrangement Chair

Brenda Brooks

TECHNICAL PROGRAM COMMITTEE

V. Phoha, Chair
A. Ray
C. Duncan
K. Balagani
T. Kosar
J. Zhang
D. Ali
S. S. Iyengar, Chair
J. Gourd
R. Selmic
E. Karim
G. Allen
P. Wahjudi

MESSAGE FROM THE GENERAL CHAIR

It is my privilege to welcome to Shreveport all the distinguished guests and participants for the Second Cyberspace Research Workshop (CRW). We appreciate the support of the organizers of the 2009 Air Force Cyberspace Symposium (AFCS) by including this workshop as part of the Symposium.

The AFCS is organized under the theme of “Collaboration in Cyberspace” reflecting the broad participation of speakers and attendees across the public and private sectors. Academia certainly recognizes the importance of collaboration in advancing a complex interdisciplinary field of science such as cyberspace, and it is most fitting that this research workshop be a part of the AFCS.

Cyberspace is rapidly emerging as an area of national priority as reflected by the numerous reports, policies and budgets that are shaping the direction of future investments. Research and education are essential components of these plans and it is important that academia provide effective leadership in advancing the agenda. This workshop provides the opportunity for academia to interact with government and private sector leaders, to share information on the latest research findings, and to develop plans for the future.

Our institutions take this opportunity to thank the Cyber Innovation Center (CIC) for all of its support in fostering collaboration, research, and technology development in the cyberspace industry. The CIC staff has worked tirelessly to bring together academic, government, military, and private sector leaders to provide innovative solutions to the nation's critical cyber security and defense needs.

We also thank the Louisiana Board of Regents for its funding that has brought together Louisiana’s leading cyber researchers to establish the new Louisiana Tech-LSU Center for Secure Cyberspace to promote research excellence in cyber-centric sensor systems. The Center has initiated research in new areas of interest to both the military and the civil sector.

Perhaps the most important measure of the success of a research conference is the quality of original research and the discussions and ideas that are generated. We hope the CRW provides you with a sense of intellectual fulfillment. Most importantly, we hope that you get opportunities to meet other researchers and take back to your institutions many ideas and friendships that will seed new research and collaborations.

Les Guice, General Chair

MESSAGE FROM THE CONFERENCE CHAIRS

Welcome to the Second Cyberspace Research Workshop.

The first decade of the 21st century has provided the basis for fascinating new technologies that are *merging* in cyberspace. These technologies are changing the way we live, are bringing new challenges in security and privacy of information, and are changing the way the nation, states and individuals fight and defend. Of the many new transformative trends, we think that three major technologies—cloud computing, social networks, and integration of cyber and physical systems—will usher new online landscapes. The rapid changes in cyberspace have brought new challenges: how will these technologies shape the cyberspace landscape? What social changes will they affect? What security and protection problems will they spawn? And what will the solutions to these problems be?

The purpose of this workshop is to gather, at one place, researchers, practitioners, and users of emerging technologies to discuss not only the foundational research that forms the basis of building secure systems for these transformative technologies, but also to ponder where the cyberspace landscape is heading, and whether we can (or should) steer it into a safer, more guided environment. Or perhaps the dangers of manipulating a huge dynamic force—the cyberspace domain will prove to be too much to control. Will any effort result in catastrophe? Are we better off to let nature take its course? Whatever the emerging landscape becomes, it is imperative that we respond to it and develop technologies to make it more secure. We hope that this workshop will start a "thinking" towards answering these questions.

We have organized the workshop to present fundamental research—you will see 11 peer-reviewed papers that discuss foundational methods and a work-in-progress session where preliminary and ongoing work is presented. Integration of research with practice is very important for a field such as cyberspace, and that is why it is by design that we are holding this workshop in conjunction with the Air Force Cyberspace Symposium.

This conference has required the efforts of many people: the keynote speakers, the authors of the papers, the reviewers, the program committee, and others too numerous to list here. We thank them all.

Vir V. Phoha and S. S. Iyengar, Program Chairs

TABLE OF CONTENTS

Integration of the Visual Authentication of Spatial Data with Spatial-Temporal Class Taxonomies for Advanced Spatial Authentication Modeling to Create *Pretty Good Security*

*Greg Vert, **Jean Gourd, *S.S. Iyengar
*Dept. of Computer Science, LSU, Baton Rouge, LA 70809
**Dept. of Computer Science, LaTech, Ruston, LA 70809
e-mail: gvert12@csc.lsu.edu

Abstract

Due to the criticality of spatial data in decision making processes that range from military targeting to urban planning it is vital that transmission of spatial data be authenticable and secure. Cryptographic methods can be utilized for this purpose; however, they can be relatively slow especially when encrypting voluminous quantities of data such as is found with spatial data. Previously a promising and fast method of low overhead spatially based visual authentication has been developed. This method considered the angular and temporal relationships of spatial object data. It was initially shown to be extremely fast and easily extended to an intuitive visual algebra that makes it easy for a human being to identify modifications to data such as deletion, movement or additions of spatial objects. Additionally, work was done to taxonomically classify spatial objects based on their spatial and temporal properties. This paper integrates the two approaches to i) introduce a new concept in security; that of *pretty good security* and ii) potentially dramatically increase the speed of the authentication method on top of the visual signature authentication methods developed previously. The approach integrates the notion of spatial and temporal taxonomic relationships when determining what key spatial objects should be authenticated.

1. Introduction

Spatial data sets or maps get transmitted over the Internet all the time for planning processes and decision-making support ranging from resource management to urban planning [1, 2, 3, 8, 11]. This highlights the need to create techniques to protect and secure the transmitted spatial data. Authentication is a method of determining whether a data item has been modified. It enables computers at the receiving end to verify the contents of the message [4, 5]. Authentication can range from simple functions such as using passwords to very complicated identifiers. Advanced approaches to authentication may increase the amount of authentication required, based on the perceived risk associated with accessed resources. This is referred to as risk based authentication.

In this paper, an approach to authentication focused on determining what is important to authenticate based on spatial temporal category relationships is presented. A previously developed visual method for authentication of spatial data is integrated with taxonomically-based spatial-temporal classifications to improve the authentication speed of the visual authentication method. This method can create ultra fast authentication where only relevant parts of the spatial data based on taxonomic relationship are authenticated. The result is defined as *pretty good security* and has the potential to be considerably faster than currently existing methods.

1.1 Background

Encryption is a widely utilized method to authenticate and protect data. There are many encryption techniques available and commonly used such as systematic encryption, and Public-Key encryption. When one considers the application of such methods to spatial data there are several questions that must be considered:

- Cryptographic algorithms tend to be designed to work on relatively small

amounts of data and thus can be computationally expensive.
- When considering the application to spatial data, the question often becomes which data needs to be encrypted and thus does all data need to be encrypted.

Very little work appears to have been done on the development of authentication methods based on properties describing spatial data. This has become the motivation for the development of a new method for doing spatial data authentication inspired from the concepts of biometrics. This approach utilizes taxonomically related classes of spatial information to select subsets of spatial objects and a visualized mathematical model to generate a geometric signature for the data sets that can be used for authentication and can visually point to modified objects in a spatial dataset. The approach is based on the spatial and temporal properties of objects. The result is a method that can be ultra fast and selective in what is authenticated.

2. Previous Work

In previous work, the notion of visual authentication of spatial objects has been developed [6]. Additionally, the concept of spatial taxonomic classes of objects was defined based on their temporal and spatial properties defined in other work [12, 13, 14]. This section presents an overview of this work to set the grounds for their unification into an authentication scheme based on the integrations of the ideas.

The question of how to identify spatial objects for authentication signatures is based on research similar to the classification categories of Peuquet [2]. Our research extended this previous work to classify spatial objects based on the effect time has on them. That means every object was studied with respect to time and what changes can occur to that object due to time. We define the term "degree of temporality" as being how long it takes an object to change its spatial geometry and define this concept as:

$$Degree\ of\ temporality = \frac{\Delta\ Spatial\ Geometry}{Time}$$

From this definition, we derived the following classifications for objects:

- Temporal Continuous (TC) – an object whose degree of temporality and attributes change continuously
- Temporal Sporadic (TS) – an object whose degree of temporality and attributes change in an unpredictable fashion
- Static (S) – an object that has no change in degree of temporality or attributes
- Static Temporal (ST) – an object that is typically static, but may—under certain situations—have changes in degree of temporality and attributes

The following presents a sample summary of some spatial objects and how they may fit into these classifications.

Static	***Static-Temporal***	***Temporal-Continuous***	***Temporal-Sporadic***
Ocean	Sea	Ice Mass	Port
Island		Desert water	Land
Rocks		Shore	Farm
Forest		Sand	Park
Summit		Silt	University
Mountain		Clay	Parcel data
Hill		Bushes	Corral
Valley		Lake	Dam/Weir
		River	Mines

Table 1. Temporality classification of the taxonomy

3. Using a Visual Glyph to Authenticate Spatial Data

Visual authentication can be a fast method to create an authentication signature compared to encryption because the algorithmic complexities of trigonometric mathematics are simpler than encryption algorithms. The

following section therefore presents the development of a visual authentication method that can be applied to spatial data.

3.1 Spicule Visualization Tool

Takeyama and Couclelis have shown that GIS layering abstraction of a location is equivalent to a set of multiple attributes [9]. So, the map can be looked at as a 3D set of layers on top of each other. In this 3D paradigm of layered spatial data, the spicule can be utilized to create a mathematical signature for authenticating spatial data by mapping the tips of vectors on the spicule to the unique spatial objects identified from the taxonomy. The signature that can be generated using this approach becomes an *n*-tuple which can be visually subtracted using the spicule to detect changes in the spatial data.

The spicule was developed [6, 7] as a tool for detecting intrusions of malicious software and individuals on computer systems. It is a visually based glyph that has a strong mathematical foundation based in linear algebra. As a research tool, it is still being investigated for application to a wide area of pattern recognition problems. One of the interesting properties of the spicule is that it has a simple but powerful algebra that allows rapid visualization and detection of changes in data sets that the spicule has been mapped onto [6, 7]. Thus the mathematics of the spicule can be utilized to authenticate and detect changes in the relative geometries of spatial objects

To illustrate the spicule concept consider Fig. 1 and Fig 3. In these figures one observes a ball floating in 3D space. This ball has—attached to its circumference—vectors that can be mapped onto objects in 3D space such as spatial objects found in spatial data. As an example, the tip of H vector may be mapped onto the specific intersection of a street system that has been classified as a static temporal object. The spicule has a variety of different vector types, each with its own unique properties. In the figure, a new type of vector is defined. This vector is based at the equator of the spicule's main ball and has the head of the vector mapped to spatial objects in a data set. When considering this scheme it is fairly straight forward to see that such a mapping can create a series of vectors each with a unique mathematical description of orientation, angle and direction. This descriptive information constitutes the authentication signature for the transmitted spatial data.

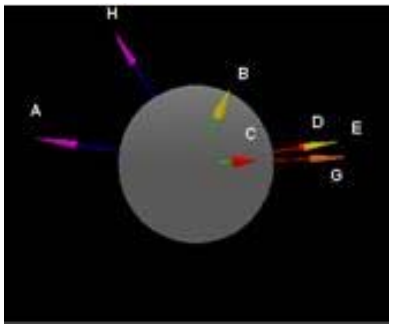


Fig. 1. Sample Spicule glyph

In Fig. 2, an example of how a spicule might map onto significant spatial features found in various GIS data layers is shown. In this illustration, layer one may represent a street system, layer two may represent the distribution of forested areas, and layer three may represent houses. Each vector on the spicule has a unique set of descriptive attributes such as vector magnitude, vector angular orientation, and location of a vector on the 3D central ball. A vector's descriptive orientation can then become a signature of the spatial object it has been mapped onto [10].

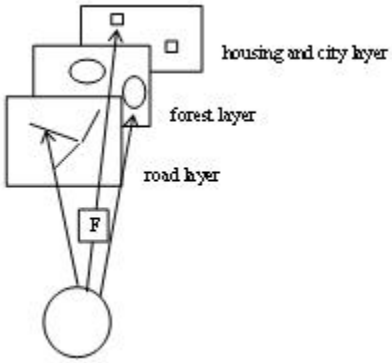


Fig. 2. Sample mapping of spicule glyph to features in GIS spatial data

The current form of descriptive signature equation is given by:

$$V = V_1 + V_2 + \ldots . V_x$$

Where V_x is given as:

$V_x = (|V_x|, \text{elevation, equatorial location})$

Where:

$\|V_x\|$	magnitude
Elevation	angular degrees above horizontal
Equatorial location	angular degree of vector
	tail when mapped on the
	spicule ball

This signature creates a unique description of the orientation of a spatial object within the 3D data space.

In this scheme a vector pointing from the center of the spicule, at the origin, to each point or spatial object selected from the taxonomic spatial temporal plot is used to create a signature. The *n*-level data layers shown in Fig. 2 are initially proposed to be placed at one vertical unit apart from the spicule layer. So, the first layer points will have coordinates of (*x*, *y*, 1), the second layer point coordinates will be (*x*, *y*, 2), and the third layer point coordinates will be (*x*, *y*, 3). Based on this, the vector attributes for each authentication point in the three layers will be:

$$Mag_i = \sqrt{x^2 + y^2 + i^2} \quad (1)$$

Where:

i is the data layer number

x, y are point original coordinates

Mag_i is the magnitude of the vector from (0,0,0) to a point in layer *i*.

$$Sin\theta_{ei} = \frac{x}{\sqrt{x^2 + y^2}} \Rightarrow \theta_{ei} = Sin^{-1}\frac{x}{\sqrt{x^2 + y^2}} \quad (2)$$

$$Sin\theta_{vi} = \frac{i}{\sqrt{i^2 + y^2}} \Rightarrow \theta_{vi} = Sin^{-1}\frac{i}{\sqrt{i^2 + y^2}} \quad (3)$$

Equations 2 and 3 are used to calculate the equator and the vertical angles respectively,

Where:

i is the data layer number

θ_{vi} is the vertical angle degrees for a vector from (0,0,0) to a point in layer *i*.

θ_{ei} is the equator angle degrees for a vector from (0,0,0) to a point in layer *i*.

The collection of attributes and angles for all authentication vectors forms a two-dimensional matrix that is used for the authentication signature and the spicule visualization authentication process (Fig. 5).

The signature calculation process is done when a spatial dataset is requested to be transmitted over the internet. Table 2 shows a sample calculated vector matrix.

Object ID	Layer	Mag_i	θ_{vi}	θ_{ei}
1	3	7.68	66.8	18.43
2	2	16.31	42.51	4.76
		. .	. .	. .
N	i	29.22	51.95	3.18

Table 2. Sample calculated vector matrix

At the receiving end, the same process to create a signature matrix from the received spatial dataset was applied. By visualizing the mathematical *difference* between the received spatial data set matrix and the transmitted matrix, as shown in Fig. 3, it can be determined if the dataset has been intercepted or altered during transmission. If no modifications have been made, the result is a featureless, smooth ball. In Fig. 3 the resulting vector points to an object that has been modified.

4. Comparative Authentication Signature Generation Performance

Spatial data may be protected for transmission by encryption or by the generation of a signature using MD5, SHA or RIPEMD. In order to compare the performance of the spatial signature approach to that of above traditional methods a test suite was set up on a PC running at 2.4 Ghz with a P4 processor. The Crypto++ package was utilized for comparison with timing figures measured down to the millisecond.

Crypto++ has a program called Cryptest that may be called with a command line switch to encrypt symmetrically, and decrypt and generate SHA, MD5 and RIPEMD160 digests. The comparative speeds from this initial performance testing are shown in Table 3.

Test Type	Pass 1 (10x)	Pass 2 (10x)	Pass 3 (10x)
Shell	63.00	58.00	57.00
Encrypt (symmetric)	126.60	123.4	121.90
Decrypt (symmetric)	115.60	123.5	121.90
MD5/SHA/RIPEMD	67.20	67.20	64.00
Spatial Authentication	< .01 millisecond	< .01 millisecond	< .01 millisecond

Table 3. Average performance comparison of Spatial Authentication versus Symmetric encryption, SHA, MD5, RIPED (milliseconds) on test data

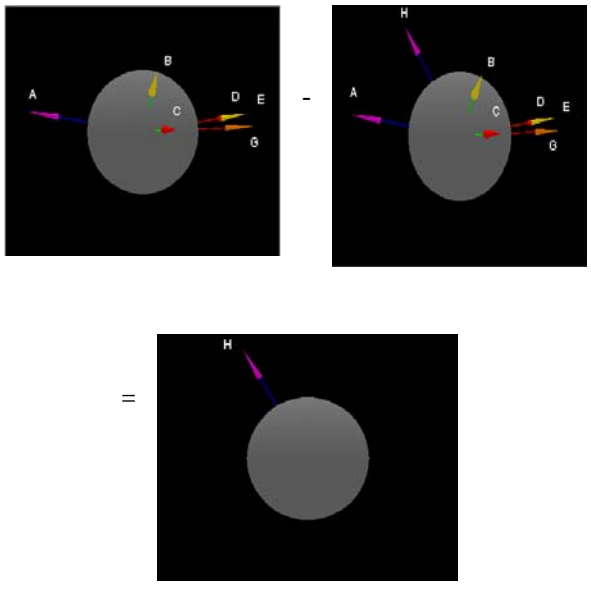


Figure 3. Visualization of authentication signature: Template form (left) - Authentication form (middle) = Change form (bottom); indication H has been added to the transmitted data set generating the Authentication form

5. Selective Authentication

Not all spatial or temporal objects need to be authenticated. This is due to the fact that some objects may not have a strong relationship to objects in another taxonomic class. For instance, static objects may not be related to objects that are continuously changing. This observation leads to the notion that i) authentication can be done based on taxonomic classes of objects that are of interest to a user and ii) partial authentication can reduce the already fast authentication speeds from the visual approach. This idea is referred to as *pretty good security*. The selection of what is authenticated becomes a function of the *relationship among objects* and thus can be defined by the user and is a subject for further research.

Such an approach is suggested via the implementation of a similarity matrix that has the following format:

	S	ST	TC	TS
S	1	.75	.5	.25
ST	.75	1	.75	.5
TC	.50	.75	1	.75
TS	.25	.50	.75	1

Table 4. Similarity matrix of taxonomic classes utilized for object authentication

where the taxonomic classes are abbreviated as defined previously. In the above matrix, a hierarchy of relationships among classes defined in the spatial-temporal taxonomy is defined. A value of one indicates absolute relationship and lower values indicate less relationship. This hierarchy can then be utilized to determine by a user what objects (from Table 1) in a spatial data set should be authenticated using visual signatures. For example, if a user wants to only visually authenticate objects that are Static (S), such objects {ocean, islands, rocks, etc} would be used in the visual authentication process. Whereas if a user wanted to authenticate all objects that are greater than or equal to .75 (a high degree or relation) then the objects in the ST and S classes would be authenticated. The result is that a user can select to authenticate only highly related objects and thus increase the already fast authentication times shown previously for the visual authentication method. In this similarity matrix the class relations are split evenly due to there only being four classes

at the present in the spatial-temporal taxonomy. However, this approach in the future should be studied to determine better adaptive schemes for given security situations that might be applied to the similarity matrix.

6. Conclusions and Future Work

The integration of the visual authentication method with taxonomic classes can provide for dramatic increases in the speed of authentication based on the new notion of authentication of related spatial-temporal objects. This concept is referred to as *pretty good security*. This work builds on previously defined research topics in the area of spatial authentication. The level of confidence in the authentication is left up to the user and should be the subject of future empirical work. Other work could involve the study of dynamically changing relationship values, how they might be defined dynamically and the effect on speed. Additionally, this method has terrific potential in the newly developing paradigm of a global information system based on spatial temporal relationships among data objects and global contextual processing.

References

[1] ESRI Data & Maps, Media Kit. 2002. Esri ArcGIS. www.esri.com

[2] "An Introduction to Geographical Information Systems", by Ian Heywood, Sarah Cornelius, and Steve Carver. Second Edition, 2002. Prentice Hall.

[3] Environmental Modeling Systems, Inc. WMS 7.1 Overview. http://www.ems-i.com/WMS/WMS_Overview/wms_overview.html

[4] William Stallings. 2003. Network Security Essentials, Applications and Standards. Prentice Hall.

[5] Charlie Kaufman, Radia Perlman, Mike Speciner. 2002. Network Security, Private Communication in a Public World. Prentice Hall PTR.

[6] Vert, G. Yuan, B. Cole, N. A Visual Algebra for Detecting Port Attacks on Computer Systems, Proceedings of the Intl. Conf. on Computer Applications in Industry and Engineering (CAINE-2003), November 2003, Las Vegas, NV, pp 131-135.

[7] Alexandria Digital Library Feature Type Thesaurus. University of California, Santa Barbara. Version of July 3, 2002. http://www.alexandria.ucsb.edu/gazetteer/FeatureTypes/ver070302/index.htm

[8] Introduction to ArcView 3.x. ESRI Virtual Campus, GIS Education and Training on the Web. http://campus.esri.com/

[9] Takeyama, M., and Couclelis, H., 1997, Map dynamics: integrating cellular automata and GIS through Geo-Algebra. *International Journal of geographical Information Science* 11: 73-91.

[10] Jensen, C.S., and R. Snodgrass. 1994. Temporal Specialization and Generalization. *IEEE Transactions on Knowledge and Data Engineering* 6(6): 954-974.

[11] Onsrud, H.J., and G. Rushton. 1995. Sharing Geographic Information. Center For Urban Policy Research, New Brunswick, N.J. 510pp.

[12] Guimaraes, G., V.S. Lobo, and F. Moura-Pires. 2003.A Taxonomy of Self-Organizing Maps for Temporal Sequence Processing. *Intelligent data Analysis* 4:269-290.

[13] Heaton, Jill. Class lecture. University of Nevada, Reno. 08/23/2004.

[14] Calkins, H. W.; Obermeyer, N. J.; Taxonomy for Surveying the Use and Value of Geographical Information. *International Journal of Geographic Information Systems* V. 5, N. 3, July-September 1991, pp. 341-351.

Maturing Cyber Security Using BioThreat Experiences and Resources

Norman Lee Johnson
Referentia Systems Inc
njohnson@referentia.com

Tim Williams
Referentia Systems Inc.
twilliams@referentia.com

Abstract

How does the current planning and response to cyber threats compare to biological threats planning and response? How do the resources of each compare? Biothreats have been a concern for millennia, and humans systems have had significant time and funding to develop a mature response. In this paper we observe that by comparison, cyber response is still in a relatively immature stage, possibly comparable to the state of public health protection prior to the implementation of safe water, sanitary conditions and vaccinations. Furthermore, we argue that because of the similarity between bio- and cyber systems, there are significant opportunities to advance the maturity of cyber research and response, either by using bio analogies for inspiration or by the direct transfer of resources. An analysis of existing cyber resources and gaps are compared to available bio resources. Specific examples are provided for the application of bio-resources to cyber systems.

1. Introduction

Cyber attacks are the most asymmetric of threats facing our nation today. A few individuals acting remotely can damage or destroy the operational capabilities of an entire government, military, and/or commercial sector – with minimal resources and preparation, with almost no risk during the attack, and with low likelihood of attribution. Our cyber vulnerability is partially persistent because of our limited success in managing our growing infrastructure complexity, in addition to the challenge of addressing known, resolvable cyber-security issues. Daily cyber attacks against commercial and government infrastructures are on the rise, and a report from 160 CEOs [i] suggest we are at risk of a "Cyber Katrina" unless action is taken. There are no shortages of studies identifying the problem and recommendations to solve it.[ii] Policy statements, national position papers, and strategic federal agency plans have repeatedly identified strategic and operational cyber vulnerabilities, provided recommendations, and defined courses of actions over the last 5 years. The strongest recommendations are that:

- The greatest current challenge is our inability to address known vulnerabilities.
- Our information infrastructures, originally developed as security-neutral, must be transitioned to secure technologies, for example, making information assurance and identity management part of the infrastructure.
- The long-term management of the cyber challenge requires a system-wide engagement and commitment of all stakeholders, likely with a greater role for federal agencies.

The first two recommendations above are being addressed at some level by the nation and the armed services in the development of new cyber-security resources including detection, monitoring, analysis tools, training programs, and testbeds. But the final recommendation appears difficult to motivate and is illustrated by the observation that there is currently no capability to rank consequences against

mitigation costs, particularly for high-impact but rare events such as a "Cyber Katrina." Another indicator of the lack of addressing the last recommendation is for preparedness planning: only one of the above-cited, high-level planning reports[iii] call for predictive analysis technologies with risk assessment and consequence management to address the need for planning and response. Yet, predictive analysis technologies are central tools to other threat areas (chemical, biological, nuclear, radiological, etc.). This suggests that a major difference in maturity of planning and response systems exist between cyber and other threat areas.

The remainder of this paper examines the similarities between public and cyber health systems, how relatively mature the two domains are, and finally how activities in the bio-threat domain may help mature the cyber domain. For completeness we note that there are two application areas in the cyber domain which were inspired by the bio domain: computer security based on the adaptive immune systems [iv] and simulations of the spread of computer viruses (or other replicating threats) based on epidemiology.[v] As will become obvious, these two areas of study, while important contributions to the cyber domain, represent a small part of the full opportunity.

2. The Difference in Maturation of Public and Cyber Health

A review of how public health resources has matured over time for biological threats is a helpful perspective for cyber preparedness. Figure 1 shows how until fairly recent times (150 years ago), public health experienced unstoppable and unexpected waves of epidemics, not too unlike our current experience with cyber threats. Removing these frequent epidemics from our society required that we develop healthy practices and infrastructures (safe water/food, sanitation) and specifically address certain known and reoccurring threats (smallpox, dysentery, bubonic plague, etc.) with vaccination or therapeutics. Once these preventative measures were operational, the public-health systems could focus on the relatively infrequent outbreaks of more difficult or unknown threats.

As we shift our cyber-health system by the implementation known countermeasures for common cyber threats, we will enter a similar phase of reduced "cyber epidemics."

In the above comparison of the development of biological and cyber health systems, the broad similarities are apparent. But, some might counter that there is a fundamental difference: biological systems have had the same host "technology" for millennia (our bodies), where technologies in cyber systems (host, networks, etc.) are constantly changing. This suggests that we could forever live in an epidemic-ridden cyber world, and never achieve the stable, mostly disease-free second stage found in public health.

A counterargument is that the bio-world is

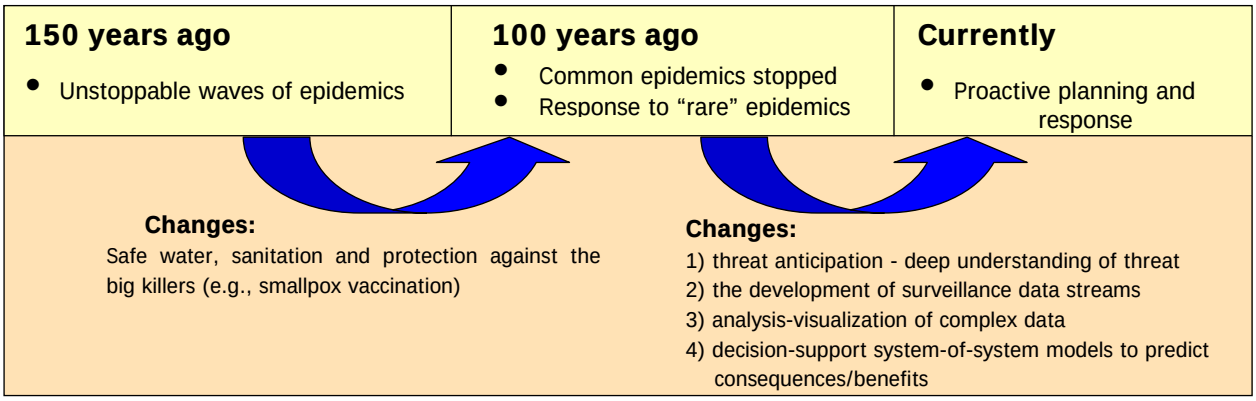


Figure 1. How public health has changed over the last 150 years in the Western World.

equally adept at developing new “technologies” which exploit vulnerabilities, and our bodies have developed sophisticated multi-layered immune systems that have sustained the balance towards health. Furthermore, while the body “technologies” are unchanged, the interface between our bodies and our public health systems is complex and constantly changing as new health technologies are developed. In this argument, we are optimistic that comparable cyber-immune systems will be developed, and that a similar relatively “disease-free” cyber-health stage can occur.

Another lens on the relatively maturity of bio- and cyber-response programs is to examine defender activities that occur before and after an attack. One extreme is a purely responsive posture where you are primarily focused on containment of the attack and consequence management. The other extreme is where threat planning and response programs become more mature, as in the bio-threat space. Here, program activities address issues and opportunities well after an event (because they don’t have to hunker down for the next attack) and well before (because of better preparation and understanding of the nature of the attacks and perpetrators). Post-event bio activities and programs include – listed from the event to much after – situational awareness, containment, consequence management, mitigation, forensics, remediation, and recovery. Pre-event bio programs include – listed from the event to much before – interdiction (stopping an attack closer to the source), anticipation, monitoring and detection, intelligence gathering on groups and possible resources, custom activities to limit entry of threats, export controls to limit technology leaks, and treaties and safeguards for nations to collaborate on reducing threats. All of these are on top of a public-health infrastructure which are coupled to these activities and minimize vulnerabilities.

Interestingly, cyber programs do have a few examples of these pre- and post-event activities (e.g., export controls on encryption, surveillance/monitoring resources, etc.), but generally resources are deployed by companies rather than federal or international agencies, unlike for biothreats where federal and international programs are the main source of funding and regulation.

From the broad perspective above, cyber programs are far less mature than the bio programs. It is therefore no surprise that recent policy positions of greater federal involvement and international cooperation on cyber threats are important to maturing our cyber defense. But how was this final maturation of bio-threat response accomplished at a more technical level? The relevance of the final transition of public health in Figure 1, occurring in the last 10 years or so, to cyber health is the focus of this paper: public health is undergoing a transition from a responsive posture (create a system to deal with the unknown threats as possible) to a proactive, defensive planning and response posture. The viewpoint of this paper is that the research and development activities for maturing cyber systems can be greatly advanced by a comparison to bio-planning and response programs. Instead of reinventing the wheel (and the car and the supporting infrastructure) for cyber security, can we leverage the knowledge and resources from existing, effective bio-threat programs?

3. Process/Functional Similarities

There are many levels of similarity between the processes of cyber threats and bio-threats which enable the use of bio-threat solutions and tools as templates – if not actual resources – for cyber research and tools. The greatest similarity, and the one that drives our choice of vocabulary for cyber threats, is the infectious processes. This process can be made more general, again using the bio-threat understanding, by

identifying a threat-host process (of which the infection process is a subset), because some threats do not involve infection, such as allergies or denial-of-service attacks. Familiar bio-cyber examples for the threat-host process are:

- The viral spread by a compressed code that highjacks host processes,
- The signatures in the "genetic" code that can be used for identification,
- Signatures of the threat from its activity within the host or between hosts (in cyber systems these are, for example, access logs or non-essential files; in bio-systems these are non-essential bio-compounds), and
- The self-destructive immune response in the host to the presence of a threat.

On the host-response side, there are strong similarities at the function and process levels:

- The host immune state – as determined by immunization or prior or current infections – determines susceptibility,
- The host defensive options are similar in form, function and process – firewall-cell wall with preferential transport, layered defense systems, innate (always active) and adaptive (takes time to be active) immune response, system isolation and, if necessary, death of the host.

4. System-Wide Consequence Similarities

There are also similarities of the consequences due to changes in host activity on system-wide functions from a system-of-systems viewpoint. These can be broken down into direct and indirect or secondary consequences.

Direct system-wide consequences reflect the impact of the reduced activity or removal of the host on the system: the host both performs activities useful to the greater system (as a DNS server or a soldier), as well as being a repository of information for the rest of the system. Direct consequences can have short, medium and long-term impacts on the greater system depending on their function and how long they are degraded or removed from service. This bio-cyber similarity may enable some cost-benefit analysis resources that are used in bio-systems to be applicable to cyber systems. This statement needs to be qualified somewhat because of the observation that human and cyber coupling has quantitative differences in coupling with the greater system: humans require extensive coupling with other systems (transportation, different places to work and live, etc.) in comparison to cyber systems (e.g., cyber hosts don't work and live in different environments). But even this observation is rapidly changing as greater interdependence is becoming core to host functions in cyber hosts, such as the trend toward cloud computing/storage.

The indirect or secondary consequences – those system consequences that result indirectly from changes in the host activity or function, often due to interdependence of infrastructures – have greater similarity and, consequently, greater opportunities. A simple example is our power-generation and distribution systems: both rely on human and cyber support for continued operation. As human and cyber systems are compromised, the power grid becomes at greater risk of intermittency and possible collapse. Similar statements can be made for other infrastructures: banking, finance, water, food, transportation, etc. and many studies are being developed about the interdependencies of different infrastructures.[vi] It is telling to note that critical infrastructure studies are only recently including cyber systems.[vii]

5. Maturing the Domain: General Considerations

A detailed review of the mature programs and resources in responding to bio-threats (both those that naturally emerge, as in pandemic influenza, and those that are intentionally created, as in weaponized anthrax spores) is

beyond the scope of this paper and is available elsewhere.[viii]

In the previous discussion of the relative maturity of bio- and cyber-security programs, we observed that mature programs address the threat from end-to-end: from the control of technologies that can be used to develop threats, to the discovery and monitoring of potential attacking groups to addressing the long-term consequences of an event. Here we consider the relative maturity in more detail.

A specific example of mature bio-programs is the current planning, preparation and surveillance for pandemic influenza. The world has developed an extensive and cooperative sampling and surveillance system to monitor and warn of the expansion of "bird flu" and the occurrence of a contagious human form. Additionally, predictive planning and response tools were developed and used to assess different mitigation options and to deploy systems for the response to the pandemic. One tool[ix] was developed and applied which simulated an epidemic in the entire U.S. population – 300 million people, the largest agent-based simulations used in production at the time – driven by census, workflow data and transportation data. Major changes in the mitigation strategies resulted from the application of this simulation tool in support of the White House's "National Strategy for Pandemic Influenza: Implementation Plan", May 2006.[x] No equivalent predictive planning resources or response plans exist or are being developed in the cyber realm.

Figure 1 identifies the resource components that brought about the most recent maturity of the bio-threat programs in order to transition from a responsive to proactive posture:

1) *Threat anticipation* – a deep understanding of the threat – its origins, forms, signatures, and, most importantly, potential variations;
2) *Surveillance of data streams* – providing indicators of the early stages of a possible attack and situational awareness of an ongoing attack;
3) *Analysis-visualization resources* of complex time-varying, heterogeneous data that result from 1 and 2 above; and
4) *Decision-support system-of-system models* to predict consequences/benefits/costs for planning and for forecasting the evolution of the current attack and assessing different mitigation options.

An analogy to a more simple threat system clarifies these resource components. Severe weather prediction, preparation and response have also undergone major advancements due to the development of the four components above, in particular, the development of data streams worldwide, simulation and analysis tools that are driven by these data streams, and decision-support tools.

An important observation is that the inherent, chaotic nature of weather systems requires a heavily data-driven approach – theory plus limited data is not sufficient. The same data-driven requirement is also true for bio-threats, both because of the inherent randomness of the system (such as the influence of random human-human contacts in the early stages on an epidemic), and because the attacker-protector dynamics (such as rapid change of virus from immune system pressure). Both of these sources of chaotic change can be observed in surveillance data, but are difficult to predict from theory (at best we can bound the degree of change – useful for planning but of limited utility in responding to a threat). In the absence of "theory" or detailed knowledge of the threat, then the data-driven approach becomes even more important.

Because of the similarities of weather-bio-cyber systems, we also expect cyber-security planning and response systems to equally require a data-driven approach. This approach includes using data streams for characterizing

the range of threats and responses for planning, for surveillance of new threats, and for tracking the real-time system response to an evolving threat and attempted mitigations.

6. Maturing the Domain: Specific Guidelines from the Bio-Experience

Table 1 summarizes the resource components listed in the last section for the identification of resources and gaps to mature the cyber domain and then identifies the potential enabling bio-resources.

Because of the important role and opportunities of the different aspects of decision-support tools, three essential steps are identified in the maturation of decision-support tools for cyber programs:

1. The development of forecasting resources (typically in the form of simulations) – where we use the word forecast over prediction to indicate the chaotic nature of the systems and the need for a stochastic treatment,
2. The development of cost-benefit analysis resources (typically risk assessment and management tools) and
3. The development of integrated decision-support tools that combine all of the previous developments (data generation to analysis to prediction to cost-benefit).

Table 1 is far from being exhaustive and represents the authors' experiences (possibly myopic) into the bio- and cyber domains. Yet, even with this qualification, the gaps in a mature cyber-security programs are evident and intuitive. And, with some familiarity with the bio-threat resources, the possible opportunities for inspiration from the bio-domain, if not actual resources, are apparent. The next section provides definitions of bio-vocabulary or research areas that may be unfamiliar.

7. Useful Definitions in Bio-Threats

Threat Phylogeny: using the genetic code in the "genome" to determine the relationship between threats and their variations – often indicating their evolutionary linage and separation.

Virulence databases: a database (and understanding) of the genomic components that make a threat dangerous. For cyber it might be a "delete-all" call. Note that even though a genome or code may contain virulent factors, they may not be expressed.

Forensic tools: powerful analysis resources which connect the presence of a threat or characteristic to its source or history – perhaps the most developed application area for bio-threats outside of public health.

Syndromic surveillance: examines the statistics of symptoms appearing over time and location to identify health problems before physicians can diagnose them – these are becoming common in local public health departments and the military. Some bio-attacks can only be identified by this method.

Virulence change identification (ID): Identification of how a threat changes over time. We currently are tracking this for bird flu, to identify the remaining changes needed to observe a human epidemic.

Health metrics: measures of health of the public, etc.

Standardized threat scenarios: a set of scenarios (threats and deployments) that are broadly accepted by the community.

Threat anticipation: This is a very complex area. It can range from intelligence that tells you what your enemy is planning, to an analysis of your vulnerabilities and the resources available to identify where likely attacks could take place. The science-based form is essential for predicting the unexpected or unknown.

Table 1 - Illustration of how mature bio-threat resources can or may help fill gaps in cybersecurity

Cyber Resources Required for Mature Planning & Response	Existing Cyber Resources	Cyber Gaps: Needed Resources	Enabling Bio-Resources
Diverse cyber data: providing historical and real-time data of current network topology and traffic; enclave, component and user activity, access, status	**Rich and more in development -** Network flow traffic types/volume; component types & programs used	Status of components: susceptibility, symptoms of attack, readiness, activity, threat level	Genome" threat data bases, "virulence" databases, current threats, current news
Analysis and visualization of complex data streams: past and situational health, attacks, losses; global-to-local drill down, weak-signal precursors, threat ID and attribution, intuitive analysis of large data sets	**In development -** Large data set analysis identifying trends and precursors, anomalous behavior, ideally automated	Health of network and components, direct and inferred attack status, syndromic precursors to attack ID, forensics, threat attribution, …	Threat phylogeny, syndromic surveillance, health metrics, virulence change ID, forensic tools, responsiveness status, visualization resources
Predictive models of future state/losses from an attack given historical and current state, with transparency of outcome-to-cause and uncertainty quantification	**Scarce** - mostly academic simulations of network activity for limited threats; no exhaustive studies of tipping points	Databases of threats, standard threat models, emerging threat theory, effectiveness of response options	Epidemiological simulation resources, studies of mitigation options, coupled infrastructure sims, cost estimates,
Consequence - benefit resources including risk assessment, management and communication, expert-stakeholder conflict resolution, mission continuity	**Very limited for real-time response**; **limited for planning;** fundamental understanding limited	Metrics for mission readiness, threat-vulnerability mapping, integration of simulations	Standard threat scenarios for uniform preparedness, advanced risk assessment, adversary models,
Decision-support integration of above for planning and response: quantitative and transparent assessment of options, local-to-global cost-readiness tradeoffs, acquisition guidance, etc.	**Very limited** - currently wet-ware (human) based, no policy-level guidance on infrastructure acquisition, no operations support tools	Cost-benefit analysis of "what if" scenarios and response options; Risk management and communication	Threat anticipation-prediction, risk-based training, multi-stakeholder net-assessment studies, acquisition tools

8. Examples of Mapping the Bio to Cyber

Many of the "enabling bio-resources" in the right column of Table 1 require a lengthy discussion to explain and to exploit the perceived opportunities. The following highlights a few of the more easily communicated opportunities. The next section provides one detailed example of mapping a specific resource. A web search on key phases will lead the reader to more information. Please contact the authors for questions or assistance.

The opportunities that are most apparent for the cyber domain from the authors' perspective are (in the order of top-to-bottom in Table 2):

- Development of cyber-threat databases that are based on the code content, independent of the expression/use of the code, to allow quick assessment of the threat potential. A subset of this database is a virulence

database that contains coding that makes a threat "virulent" or destructive to the host or system.

- Threat phylogeny examines the evolution of threats based on their coding, to understand their origin and possible activity in certain hosts. Engineered threats make these evolutionary studies less useful, because large changes typically occur in engineered threats (even in bio-systems) in comparison to evolutionary changes. Syndromic surveillance examines the symptoms of host systems to detect a threat based on its effects rather than its direct presence. This type of surveillance is particularly useful in detecting unknown threats where the "genetic" coding is not known. Cyber surveillance has crude forms of this approach, such as observing unexplained increases in computation burden or number of files.
- Significant resources for epidemiological simulations over many scales (spatial and functional) are available in bio systems.[xi] Some of these resources may be useful for cyber-system modeling.
- Standardized threat scenarios are useful in a maturing program to focus researchers, government and industry in developing countermeasures. For bio-systems collections of scenarios that spanned the range of threat types and consequences were particularly useful maturing awareness and focusing the discussion in a complex environment. A caution is necessary from the bio experience: standardized scenarios are good in early planning but their extended use can cause inability to adapt to new threats or developing a broader threat scope.

Once the threats are well characterized and their activity in the host is well understood, programs of threat anticipation can be developed that match the threat space to current vulnerabilities to anticipate where the next threat may occur and of what type. This level of understanding can be statistical if rich data is known on threat occurrences, can be based on intelligence information of groups in the process of developing threats, and/or can be based on a deep understanding of what threats are possible for given host systems. Threat anticipation represents a current research area for bio-threats and is quickly maturing.

9. A Specific Example of Bio-Cyber Mapping: Categorizing Threats

One of the core challenges in responding to a complex threat space (true for bio- and cyber domains) is to find some categorization of the threat space that helps in the planning of response options. We know that not all threats are equal in severity, sub-systems attacked, systems affected, etc., yet the complexity of the threat-host response can prevent "getting out of the weeds" and results in treating them all equally, at worse, or crude categorization into severe threats that must be addressed and others which can be deferred, at best. For bio-systems, the threat space is very complex and for a long time the complexity limited the planning possible. As suggested in Figure 1, experiences in threat and public health did finally lead to developing healthy living conditions and addressing the severe, reoccurring threats as possible (some threats, such as influenza, defy a general solution even though each year it kills many 10s of thousands of people in the U.S. and is costly from its impact on the workforce). A common view within the bio world is that public health programs removed the most dangerous threats as was possible for reoccurring and emerging (and possibly engineered) threats by the mechanisms listed in Figure 1. Even though this first revolution in public health reduced the expected epidemics, there remains great complexity in the threat space, and this limits the

ability to develop additionally required operational responses.

One approach to simplify the threat space was proposed in a recent National Academy of Sciences report on chemical and biological threats: to divide the threat space by the responder's ability to quickly detect the threat and the ability to quickly treat the threat, as illustrated in a cyber version in Figure 2.

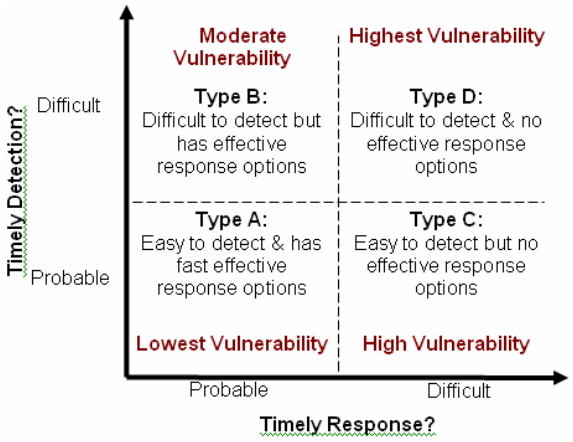


Figure 2 - An approach to the simplification of the cyber-threat space, as inspired by the approach for the bio-threat space in a National Academy study on building protection

Figure 2 is a powerful threat characterization because it:

- Puts the complex variety of threats in a comparable and understandable basis (for example it can apply to both chemical and biological threats),
- Links measurable attributes (timely detection and response) to outcome: vulnerability and consequences, and
- Points to where the biggest challenges occur: difficult detection and slow response.

While these conclusions may seem obvious, they can be difficult to communicate to the less knowledgeable. The categorization can be useful in justifying a course of actions when budgets are limited.

The next stage in the application of the threat characterization landscape is to propagate the figure with known threats and their variations. This would help in identifying an existing threat that could be modified either to be more difficult to detect or more difficult to respond to, thereby increasing its consequences.

Another application of the threat characterization landscape would be to extend its characterization by adding a third dimension to include consequences of response options (high/low), because threats that have similar timely detection and response options could differ greatly by the consequences of the mitigation, e.g., continued normal operations or suspend all operations. This axis could include "levels of regret" as used in the bio-domain, to describe unavoidable consequences from a mitigation action even in the absence of the threat, as for example, establishing a preventative quarantine or taking a host offline.

10. Conclusions

The main objective of this paper is to present the cyber security researcher a broader perspective of their activities, as seen from the lens of the complex, but more mature field, of biothreat research and programs. The full breadth of such an inquiry is not possible in this short paper, but even at a summary level many possible opportunities for new areas of research become apparent. And, just as importantly, the comparison of the two domains provides the beginnings of a roadmap for how to mature cyber security, both for research and policy. Within this context and in developing this paper, the authors were reminded how the cyber community as a whole may be excessively focused on short-term concerns and miss the opportunities at the horizon, which may lead to long-term resolutions of current challenges.

Likely each reader of this paper will see different opportunities from the bio-threat arena, but as a summary the following were significant to the authors:

- The current emphasis of policy is aligned with the immediate needs in cyber security, e.g., addressing known vulnerabilities – and rightly so, but there is a noticeable absence of planning what comes next, once a stable, lower incident environment is achieved. Now is the time to begin investing in the next stages of threat characterization, to discover what the bounds of threats are, threat anticipation, and to identify future threats and their mitigations.
- Similarly there is a push towards more federal and international engagement in cyber security, as occurred in the bio-threat domain development. Many aspects of the bio-threat programs and specific technologies can be borrowed from the bio-threat areas, as for example, global surveillance of the outbreaks of threats or the monitoring of the "syndromic" signatures that suggest the presence of a unidentified threat.
- Many technologies or approaches can be transferred directly to the cyber domain, for example, the development of threat virulence databases, simulations for planning and response, forensic resources, and particularly decision support tools for the evaluation and selection of different response and mitigation strategies.

As a final remark, there are research areas where progress can greatly benefit both cyber and public health. The prime example is the importance of human factors (cultural, social, behavioral) on the formation, spread and response of bio- and cyber threats. For example, in biothreats the greatest source of uncertainty during an outbreak is how individuals will respond. Will they panic, possibly making the problem worse or will they follow directives from authorities? Little progress has been made to reduce these uncertainties (as illustrated that behaviors in simulations are prescribed rather than adapted to the current situation[9]), making planning for outbreaks challenging. Similar arguments can be made for cyber systems. How do users respond to a real or threatened attack? Do they make the problem worse if they panic? How can they sustain their productivity in the presence of mitigation responses to an attack? At best, currently studies can be done to bound the effects of behavior, but true forecasting of cyber or bio events for either planning or response requires knowledge of how the attackers, defenders and users behave.

---

[i] "Essential Steps to Strengthen America's Cyber Terrorism Preparedness: New Priorities and Commitments" Business Roundtable's Security Task Force, June 2006.

[ii] ***Presidential directives*** (NSPD-38, NSPD-54, HSPD-12 and HSPD-23) http://www.fas.org/irp/offdocs/nspd/index.html, C***ongressional studies*** (e.g., CRS's Economic Impact of Cyber Attacks April 2004 (http://wikileaks.org/leak/crs/RL32331.pdf), IP3's National Cyber Security Research and Development Jan 2009 (http://www.thei3p.org/docs/publications/i3pnationalcybersecurity.pdf), Information Security GAO Jun04, Cyber Analysis and Warning July 2008, (http://www.gao.gov/products/GAO-08-588) Infrastructure Protection GAO-08-825 Sept 2008, (http://www.gao.gov/products/GAO-08-825) Critical Infrastructure Protection GAO-08-1157T Sept 2008, (http://www.gao.gov/products/GAO-08-1157T) ***Interagency councils*** (e.g., National Cyber Study Group, Joint Interagency Cyber Joint Task Force), ***Intelligence studies*** (e.g., National Intelligence Council's Global Trends 2025 (http://www.dni.gov/nic/NIC_2025_project.html)), ***Public policy institutions*** (e.g., CSIS's Securing Cyberspace for the 44th Presidency Dec 2008, (http://www.csis.org/media/csis/pubs/081208_securingcyberspace_44.pdf)), ***Federal agency studies*** (e.g., DHS's National Strategy to Secure Cyberspace (http://www.dhs.gov/xprevprot/programs/editorial_0329.shtm), DOE's Scientific R&D Approach to Cybersecurity Dec 2008 (http://www.er.doe.gov/ascr/ProgramDocuments/Docs/CyberSecurityScienceDec2008.pdf)), university centers (e.g., GTISC's Emerging Cyber Threats Report for 2009

(http://www.gtiscsecuritysummit.com/pdf/CyberThreatsReport2009.pdf)– Jan 2009)

[iii] (GAO Jul 2008)

[iv] S. Forrest and C. Beauchemin. Computer immunology. Immunol. Rev., 216, pp. 176-197, 2007.

[v] M. Newman, S. Forrest, J. Balthrop, Email networks and the spread of computer viruses. Physical Review E, 66, 035101:1-4, 2002.

[vi] J. Grenier, "The Challenges of CIP Interdependencies", Conference on the Future of European Crisis Management (Uppsala, 19-21 March 2001)http://www.ntia.doc.gov/cip/workshop/cipft_files/frame.htm.

[vii] M. Dunn and I. Wigert, International Critical Information Infrastructure Protection (CIIP) Handbook 2004. Swiss Federal Institute of Technology, Zurich, 2004

[viii] L.E. Lindler, F.J. Lebeda, and G. Korch Biological Weapons Defense: Infectious Diseases and Counterbioterrorism, Humana Press, New York, 2005.

[ix] T.C. Germann, K. Kadau, I.M. Longini, and C.A. Macken, "Mitigation Strategies for Pandemic Influenza in the United States," Proceedings of the National Academy of Sciences 103, 5935-40, 2006.

[x] http://www.pandemicflu.gov/plan/federal/pandemic-influenza-implementation.pdf

[xi] L. Sattenspiel, A. Lloyd. "Modeling the Geographic Spread of Infectious Diseases: Report on the Critical Review of Geographic Epidemiology Modeling Study." Prepared for the Defense Threat Reduction Agency, DTRA01-02-C-0035. April 2003. http://www.dtra.mil/asco/ascoweb/CompletedStudies.htm

Adaptive Security for MANETs via Biology

Richard Ford, Marco Carvalho*, William H. Allen, Frederic Ham
*Florida Institute of Technology, Institute for Human Machine Cognition**
<rford, wallen, fhm>@fit.edu, mcarvalho@ihmc.us

Abstract

Tactical Mobile Ad-hoc Networks (MANETs) have become a preferred way of providing edge connectivity in a diverse set of environments that do not rely on a fixed communications infrastructure. However, although MANETs can be used to fulfill diverse networking requirements, they face significant challenges with respect to security. While traditional security practices (such as patching and host hardening) are helpful, the collaborative and volatile environments in which MANETs operate require new approaches to security. In this paper, which describes our ongoing research, we describe some of the techniques which underlie a more holistic system for MANET security. Drawing heavily from biology and social interactions, we describe our new approach: BITSI - the Biologically-Inspired Tactical Security Infrastructure. By focusing on overall mission continuity as opposed to isolated simple security, we provide an adaptive framework that is mission-centric.

1. Introduction

Mobile Ad-hoc Networks (MANETs) represent a new and exciting way of providing connectivity in tactical environments. Requiring no central command and control, MANET nodes collaborate in order to route traffic and provide essential services.

While this approach has significant advantages, it is not without cost. The permeable nature of the network and the reliance of nodes on their peers makes security a challenging problem. Interaction with a malicious node disseminating bad routing information, for example, can wreak havoc. Furthermore, given that MANETs are a natural choice for battlefields and other mobile tactical environments, these systems are likely to motivate the development of novel attacks aimed at disrupting or denying critical services and communications.

In this paper, we describe our team's efforts to provide mission-centric security for MANETs. We begin by reviewing MANETs in general and why securing them requires a novel approach. Next, we briefly outline prior work in the field, and introduce the biological metaphors we draw from. Finally, we present initial results from our experiments, and their implications.

2. Tactical MANETs and Security

As described in [1] and [2], MANET security is a superset of the traditional security problems. A collaborative routing design and a continually shifting topology make approaches like network-based IDS significantly more complex – a situation exacerbated by the lack of any central point of control, disjoint connectivity, and low-computational power on many nodes.

For example, in MANETs nodes typically share information regarding their neighbors, allowing nodes to either proactively or reactively calculate the "correct" route for traffic within the network. In such a scenario, a single attacker can wreak havoc by advertising low-

cost routes to all end points and not forwarding the traffic. Similarly, an attacker could preserve power (a precious commodity in-field) by dropping traffic for others but still requesting others forward its own traffic. This type of disruptive behavior is just one of the raft of security-related issues a MANET-focused security system must not only detect but also *heal*. Note the emphasis on healing: as MANETs are typically used in dynamic environments, nodes must have the ability to fend for themselves with respect to security.

Prior researchers have taken a variety of approaches to address problems like these. [3, 4] have used reputation systems in an attempt to spot attackers collaboratively. Others, such as Sterne et al. [1] use self-organization to provide for adaptive reconfiguration of security monitoring tools. However, our belief is that while these approaches may form a useful part of an overall solution, a new paradigm for protection is needed. To this end, we designed BITSI, the Biologically-Inspired Tactical Security Infrastructure.

3. BITSI: Goals and Design

When considering the requirements for a MANET security system, our observation is that a viable solution should have the following properties:

- A MANET solution must be able to detect security threats generically, without absolute reliance on predefined signatures.
- A MANET solution must be able to dynamically reconfigure the network to mitigate the security threat autonomously
- A MANET solution must not rely on any fixed resources; instead, a MANET security solution must be able to organize autonomously, regardless of group size

With these requirements in mind, we realized that these properties are shared by many biological systems. For example, the invertebrate immune system can generically detect attackers (bacteria/viruses) in a process called innate immunity. This system then produces antibodies that allow for *adaptive immunity* where the body learns to immediately detect and kill pathogens that are already known (for an overview of biological immune systems, see [5]).

Traditionally, the mechanisms used by artificial immune systems have followed the basic roadmap described by Forrest et al. [6], based upon self/non-self discovery. However, recent advances in human immunology have questioned whether this is actually the correct biological metaphor. In a controversial paper, Matzinger [7] proposed that the immune system is modulated by the detection of damage (Danger signals) within nearby cells. This danger signal effectively turns on the adaptive part of the immune system.

While Matzinger's theory is still the subject of some debate, it defines a potentially useful framework for providing security in computers (see, for example, the body of work by Aikelin et al. [8]). Using this as our inspiration, we realized that we could also use danger signals within a MANET to drive a number of different security components, including routing, reputation and configuration. The realization of this vision is BITSI.

4. Implementation

BITSI is based on a small kernel running in a Trusted Platform Module (TPM) on host nodes (Figure 1). While a TPM is not a requirement for successful implementation, its existence significantly simplifies implementation, as it provides a trusted environment from which the BITSI kernel can monitor the machine. Using this kernel as the initial building block, BITSI uses each of the techniques described below.

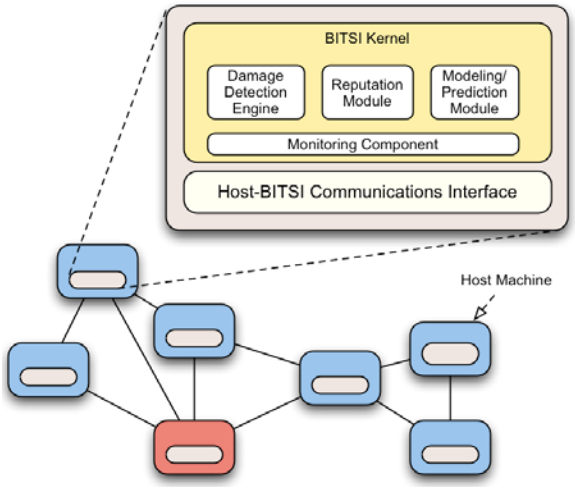


Figure 1. The BITSI kernel on a Trusted Platform Module (TPM).

4.1 Danger Theory

Our solution is based upon a seemingly nebulous concept of "damage". Making this idea more concrete has been one of the most significant challenges within BITSI. Our initial approach was based on understanding "expected" state transitions within an application and measuring highly detailed QoS parameters. However, when attempting to implement this approach we discovered it has large time requirements and ideally requires source code access to the application in question. Furthermore, our intuition is that the approach would be best for detecting the types of violation we can anticipate. Given the type of attacker envisaged, this is a poor assumption.

Based on our experiences, we have experimented with a very simple but powerful damage detection system. Each application needs to be able to provide BITSI with two damage detection events: (1) An application is not getting what it needs from another source, and (2) an application is getting "junk" or damage (for example, crashing) from a particular source. Function 1 can be easily expressed in terms of service (for example, an image collector is not receiving images it requested from a camera server). Function 2 can be detected by either detecting unexpected termination from an application, or by internally instrumenting an application to detect damaged input.

4.2 Reputation

Using our danger notifications described above, BITSI sends damage notifications hop-by-hop across the network. This is accomplished by each node keeping a ring buffer of traffic it has sent, received, or forwarded. Interestingly, our experiments have shown that this buffer of traffic need not enable perfect traceback of packets in order to provide useful results.

For both damage types, there are at least two possible causes. First, the packet may have been damaged or dropped in transit. In this case, the system should prefer, if at all possible, to not use this route again as one of the packet forwarders is unreliable. In our experiments, we leveraged the HSLS routing protocol [9] to distribute these routing reputations. This is novel as our changes do not increase the bandwidth required by the routing mechanism, and is accomplishing by changing the link weights HSLS already broadcasts when sending out route updates.

The second type of damage is caused when the packet source itself is the problem. In this case, when a node receives notification of damage, it changes its service reputation. This reputation is used when nodes choose which server should provide service. Given three servers, the one with the "best" group reputation is chosen. We have also experimented with using collaborative filtering so that the opinions that matter most are those from machines that have similar configurations to our own.

4.3 Machine Learning

Finally, we have experimented with using machine learning techniques to predict the state of the network in the future based upon reputation and host changes in the past. This technique is of particular interest as it directly addresses the challenges posed by detecting self-

replicating code *before* it has become ubiquitous.

Using a Radial Basis Function Neural Network (RBF-NN), we have been able to predict the probability of virus infection for various nodes based on changes in machine reputation and machine configuration. For example, consider a virus that is spreading using a vulnerability in a database server. For such a virus, only those machines that are unpatched are susceptible to infection. Furthermore, when the reputation of similar servers begins to decrease after a change, one should suspect a malware outbreak.

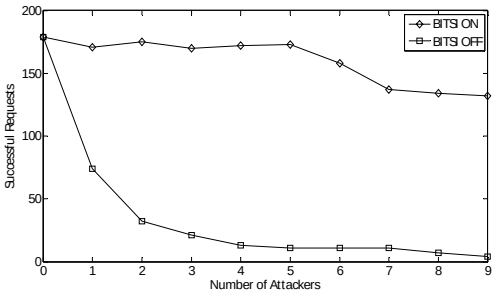


Figure 2. The performance of BITSI as a function of the number of attackers. With BITSI running, the overall throughput of the network is not significantly degraded.

5. Preliminary Results & Discussion

For each of the techniques described above, we have had encouraging results. For example, in a MANET of 22 "good" nodes (consisting of one client, one server, and twenty relay nodes) and up to 9 attackers, we demonstrated that BITSI is capable of significantly limiting the impact of attackers who deliberately corrupt traffic. In the experiment, we gradually added attackers to the network. These attackers behaved as normal relay nodes, with the caveat that they deliberately corrupt client requests to the server. As can be seen in Figure 2, BITSI provides significant protection to the system. Full details of this experiment are available in [10].

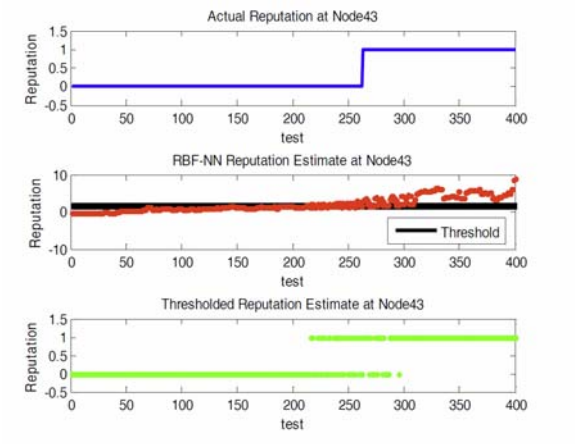


Figure 3. results from our reputation prediction work using machine-learning techniques. Note how BITSI is able to predict the change of reputation of a node based on past measurements and learned responses.

The graph shown in Figure 3 shows the output from our RBF-NN approach. This work holds great promise, and is covered in more depth in [11]. These predictive approaches are important as it allows nodes within the network to get "ahead" of outbreaks. Furthermore, these techniques can learn in an unsupervised manner, making them highly effective in a MANET environment.

6. Conclusion

In this paper, we have broadly described our work on securing MANETs using a variety of biologically-inspired techniques. To date, our results have been encouraging, and we believe these approaches have the ability to seriously impact current thinking in our field. Most notably, our approach does not attempt to impute motive; instead, we focus on the effect of the damage to the overall mission.

In terms of further work, much remains to be done. While we have built a version of BITSI that runs on real hardware in a test MANET, many interesting research issues remain. How should reputation damage be "forgiven" as a function of time? How effective is our generic damage assessment? Our current models for reputation are linear; are there more complex approaches that are as general yet more

effective? How vulnerable is BITSI to a coordinated attacked designed to cause an “auto-immune” reaction? These are only a few of the open research questions; questions we expect to be tackling for some time.

This research is part of a multi-institutional effort, supported by the Army Research Laboratory via Cooperative Agreement No. W911NF-08-2-0023.

[1] Sterne, D., Balasubramanyam, P., Carman, D., Wilson, B., Talpade, R., Ko, C., Balapari, R., Tseng, C.Y., Bowen, T., Levitt, K., and Rowe, J., A General Cooperative Intrusion Detection Architecture for MANETs, in: Proceedings of the Third IEEE International Workshop on Information Assurance (IWIA'05), Washington, D.C., pages 57--70, IEEE Computer Society, 2005

[2] Carvalho M., Security in Mobile Ad Hoc Networks, Security & Privacy, 6:2 (72-75), 2008

[3] Mundinger, J., and Le Boudec, J., Analysis of a reputation system for Mobile Ad-Hoc Networks with liars, in: Perform. Eval., 65:3-4(212--226), 2008

[4] Jaramillo, J.J., and Srikant, R., DARWIN: distributed and adaptive reputation mechanism for wireless ad-hoc networks, in: MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking, Montréal, Québec, Canada, pages 87--98, ACM, 2007

[5] Eales, L.J., Immunology for Life Scientists, John Wiley and Sons, 2003

[6] Forrest, S., Hofmeyr, S., Somayaji, A. and Longstaff, T., A Sense of Self for Unix Processes, in: Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy, pages 120--128, IEEE Computer Society Press, 1996

[7] Matzinger, P., Tolerance, Danger and the Extended Family, in: Annual Review of Immunology, 12(991--1045), 1994

[8] Kim, J., Greensmith, J., Twycross, J., and Aickelin, U., Malicious Code Execution Detection and Response Immune System Inpired by the Danger Theory, in: Adaptive and Resilient Computing Security Workshop (ARCS 2005), Santa Fe, NM, 2005

[9] Santivanez, C., and Ramanathan, R., Hazy Sighted Link State (HSLS) Routing: A Scalable Link State Algorithm, BBN Technologies, number 1301, BBN Technical Report, 2003

[10] Ford R., Allen W.H., Hoffman K., Ondi A., and Carvalho M., Generic Danger Detection for Mission Continuity, Preprint. Accepted to the 8th IEEE International Symposium on Network Computing and Applications, July 2009

[11] Ham, F. M., Imana, E. Y., Ondi, A., Ford, R., Allen, W., Reedy, M., Reputation Prediction in Mobile ad hoc Networks Using RBF Neural Networks, submitted to: 11th International Conference on Engineering Applications of Neural Networks (EANN), August 2009

Movement Speed and Camera Distance Measurement for Human Motion Detection based on Interocular Distance

Khandaker Abir Rahman
Center for Secure Cyberspace
Louisiana Tech University

Kiran S. Balagani
Center for Secure Cyberspace
Louisiana Tech University

Vir V. Phoha
Center for Secure Cyberspace
Louisiana Tech University

Chuka Okoye
Center for Secure Cyberspace
Louisiana Tech University

Email: {kar026, ksb011, phoha, cdo012}@latech.edu

Abstract

We present a novel interocular distance based human motion detection system. The distance between centers of two eyes is used to compute the speed, person to camera distance measurement and motion detection of a person. The variation in eye-distance (in pixels) with the changes in camera to person distance (in inches) is used to formulate the system. The proposed system will be relatively simple and inexpensive to implement as it does not require any other instruments other than a CCD camera.

1. Introduction

Methods based on human motion detection are widely used in many applications, such as human-robot interaction [1], smart surveillance [2] and motion analysis [3]. Methods for detecting human motion include background subtraction [4], template matching [5], optical flow [3] and temporal differencing [6]. These methods require stable background which may not always be possible. Therefore, background subtraction is not available. Because shapes and positions of moving persons always change and is hard to be described by a template, template matching is not available, either. The optical flow method is not suitable for a real time system because of its computational complexity and high sensitivity to noise. In this paper, we propose: (1) measurements of person to camera distance (2) movement speed measurement in real time.

Two widely used approaches for camera distance measurement are: contact and non-contact approaches [7]. The contact-based approach includes ultrasonic distance measurement [8, 9], laser reflection methods [10, 11]. These methods are based on the theory of reflection. If the reflection surface is not uniform, the measuring system generally performs poorly. On the other hand, non-contact measuring systems rely on pattern recognition or image analysis techniques [12, 13]. A drawback of these methods is that, they demand huge amount of storage capacity and high-speed processors. Also speed measuring systems require external tools like radar signals etc. To overcome these problems and difficulties encountered by the existing techniques, an image-based person to camera distance and speed measuring system without complex calculations is presented in this paper. The system setup and configuration of the proposed method is very simple, consisting of only a single CCD camera.

The paper is organized as follows. In Section 2, the proposed person to camera distance measurement system and speed measurement based on interocular distance is described. Experimental results and discussions are presented in Section 3. The paper conclusion is

in Section 4. Finally, the future works are described in Section 5.

2. Speed and person to camera distance measuring system

2.1 Eye distance measurement

The system forms an image pyramid of the input images and uses a template matching approach for face and eye detection [14]. An image pyramid is a set of the original image at different scales. To locate the face, a mask is moved pixel-wise over each image in the pyramid. At each position in the image, the mask is passed to a function that assesses the similarity of the image section to a face. If the similarity value is high enough with respect to specific threshold, the presence of a face at that location is assumed. From that location, the position and size of the face in the original image is generated [14]. From the detected face, eye is detected by forming an image pyramid and using a template matching approach. The Euclidian distance between two eyes is computed using the following equation (1):

$$d_{ep} = \sqrt{(E_{LX} - E_{RX})^2 + (E_{LY} - E_{RY})^2} \quad (1)$$

where (E_{LX}, E_{LY}) and (E_{RX}, E_{RY}) are the center points of left and right eyes respectively and d_{ep} is the distance between two eyes in pixels.

2.2 Formulation of person to camera distance measurement equation

Based on a preliminary study conducted over 35 people of both sexes and of different height ranges, it was found that a relation exists between eye distance (in pixels) and person to camera distance (in inches) [15]. Equations (2) and (3) are formulated on the nature of (Eye Distance) ^{2} Vs Person to Camera Distance plots of 35 people, which represents the plots in real-time [15].

$$d_{ep}{}^2 = \frac{MAX_{ed}}{(1 + \frac{d_c - Mid_G}{Mid_G})(\sqrt{d_c - MIN_{ed}} - 1)} \quad (2)$$

$$d_c' = d_c \pm V(2 - \frac{d_{ep}}{MAX_{ed}}) \quad (3)$$

where d_{ep} is the distance between two eyes, MAX_{ed} is the maximum eye distance point, MIN_{ed} is the minimum camera distance point, Mid_G is the midpoint of square of (Eye Distance) ^{2} Vs Person to Camera Distance plot , d_c is the primary camera to person distance (with error), d_c' is the corrected camera to person distance and V is the correction weight. Positions of MAX_{ed}, MIN_{ed}, Mid_G points are shown in Figure 1. These values are generalized considering the data collected of 35 people.

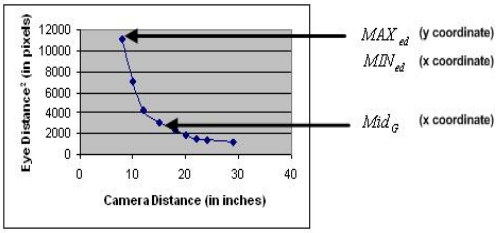


Figure 1. Relation between eye distance and object to camera distance

Before measuring the person to camera distance, the system is trained with different predefined distances from the camera starting from 7 inches and increased up-to 31 inches. During the training session corresponding person to camera distances (in inches) and eye distances are mapped and the MAX_{ed} value of that person (when the person is in the highest distance from the camera) is set by the system. It is also found that there are generally 5 categories of MAX_{ed} values ranging from 16000 to 9500 in which the persons tested have been categorized. Depending on the MAX_{ed} value, the other parameters of equations (2) and (3) are set according to Table 1. Figure 2 shows the

different square of (Eye Distance) ^{2} Vs Person to Camera Distance plots depending on different MAX_{ed} value. The values of Table 1 are set after analyzing the characteristics of square of (Eye Distance) ^{2} Vs Person to Camera Distance plots of Figure 2.

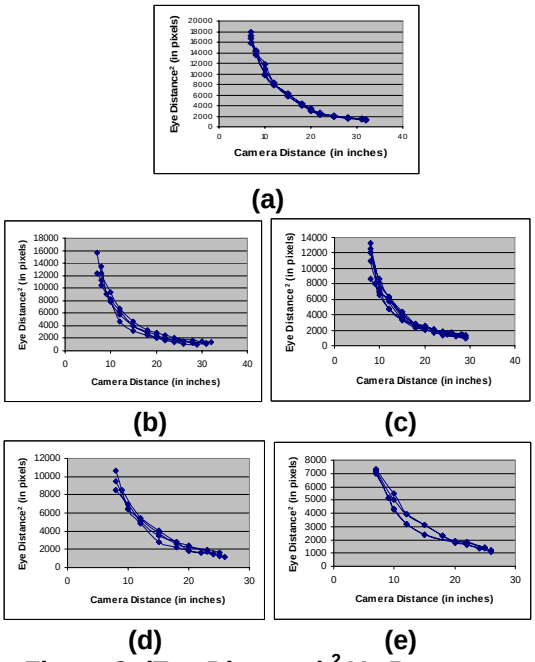


Figure 2. (Eye Distance) ^{2} Vs Person to Camera Distance plots (a) where MAX_{ed} >16000 (b) for 13000< MAX_{ed} <=16000, (c) for 11000< MAX_{ed} <=13000, (d) for 9500< MAX_{ed} <=11000 and (e) MAX_{ed} <=950

MAX_{ed} Range	MIN_{ed} Value	Mid_G Value	Value of V	Sign
MAX_{ed} >16000	8	23	8	+
13000< MAX_{ed} <=16000	8	20	6	+
11000< MAX_{ed} <=13000	8	18	4	+
9500< MAX_{ed} <=11000	8	15	0	N/A
MAX_{ed} <=9500	7	15	4	-

Table 1. Intrinsic parameter table

2.3 Person to camera distance measurement

Person to camera distance measurement is accomplished by calculating the eye distance and then mapping the corresponding person to camera distance from the generalized equations (2) and (3) with the values of the parameters from Table 2 after identifying the person along with corresponding MAX_{ed} value of that person. If the person is not identified then the default parameters values are chosen. Figure 3 shows the complete architecture of the proposed distance measuring system. The person to camera distance measurement algorithm is described bellow:

Step 1. Detect the center of the two eyes and find the Euclidian distance between them [14].

Step 2. If the person is identified then retrieve the MAX_{ed} value of that person from the database.

Step 3. Set the values of MIN_{ed}, Mid_G, V from Table 1 according to MAX_{ed}, where MAX_{ed} is the maximum eye distance point, MIN_{ed} is the minimum camera distance point, Mid_G is the mid point of Eye Distance^2-Camera Distance plot and V is the correction weight.

Step 4. Calculate primary camera to person distance, d_c from the equation (4).

$$d_c^{\,2}\,(d_c - MIN_{ed} - 1) = \left(\frac{MAX_{ed} \times MID_G}{d_{ep}^{\,2}}\right)^2 \qquad (4)$$

where d_{ep} is the distance between two eyes.

Step 5. Make correction to the camera to person distance by the following equation

$d_c' = d_c \pm V\,(2 - \frac{d_{ep}}{MAX_{ed}})$ where d_c is the primary camera to person distance (with error), d_c' is the corrected person to camera distance and V is the correction weight and return d_c'.

Step 6. If the person is not identified, set the default value as MAX_{ed} = 11000 and go Step 2.

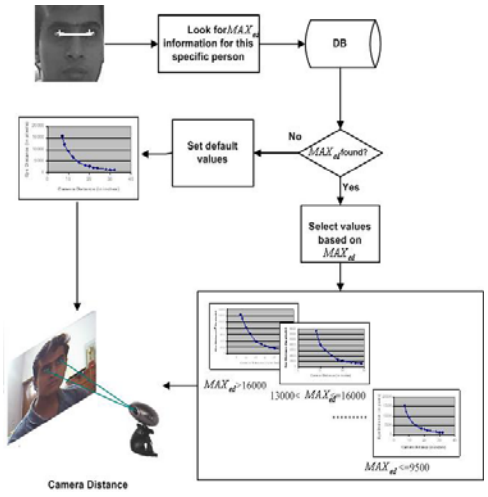


Figure 3. Person to camera distance measurement system architecture

2.4 Speed measurement

After computing the camera distance in real time the movement speed can be measured by the following equation (5)

$$s_i = \frac{\left| d_i - d_{i-1} \right|}{t_i - t_{i-1}} \qquad (5)$$

where s_i is the speed at t_i, $d_i = d_c'$ from equation (3) at t_i and $d_{i-1} = d_c'$ at t_{i-1}. Table 2 shows the relation of speed and interocular distance over time.

Time t_i (in sec.)	Interocular Distance (in pixel^2)	Speed (in ft/sec)
i = 0	1225	0
1	1370	3
2	1685	3
3	6277	13
4	2501	8
5	11211	12
6	4005	7
7	2034	7
8	8200	12
9	3000	8
10	12400	11

Table 2. Relation between interocular distance and speed

3. Experiments and results

This system uses *A4 Tech PK-336MB* CCD camera for image acquisition [16]. Each captured image is digitized into a 320×320 matrix with 24-bit color. The system captures 30 image frames per second. The system considers every 5th frame captured by camera for further processing. Thus the system processes 6 image frames per second for face area and eye detection [14].

Accuracy of distance measurement results using the proposed method are shown in Table 3, where real distances, measured distances, and accuracy (for distances from 7 inches to 31 inches) of 35 persons are recorded. Figure 4 shows the accuracy (%) of the proposed system at different predefined distances. The average accuracy of 94.11% is obtained. Though other conventional measuring results shows slight accurate where error rates range from 1 to 8% [17, 18], the proposed system validated its' superiority in terms of simplicity and cost effectiveness.

Actual person to camera distance (in inches)	System person to camera distance (in inches)	Accuracy (%)
31	33.8	88.96
28	31	90.25
25	26.7	93.2
22	23	95.45
20	20.3	98.5
18	18.2	96.88
15	14.5	96.66
12	10.71	93.25
10	9.24	92.4
8	8	97.55
7	7.76	92.14

Table 3. Accuracy of the distance measurement method

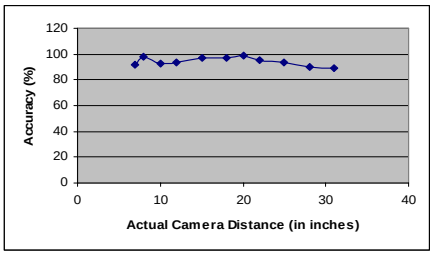


Figure 4. Accuracy (%) of the measured distance with the actual distance

Accuracy calculation of the speed measurement system is yet to be done and left for further research.

4. Conclusion

In this paper, a simple image-based person to camera distance and human movement speed measuring system is proposed. The proposed system is simple, cost effective and efficient for real-time implementation for human motion detection. Because of the simplicity of the proposed approach, obtaining a satisfactory person to camera distance measurement and speed calculation can be achieved without using hardware-intensive techniques, such as echo detection, additional CCD cameras, laser projector [19], flash lights etc. We plan to extend the current system for other human motion detection aspects such as direction of movement calculation based on interocular distance with a practical potential in the fields of person identification [20], security and robotics.

5. Future works

We would further extend this work in following direction:

a) Complete human motion detection for video surveillance
b) Accuracy measurement of the human motion detection and movement speed measurement.
c) Consideration of side face views and face rotation for improving the accuracy of measurement.
d) Human height and weight which influences the interocular distance needs to be addressed.
e) 3-d interocular distance consideration for more robustness.

6. References

[1] K. Tanaka, K. Yamano, K. Kondo, and K. Kimuro, "A vision system for detecting mobile robots in office environments," presented at Proceedings of IEEE International Conference on Robotics and Automation, 2004.

[2] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder:Real-time tracking of the human body," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 780-785, 1997.

[3] D. M. Gavrila, "The visual analysis of human movement: A survey, Computer Vision and Image Understanding," vol. 73, pp. 82-98, 1999.

[4] S. McKenna, S. Jabri, and Z. Duric, "Tracking groups of people, Computer Vision and Image Understanding," vol. 80, pp. 42-56, 2000.

[5] A. Lipton, H. Fujiyoshi, and R. Patil, "Moving target classification and tracking from real-time video," *In Proc. IEEE Workshop on Application of Computer Vision*, 1998.

[6] B. Jung and S. Sukhatme, "Detecting Moving Objects using a Single Camera on a Mobile Robot in an Outdoor Environment," *The 8th Conference on Intelligent Autonomous Systems*, pp. 980-987, 2004.

[7] C. Chen, C. Hsu, T. Wang, and C. Huang, "Three –Dimensional Measuremnt of a Remote Object with a Single CCD Camera," *Proceedings of the 7th Int. Conf. on Signal Processing, Computaional Geometry & Artificial Vision*, vol. 7, pp. 141-146, 2007.

[8] A. Carullo, F. Ferraris, and S. Graziani, "Ultrasonic Distance Sensor Improvement Using a Two-Level Neural Network," *IEEE Transactions on Instrumentation and Measurement*, vol. 45, pp. 667, 1996.

[9] A.Caarullo and M. Parvis, "An ultrasonic sensor for distance measurement in automotive applications," *IEEE Sensors Journal*, vol. 1, pp. 143-147, 2001.

[10] K. Osugi, K. Miyauchi, N. Furui, and H. Miyakoshi, "Development of the scanning laser radar for ACC system," *JSAE Rev*, vol. 20, pp. 554-579, 1999.

[11] H.-T. Shin, "Vehicles Crashproof Laser Radar," *M.S. thesis*, 2000.

[12] T. Kanade, H. Kano, and K. S, "Development of a Video-Rate Stereo Machine," *Proc 1995 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 3, pp. 95-100, 1995.

[13] K. Tanaka, g. Y, I. Nagai, and A. Mohamed, "Development of a Compact Video-rate Range finder and its application," *Proc. 3rd Int. Conf. on Advanced Mechatronics*, pp. 97-102, 1998.

[14] FaceVACS-SDK, vol. Version 1.9.9.

[15] K. A. Rahman, M. S. Hossain, M. A.-A. Bhuiyan, T. Zhang, M. Hasanuzzaman, and H. Ueno, "Person to Camera Distance Measurement based on Eye Distance," *Proceedings of the 3rd International Conference on Multimedia and Ubiquitous Engineering(MUE)*, 2009.

[16] www.a4tech.com.

[17] T.-H. Wang, M.-C. Lu, C.-C. Hsu, W.-Y. Wang, C.-P. Tsai, and C.-c. Chen, "A Method of Distance Measurement by Digital Camera," *Proceedings of 2006 CACS Automatic Control Conference*, pp. 1065-1069, 2006.

[18] M.-C. Lu, W.-Y. Wang, and C.-Y. Chu, "Image-Based Distance and Area Measuring Systems," *IEEE Sensors Journal*, vol. 6, pp. 495-503, 2006.

[19] T. Wang, M. Lu, W. Wang, and C. Tsai, "Distance Measurement Using Non-metric CCD Camera," *Proceedings of the 7th Int. Conf. on Signal Processing, Computaional Geometry & Artificial Vision*, vol. 7, pp. 1-6, 2007.

[20] K. A. Rahman, M. S. Hossain, M. A.-A. Bhuiyan, T. Zhang, M. Hasanuzzaman, and H. Ueno, "Eye Distance Based Mask Selection for Person Identification," *Proceedings of the 3rd International Conference on Multimedia and Ubiquitous Engineering(MUE)*, 2009.

Detecting (Approximate) Hole Coverage Areas in Wireless Sensor Networks*

Christian A. Duncan
Computer Science
Louisiana Tech Univ.
duncan@latech.edu

Jinko Kanno
Mathematics & Statistics
Louisiana Tech Univ.
jkanno@latech.edu

Rastko Selmic
Electrical Engineering
Louisiana Tech Univ.
rselmic@latech.edu

Abstract

In this work-in-progress paper, we introduce a formal mathematical definition of an approximate hole coverage area for wireless sensor networks. We then present a simple proof for a decentralized solution to the approximate hole coverage problem. The solution requires that each sensor has knowledge of its exact location but only requires communication with its one-hop neighbors. The aim is to extend the proof to more realistic models where exact location is not known but other less precise (or less costly) information is available.

1 Introduction

Considerable research has emerged on the use of Wireless Sensor Networks (WSNs) in a wide range of fields with applications ranging from the monitoring of environmental conditions such as soil temperature or fire detection to securing vulnerable sites such as the detection of chemical or nuclear agents.[1] To be effective, many of these problems require some guarantee on the coverage of the area sensored. That is, any gaps in sensor coverage could cause the WSN to miss critical information. An important measure of Quality of Service (QoS) in WSNs is the ability to detect and patch coverage holes. Once deployed, however, detecting gaps in the coverage of a WSN is complicated by several factors.

- With a low sensor range, many WSNs use large-scale deployment with many individual (inexpensive and small) sensors for maximal coverage.
- Since manual deployment can be either prohibitively costly (e.g., man-power) or impractical (e.g., dangerous terrain), a common solution is to use random deployment such as aerial dispersal.
- Since sensors often require low-power consumption to increase life span, they often lack GPS capability and must rely on some form of localization based on (limited) wireless communication.
- Even with GPS but particularly when using communication only, imprecisions exist about physical location.

There has been some progress in detecting and patching the coverage holes in wireless sensor networks. Buchart [2] used the connectivity of the sensors' communication network to model the WSN. Yao *et al.* [6] took this approach further by transforming the communication graph into a planar simplicial complex and identifying potential holes from this graph. Both approaches, however, addressed the problem heuristically, and the results were verified experimentally using a combination of simulated large-scale networks and smaller scale real-life deployment tests.

*This work was partially funded by Louisiana Board of Regents through PKSFI grant LEQSF(2007-12)-ENH-PKSFI-PRS-03.

Ghrist *et al.* [3, 4] approach the problem by attempting to infer sensing coverage using only local connectivity information, with no positioning. They introduce the novel idea of using the seemingly unrelated mathematical field of homology to solve sensor network coverage problems. However, their method works only for sensor networks which have a single hole, and in networks having multiple holes it likely will not identify all holes. Furthermore, there is no guarantee that their algorithm will exactly locate the hole.

1.1 Our Work

In this paper, we present work that we are pursuing to address the coverage issue from both a practical *and* theoretical perspective. In particular, we refine the coverage problem to one of detecting an approximate coverage problem and define several potential directions for proving and finding viable algorithms based on various levels of sensor information.

In Section 2, we define the coverage and approximate coverage problem. In Section 3, we discuss some of the possible variants of sensor networks that could lead to formal proofs of (approximate) coverage. We finish with some concluding remarks in Section 4.

2 (Approximate) Boundary Coverage in Wireless Sensor Networks

In order to begin to prove formally the accuracy of any algorithm to detect coverage holes in WSNs, we start by defining some key terms leading to a formal definition of what constitutes a valid detection of holes. In particular, our aim is to identify a collection of "boundary" sensors that are identifiers of the gaps in coverage. We begin by defining the regular boundary coverage problem.

Let S be a collection of sensors distributed in the Euclidean plane. Let r_c and r_s represent the communication and sensor radii of the sensors.

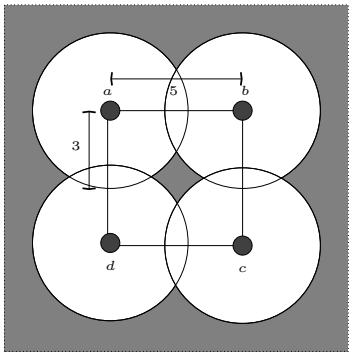


Figure 1: Boundary coverage for four sensors a, b, c, and d. Here $r_c = 5$ and $r_s = 3$. The shaded region shows the entire hole region. In this example, all four sensors are boundary sensors.

For any two points p and q, let $d(p, q)$ represent the (standard) Euclidean distance between the points. Note, for now we focus on Euclidean space, but point out that in practice other non-Euclidean and non-metric spaces are common, for example, due to terrains with obstructions, and are worthy of further study as well.

Two sensors s and t can ***communicate*** if $d(s,t) < r_c$. A point p (on the plane) is ***covered*** if there exists a sensor $s \in S$ such that $d(s,p) \leq r_s$. The union H of all ***uncovered*** points in the plane forms a collection of disjoint ***hole regions (holes)***. Let δ_H represent the boundary of these hole regions. A ***boundary sensor*** s is defined to be on the boundary of a hole if there exists a point $p \in \delta_H$ such that $d(s,p) = r_s$. The ***boundary coverage problem*** is to find the set of all boundary sensors in S, see Figure 1.

Ideally, our goal would be to identify every sensor that lies on the boundary of the hole regions. This allows for the identification of locations for the deployment of further sensors to the regions to attempt to cover the holes. However, as is common, if the exact GPS location of each sensor is not known a priori, for example due to a lack of a GPS on the sensors, this ideal goal becomes infeasible. As a result, we propose solving an *approximate* boundary coverage problem. To that end, we refine a few

terms and refer to these updated definitions for the remainder of the text.

Let r_s^+ and r_s^- represent an upper and lower bound on the sensor radius (outer and inner radii) as follows. A point p is ***covered*** if there exists a sensor $s \in S$ such that $d(s,p) < r_s^-$. (That is, the point is within the inner radius of some sensor.) A point p is ***uncovered*** if $d(s,p) > r_s^+$ for every sensor $s \in S$. (That is, the point is outside of the outer radius of every sensor.) Another way of looking at it is a point p is covered if the (open) circle of radius r_s^- centered at p contains a sensor s and is uncovered if the (closed) circle of radius r_s^+ centered at p is empty (of sensors). The ***unclassified*** points that fall in between these two categories may or may not be considered covered by the algorithm. See Figure 2(a).

Let an ***approximate hole region*** H be a subset of the plane such that for every p in the plane if p is uncovered then $p \in H$ and if p is covered then $p \notin H$. That is, H contains *all uncovered* points and no *covered* points. It may contain any subset of *unclassified* points. Again, let δ_H represent the boundary of this (approximate) hole region. An **approximate boundary sensor** s is defined to be a sensor (in S) such that there exists a point $p \in \delta_H$ such that $d(s,p) \leq r_s^+$. The **approximate boundary coverage problem** is to find a subset S' of the approximate boundary sensors such that for every boundary point $p \in \delta_H$ there exists at least one $s \in S'$ such that $d(s,p) \leq r_s^+$.

Essentially, we are finding a subset of the sensors that together covers the boundary of the approximate hole region, see Figure 2(b). The identification of these sensors provides an alert to where potential gaps in coverage exist and indicates areas where new sensors need to be deployed.

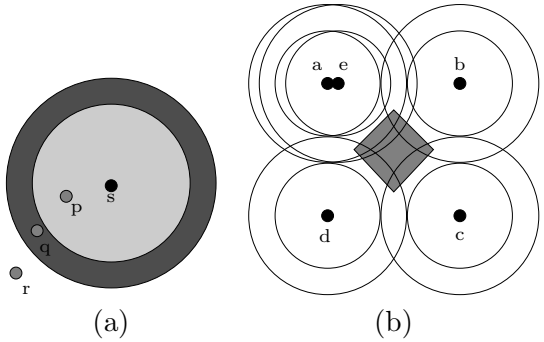


Figure 2: (a) With respect to the sensor node s, point p is covered, r is uncovered, and q is unclassified. The lightly shaded region represents the set of all points covered by s and the darker annulus region represnts those points possibly covered by s. (b) In this example, $r_c = 5$, $r_s^+ = 3$, and $r_s^- = 2$. Here an approximate hole region (not counting the exterior) is highlighted. The sensor nodes a, b, c, and d form an approximate boundary coverage solution (even though node e is also an approximate boundary sensor).

3 Wireless Sensor Network Variations

There are several variations to the problem depending on the type of sensor network information that we have. These variations can be categorized in numerous ways. For instance, a ***centralized solution*** would allow the use of a single (central) server to gather information and determine hole boundaries and deployment requirements. Whereas, a ***decentralized (localized) solution*** would require that *individual* sensors using information gathered in their general neighborhood, the definition of which yields further variations, determine the hole boundaries. Our focus for this work shall be on the *decentralized* version. Let us address a few extremes of this variant.

If every sensor has a GPS device and thus (nearly) complete knowledge of its location and, via communication, those of its neighbors the problem seems quite reasonable. In fact, a centralized solution is quite trivial. It is simply a straightforward geometric problem once the locations are all determined and collected by the central server. When only local information is known, however, the solution is not quite so trivial even if it is intuitively simple. Using our problem definition, we can, however, go be-

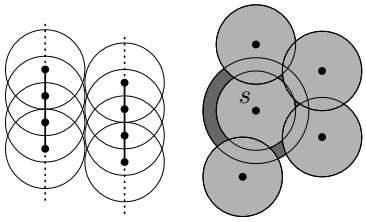


Figure 3: (a) A simple example of the need for a reasonable communication size. Here the nodes on the left cannot communicate with the nodes on the right. The removal of the holes depends on shifting the nodes only slightly closer to each other where they still remain outside of communication range. (b) The annulus region A_s around sensor s and B, the union of the ball regions around s's neighbors. The dark shaded region represents all of the potentially uncovered points that indicate s to be a boundary sensor node.

yond intuition and prove the correctness of a given algorithmic solution. First, assume that $r_c \geq 2r_s^+$, the communication extends beyond a sensor's outer diameter. With communication smaller than the sensor's radius, particularly $r_c < 2r_s^-$, it can become nearly impossible to determine boundary sensors using local information only, see Figure 3(a).

Theorem 1. *For $r_c \geq 2r_s^+$, if every sensor in the network is aware of its exact position, there exists a decentralized solution to the approximate boundary coverage problem.*

Proof: For each sensor $s \in S$, let T be the set of sensors *local* to s, that is, the set of sensors (not including s) that can directly communicate with s (one-hop distance). Let $A_s = \{p \in R^2 : r_s^- \leq d(s,p) \leq r_s^+\}$ represent the annulus centered at s with inner radius r_s^- and outer radius r_s^+. This annulus represents the set of points on the *fringe* of s, those that may or may not be covered by s, see Figure 2(a). For any sensor $t \in S$, let $B_t = \{p \in R^2 : d(t,p) < r_s^-\}$ be the ball (circle) of radius r_s^- centered around t. That is, B_t is the set of points explicitly covered by t. Let $B = \bigcup_{t \in T} B_t$ be the union of all the balls associated with the neighboring sensors of s. If there exists any point $p \in A_s$ that is not in B, in other words if $A_s \setminus B \neq \emptyset$, then s considers itself a boundary node, see Figure 3(b). Otherwise, it does not.

We claim that the above localized (decentralized) algorithm correctly solves the approximate boundary coverage problem. That is, the set S' of sensor nodes that consider themselves boundary nodes is a valid solution to the problem. To prove this we must show two things. First, that each node in S' is in fact an approximate boundary node, and secondly, that every boundary point $p \in \delta_H$ has at least one sensor $s \in S'$ such that $d(s,p) \leq r_s^+$.

Recall that an approximate boundary sensor s is a sensor such that there exists a point $p \in \delta_H$ such that $d(s,p) \leq r_s^+$. Let s be a sensor which our algorithm determined to be a boundary node. We know there must be a $p \in A_s \setminus B$ (as the region is not empty). We claim that p is on the boundary of some approximate hole region H. To do this we must prove that p is either uncovered or unclassified, or simply *not* covered.

Assume for the sake of contradiction that p is covered. This means that there exists a sensor $u \in S$ such that $d(u,p) < r_s^-$. We know that $u \neq s$ since p is in A_s and so $d(s,p) \geq r_s^-$. Similarly, we know that $u \notin T$ (the neighbors of s) because p lies outside B and so $d(t,p) \geq r_s^-$ for all $t \in T$. If u is a sensor not within communication range of s then we know that $d(s,u) > r_c \geq 2r_s^+$. Since $p \in A_s$, however, we know that $d(s,p) \leq r_s^+$. By the triangle inequality, we know that $d(s,u) \leq d(s,p) + d(u,p)$. Or, $d(u,p) \geq d(s,u) - d(s,p) > 2r_s^+ - r_s^+$. So u cannot be one of these sensor nodes either. This contradicts the fact that p is covered. And consequently, s is a valid boundary sensor.

Now let $p \in \delta_H$ be any point on the boundary of an approximate hole region H. This implies that there exists at least one sensor $s \in S$ for which $d(s,p) \leq r_s^+$ (within its outer radius) and no sensor t for which $d(t,p) < r_s^-$ (outside all inner sensor radii). We claim that s would re-

port itself as a boundary sensor node. To do this, we must only show that $p \in A_s \setminus B$. Since $d(s,p) \leq r_s^+$ and $d(s,p) \geq r_s^-$ we know that $p \in A_s$. Since $d(t,p) \geq r_s^-$ we know that $p \notin B_t$ for any $t \in T$. Therefore, $p \notin B$ and subsequently $p \in A_s \setminus B$ implying that s reports itself as a boundary sensor.

Hence, the set of sensor nodes identified as (approximate) boundary nodes is a valid decentralized solution. □

We note that our solution and proof does not actually rely on any real distinction between r_s^+ and r_s^- and hence could technically be proven for an exact solution (assuming exact positions are known). This of course is only useful when GPS devices are available for every sensor node.

We wish to loosen this constraint to allow for inexact locations and exploit the approximate nature of the problem statement to modify the above proof. In fact, neither our algorithm nor our proof relies explicitly on location, simply relative positions with regard to each individual sensor node. This means that we do not need to do actual *localization*, a problem well studied in the literature. These problems in localization however typically demonstrate the viability of the solutions *empirically*, since obtaining a true global position is complex to show. We can instead use similar strategies to prove more rigorously that we have a good *local* estimate of position and subsequently construct and prove a decentralized solution to the approximate boundary nodes.

4 Closing Remarks and Future Directions

In this paper, we outlined a general framework for solving the approximate boundary coverage problem and showed a very simple mathematical proof for a *decentralized* algorithm where each sensor is capable of making a boundary decision by only using communication from its single-hop neighbors. The method unfortunately relies on the knowledge of global position, though no knowledge of the entire network.

Our goal is to answer the problem for less costly sensors. In the absence of GPS devices, several approaches are available for getting a rough idea of position. As in [5], we can look at sensors schemes that are ***range-based*** and ***range-free***. In the former, the sensors, using communication strengths for example, know absolute distance and/or angular estimates to their neighbors. The latter case does not make such an assumption.

Assume that for each pair of neighboring sensor nodes, s and t, we have an estimate on their distance $d(s,t)$ within an error margin (factor) of $\pm\epsilon$. We feel that under certain conditions for r_s^-, r_s^+, and r_c, dependent on ϵ, we can prove in a similar manner to the above theorem that a *decentralized* algorithm exists to compute the approximate boundary coverage problem.

Our current plans are to determine experimentally what are good estimate bounds for ϵ, to prove the relationship between the range in sensor radii (outer and inner) and this ϵ value, and to implement and experimentally validate the resultant algorithm both in a simulated and real environment.

References

[1] I.F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102–114, Aug 2002.

[2] J. T. Buchart. Detecting coverage holes in wireless sensor networks. Master's thesis, Louisiana Tech University, 2008.

[3] V. de Silva and R. Ghrist. Homological sensor networks. *Notices of the American Mathematical Society*, 54(1):10–17, 2007.

[4] Robert Ghrist and Abubakr Muhammad. Coverage and hole-detection in sensor networks via homology. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*,

page 34, Piscataway, NJ, USA, 2005. IEEE Press.

[5] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek F. Abdelzaher. Range-free localization and its impact on large scale sensor networks. *Trans. on Embedded Computing Sys.*, 4(4):877–906, 2005.

[6] Jixing Yao, Guyu Zhang, Jinko Kanno, and Rastko Selmic. Decentralized detection and patching of coverage holes in wireless sensor networks. In Stephen Mott, John F. Buford, Gabe Jakobson, and Michael J. Mendenhall, editors, *Proc. of SPIE*, volume 7352, page 73520V. SPIE, 2009.

Detecting and Combating Compromised Platforms in a Mobile Agent Infrastructure

Jeremy Kackley
School of Computing
University of Southern Mississippi
Hattiesburg, MS 39406-5106
Jeremy.Kackley@gmail.com

Paulus Wahjudi
Weisberg Division of Engineering
and Computer Science
Marshall University
Huntington, WV 25755
Paulus.Wahjudi@gmail.com

Abstract

Mobile code is an interesting idea that unfortunately does not see much real world application primarily due to security concerns. There are several avenues available for addressing these concerns. We present a preliminary methodology for detecting and dealing with compromised devices within a network of computers hosting sandboxes for mobile agents. This methodology makes use of the mobile agents traversing the network for detecting and dealing with the compromised devices.

1. Introduction

Mobile agent technology offers a new computing paradigm in which a program, in the form of a software agent, can suspend its execution on a host computer, transfer itself to another agent-enabled host on the network, and resume execution on the new host. The state of the running program is saved, transported to the new host, and restored, allowing the program to continue where it left off [3]. Mobile-agent systems differ from process-migration systems in that the agents move when they choose, typically through a ``jump" or ``go" statement, whereas in a process-migration system the system decides when and where to move the running process. As the sophistication of mobile software has increased over time, so to have the associated threats to security.

There has been a great deal of research and speculation about mobile code. Unfortunately, real world implementations of mobile code are fairly rare while actual applications are even rarer. This is for a variety of reasons; but one of the main reasons for the sluggish adoption of this paradigm is security[7,8]. Securing data, and work stations, is very important for all applications, but is extremely critical for applications dealing with government agencies. One of the underlying problem with network security in general and mobile agents in specific is authentication. The vast majority of effort is spent on positively identifying an entity as an authorized and trusted user. Unfortunately, authentication alone does not provide sufficient security [1,2]. A trusted entity might become compromised, and thus untrustworthy, despite being positively identified. Determining if an entity has been compromised can be a complicated process, and would usually be domain specific [10].

In terms of mobile code, or mobile agents, there are two aspects of trustworthiness. The first aspect is the trustworthiness of an individual piece of code, or agent. This is difficult to do, and generally speaking, the idea of a 'sandbox' is employed to theoretically ensure that the mobile code is incapable of doing anything nefarious or damaging. This idea is illustrated in figure 1. The second aspect is

determining if the agency, or sandbox, in which the agent resides is trustworthy. The payloads of agents might be critical. Additionally, sending agents into a 'bad' agency is wasteful, so this should be avoided. We address the determination of agency trustworthiness.

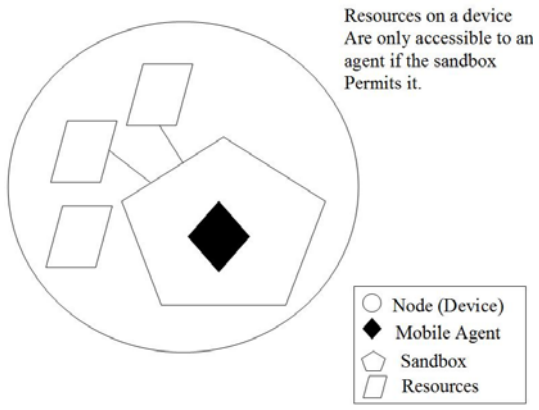


Figure 1 "Sandbox": the Agent is unable to access resources on the computer unless the sandbox provides an accessors to the agent.

In the next section we propose a methodology for monitoring the activity of a network of agencies for the purpose of automatically detecting suspicious behavior. The approach includes both active and passive detection mechanism. We define various stages of network situational awareness with corresponding protocols to be followed and executed.

2. Network Threat Levels

Four threat levels, numbered from one to four are proposed. Threat level one is the default threat level and is primarily a passive information gathering stage. Each threat level is increasingly proactive, with the final threat level taking action against suspicious nodes. There is a threshold value defined as T_n for the acceleration of each threat level. This value has yet to be determined, however, it is expected to require experimentation to accurately determine an accurate value. Indeed, it is possible that the appropriate threshold value might vary over time, even for the same network, thus this parameter will be left up to the network administrator.

2.1 Level One: Network Observation

Level One is the minimum threat level. It can be considered as situation normal. The key idea of this level revolves around seeding the network with probes that stay in place on each node. These probes are specialized agents that limit themselves to a single instance per node. They stay in memory and collect various data about the events that occur at the node. Threat level one also defines a Central Authority Node (CAN). This node is dedicated to threat detection/prevention, and the security of the system depends upon this node. As regular agents travel around the network, they collect data from the nodes they happen to visit, and eventually, upon task completion and return, send this data to the Central Authority Node. This process is shown in figure 2.

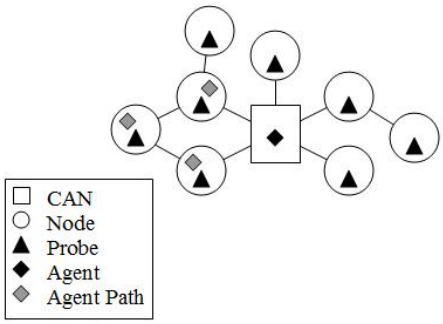


Figure 2 Network Observation: An agent traversing a network and bringing data back to the CAN.

The idea behind this scheme is to continuously gather survey data about the network, without utilizing excess network traffic [6]. This survey data is likely to be incomplete, especially if there are remote parts of the network that do not have high traffic. This is an

acceptable trade off for performance, considering that isolated nodes are likely to be less critical and important. Additionally, low traffic probably also corresponds to a lessened risk of corruption.

The aggregated survey data is periodically analyzed for anomalies by the Central Authority Node. The anomalies of interest should vary a bit depending on the domain. In the most basic model, the ratio of agents sent to a node, and arriving from a node over a period of time, should approach 1; and if it does not, this is an anomaly. Other types of anomalies can be defined dependent on the domain. While a certain number of anomalies can be expected due to the nature of network traffic, high levels of anomalous data might indicate a compromised node. The focus is on data from adjacent nodes, since nodes that are not adjacent should not have a large bearing on each other. The threshold value T_1 represents the amount of anomalies around a node before that node is considered a suspect. This threshold value again depends on both the anomalies and domain. Once anomalies are defined for a system; then a base frequency of anomalies can be measured. This frequency can then be used, to determine a threshold value. Once the defined threshold has been reached, the system progresses to threat level two without interrupting the processes of threat level one.

2.2 Level Two: Network Suspected Compromise Investigation

Level Two takes effect when the Central Authority Node has a list of suspected nodes. The central authority dispatches additional agents to further analyze the suspected nodes. These agents are Commander Agents and Detective Agents. Several detective agents report to a single commander agent. Detective agents are dispatched to each node adjacent to the suspect node. Detective agents are proactive versions of the passive probes from level one. These agents actively monitor the traffic around the suspected agency, and provide real time reports to the commander agent where possible. Any node that has a suspected node as a bottleneck cannot be trusted; and therefore cannot contribute to the investigation. The arrangement of commander and detective agents are depicted in figure 3. These reports are very similar to the anomaly detection defined in level one. The primary difference is the fact that the reports should incur very little latency before being received by the Commander Agent. This helps to minimize network latency; which could have a big impact on anomaly frequency. Additionally, the commander agent accesses which nodes are trustworthy when placing detective agents, and thus avoids manipulation of aggregate data by the compromised node. Depending on the domain; it might make sense to define more types of anomalies for this level than are defined for level 1.

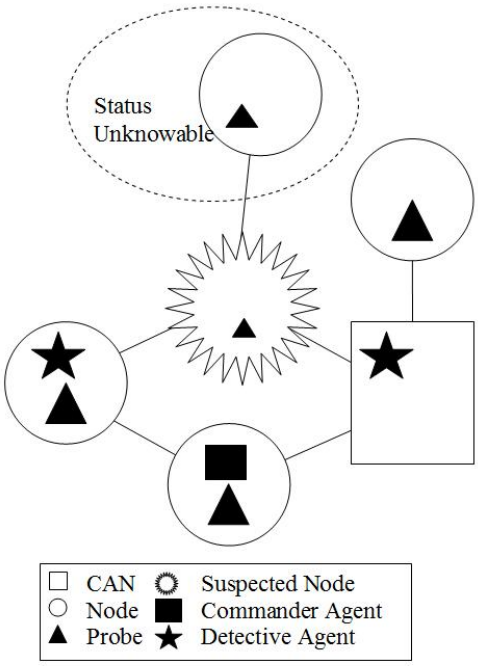


Figure 3 Network Suspected Compromise Investigation: Commander Agent and Detective Agents are dispatched to a Suspected Node.

Commander agents can be thought of as a super agent, or a miniaturized version of the central authority. Commander agents are dispatched to the most trusted node nearby the suspected node, and from here analyze the information gathered by their attached detective agents. This real time data should contain fewer anomalies than the passively gathered data, if there is no compromise. In the case of a compromised node, there is a second threshold value, T2, which must be passed in order to accelerate to level three. The threshold value T2; similarly to T1, is dependent upon the anomaly frequency of the domain.

2.3 Level Three: Network Compromise Confirmation

Once elevated to level three, there is sufficient evidence that a node has been compromised and such threat must be taken seriously. A Secret agent is created, and sent to the suspect node. The assumption is made that an agent within a sandbox, is completely at the mercy of its environment. At best, it can encrypt its data and prevent the environment from reading it. The secret agent is called a secret agent because it is designed to appear as just another agent. In fact, once it arrives at the suspect node, it will act like a regular agent, taking innocuous actions. The difference is that the actions and their sequence is predetermined and encrypted by the Commander agent.

Immediately after the secret agent has completed all of its actions, it attempts to establish contact the Commander agent, communicating the perceived results of its action. The commander agent has the observed results of the actions from the detective agents, and can compare what was supposed to happen, to what actually happened. There is a high probability that the agent will never be heard from again and in that case the only recourse is to retry with a new agent until feedback is gained or a predetermined number of attempts have been made. There is a possibility that computer and network errors could crease the anomalies that caused the threat level to be raised, and caused the secret agent to vanish, and retrying attempts to distinguish between error and maliciousness. Irregardless of which event occurs, at that point, either the agency is exonerated, or the threat level will be raised to level four. The interaction of the agents in this level are portrayed by figure 4.

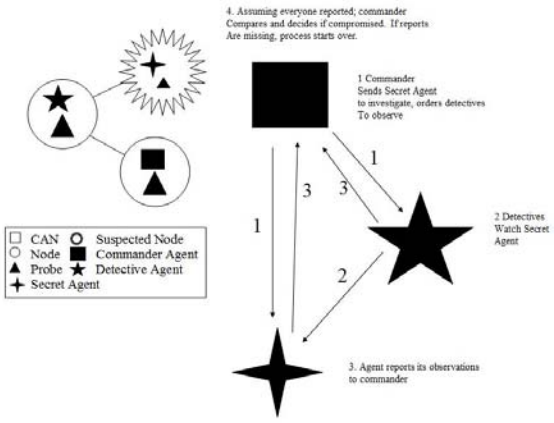


Figure 4: Network Compromise Confirmation: Coordination of Detective Agents and Secret Agents by Commander Agent in order to confirm a compromise has occurred.

2.4 Level Four: Network Compromise Resolution

The final threat level implies that a particular suspected agency is considered compromised by the system. There are a variety of actions that can take place at this point. The action taken would depend on the domain; where the importances of resources being available are weighted against the risk inherent in compromised resources. Actions fall into two groups, human decision and automated response.

In situations where resources being available are more important than information security, the system could simply raise an alert and ask a network administrator to make a judgment call

and take action. The action taken might involve many things, but in general involve the human administrator trying to ascertain if the machine actually is compromised, and attempting to clean it if so. Sometimes, security might outweigh the important of having resources on line. This is particularly true if the data is sensitive, and or the resources are redundant. In this case, an automated response might be in order; if only because an automated response would react much quicker than a human being would be able to. These can be rated in order of severity, which again corresponds to how important security is. Figure 5 presents four different potential responses to a compromised node. The simplest action, shown in figure 5 diagram 1, is to simply notify the system administrator so that a human can determine if the node is actually compromised and decide what action to take.

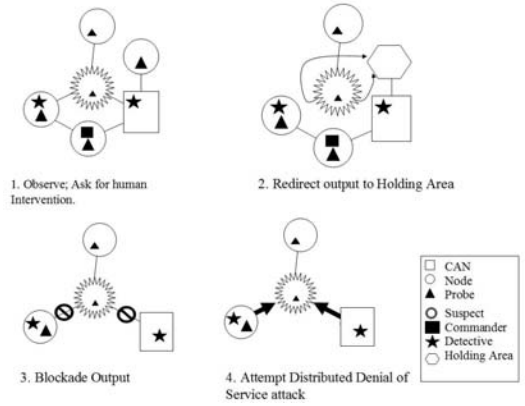


Figure 5 Network Compromise Resolution: A selection of potential resolutions for a compromised node.

Of the automated actions, the least radical action is to reroute all requests from that node to a secure sandbox that traps them and stores them for further analysis. This is shown in figure 5, diagram 2. Taking this action would protect other nodes from compromise, while not making it obvious that the compromise has been detected. Presumably, this situation would persist until another mechanism took over, and indicated to the system that the agency was cleaned. Another potential action is to black list the compromised agency. This would involve preventing any agents from traveling to the agency, and stopping any agents from leaving it. In effect, blockading the agency until the situation can be dealt with by outside forces, similar to the rerouting response. Diagram 3 in figure 5 illustrates the blockading of an agency.

The most severe automated action is to initiate a distributed denial of service attack against the compromised resource, possibly in conjunction with other methods. This action is illustrated in figure 5, diagram 4. The hope is that the strain will force the resource off line and prevent further damage to it and access to its data. This situation would likely be accompanied by retesting if the machine came back on line. Due to the severe nature of the action, a human analyst will be notified to ensure a resource is clean and diffuse the situation.

3. Future Work

First and most obviously, research into what combinations of data can most easily indicate a compromised node needs to be conducted. This data will be difficult to obtain quantitatively due to the relative rarity of mobile agent implementations and the difficulty of detecting a real world compromised node and studying it. The data will most likely be selected logically initially. Once implemented, it should be possible to have a feedback system in which the analysis of actual compromised nodes can be used to quantitatively asses the data and refine its selection.

Additionally, algorithms need to be developed to efficiently mine this data to detect anomalies that indicate a compromised system. These algorithms will depend on the data selected. A logical choice, assuming the data can be normalized, would be a type of clustering algorithm, because outliers would naturally be

anomalies. Care must be taken not to discard the concept of time, as outlier detection algorithms often do. Time in this situation is very important; naturally data should be compared to data that is temporally related to it.

Quantitative work must be undertaken to determine reasonable threshold values for the escalation of threat levels. While this is something of a parameter of the system; it is hard to predict these values without feedback. Research must be undertaken to determine the rate of anomalies in an non-compromised system, because the threshold values should be greater than this rate. This rate, itself, is likely to be dependent upon the network architecture. Universal threshold values might be impossible to determine; and reasonable thresholds for error might vary depending on network load.

There is an additional issue that comes into play in level 2. This issue revolves around the fact that once a node is suspected of being compromised; any communications passing through this node also automatically become suspect. If this node is a bottleneck which must forward many messages in order for them to reach the CAN, this could be a serious issue. This issue is indicated by figure 4 but the reason this could be a major issue is better illustrated by figure 6. In figure 6, the majority of the network becomes untrustworthy because it is 'behind' a potentially compromised node. This issue complicates detective and commander assignments in level 2 due to the fact that neither a detective nor commander should be allowed to traverse a compromised node. Care will have to be taken when developing the algorithms for detective and commander assignment. Additionally; research should be undertaken to determine a way to circumnavigate this issue due to the fact that nodes on the other side of a compromised node are unreliable makes it harder to detect anomalies due to a reduced sample size. A possible approach is encryption, although encryption alone might be insufficient.

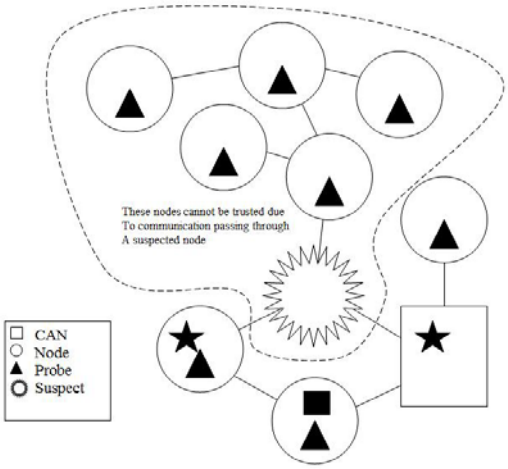


Figure 6 Trust Issues: Due to the arrangement of the network; many nodes in this situation cannot be trusted, and thus cannot be used to ascertain whether or not the suspected node is compromised.

4. References

[1] M. Asaka, S. Okazawa, A. Taguchi and S. Goto, "A method for tracing intruders by use of mobile agents", INET'99, June 1999.

[2] J. Balasubramaniya, J.O Garcia-Fernandez, D. Isacoff, E.H. Spafford and D. Zamboni, "An architecture for intrusion detection using autonomous agents", Department of Computer Science, Purdue University, Coast TR 98-05, 1998.

[3] M.J Bradshaw, "An introduction to Software agents", In Jeffrey M. Bradshaw, Editor, Software Agents, Chapter 1, AAAI press, The MIT press, 1997.

[4] K. Deeter, K. Singh, S. Willson, L. Filipozzi and S. Vuong, "APHIDS: A Mobile Agent-Based Programmable Hybrid Intrusion Detection", Journal of Systems and Software, 2004.

[5] J.D. De Querioz, L.F.R Da Costa Carmo and L. Pirmez, "Micael: An autonomous mobile agent system to protect new generation networked applications", 2nd annual workshop on Recent Advances in Intrusion Detection, September 1999.

[6] G. Helmer, J.S.K Wong, V. Honavar, L. Miller and Y. Wang, “Lightweight agents for intrusion detection”, Journal of Systems and Software, 2003.

[7] W. Janse, P. Mell, T. Karygiannis and D. Marks, “Applying Mobile Agent to Intrusion Detection and Response”, NIST Interim Report IR–6416, 1999.

[8] C. Kruegel and T. Toth, “Applying Mobile Agent Technology to Intrusion Detection”,Technical Report Number TUV-1841-2002-31, Technical University of Vienna, 2002.

[9] C. Kruegel, T. Toth and E. Kirda, “Sparta – A Mobile Agent-based Intrusion Detection System”, Technical Report Number TUV-1841-2002-24, Technical University of Vienna, 2002.

[10] W. Wang, S.R Behera, J. Wong, G. Helmer, V. Honavar, L. Miller, R. Lutz and M. Slagel, “Towards the Automatic Generation of Mobile Agents for Distributed Intrusion Detection System”, Journal of Systems and Software, 2006, pp.1–14.

Integrating Fuzzy Logic with FPGA-based Technology for Network Intrusion Detection

Marbin Pazos-Revilla
Computer Science M.S. Student
Tennessee Technological University
mpazos@tntech.edu

Ambareen Siraj
Computer Science Assistant Professor
Tennessee Technological University
asiraj@tntech.edu

Abstract

The costs associated with the disruption of crucial network services, and the damages caused by malicious attacks can be devastating to any organization. To prevent and mitigate these attacks considerable amounts of resources are used in deploying devices like Intrusion Detection Systems (IDS). IDSs act as security watch dogs and report security violations resulting from attacks. Although they have been proven useful, the inherent nature of conventional rule-based IDSs and the trends in bandwidth growth, among other factors, still provide loopholes allowing attacks to fall through cracks and remain outside radar. This research presents a novel approach integrating Field Programmable Gate Arrays (FPGA) and Fuzzy Logic in the field of network intrusion detection. The FPGA-based Fuzzy IDS addresses the aforementioned issues in conventional rule-based IDSs and have the potential to provide high throughput, parallelism, low non-recurring engineering costs, and the capability of inexact reasoning with its embedded Fuzzy Inference Engine.

1. Introduction

For several years the use of Intrusion Detection Systems (IDS)s in enterprise networks as means to detect possible attacks or suspicious network traffic has proven to be of paramount importance. Especially in today's environments where computing resources tend to have an increasing distributed nature crossing organizational and geographical boundaries and the reach of potential target nodes adding up to be millions, the use of these IDS tools have become essential part of the security infrastructure and would remain so in the 21st century.

IDSs that are network based watch out for suspicious network traffic and generate alerts based on matching malicious patterns. The rule base is usually kept up to date with the latest threats by security experts in industry and research community. The rule-based type of IDSs has proven to be effective in networks with relatively low bandwidth. However, the false negatives in the presence of unknown or complex attacks with variations in known patterns create loopholes in the network security infrastructure that can potentially lead to entire network disruption.

There are some additional problems. Typically we can find IDs running on high end computing systems or as network appliances; however, in spite of their obvious benefits of detecting malicious traffic, both of these setups have drawbacks. Despite the fast speeds of general purpose processors included on high end computing systems and the relatively recent inclusion of multiple cores with high processing capacity, the operations are executed in a sequential fashion. In addition, these processors often have to dedicate resources to serve operating system calls that run with higher degree of priority, leaving secondary application calls of IDSs to run at lower priority levels - even halted temporarily.

The scenario of using sequential processors does not present a viable and scalable solution in environments where the demand for computational resources are very high, as in the case of network IDS systems tapping into gigabit network lines [3]. If we add to this the fact that future expectations for bandwidth will grow faster than the future expectations of CPU processing powers in the 21st century, we can foresee a grim future for intrusion detection systems or security as a whole, unless we consider more scalable and

intelligent approaches to solve the problems mentioned above.

To enhance processing capabilities, network appliances have been using ASIC (Application Specific Integrated Circuits) as the technology of choice to embed their intelligence. Although this technology can achieve high performance coupled with much higher efficiency as those of general purpose processors, the process of producing a finalized ASIC chip is costly and time consuming. That is why today researchers are focusing on FPGA (Field Programmable Gate Arrays) based technologies as the choice for more flexible platform without sacrificing significantly on costs and performance.

FPGA provide the means for a hardware-based approach capable of high throughput and performance in IDS systems and have several advantages over general purpose CPUs and ASIC technologies, such as low NRE (Non-Recurring Engineering Costs) cost, reconfigurability, parallelism and uninterrupted operation.

The idea of using FPGAs for intrusion detection for network security is not novel; recent solutions have been proposed to improve attack detection [1, 2, 5, 7, 8, 10, 13 and 15]. However, only conventional rule-based systems with exact reasoning have been considered for implementing analytical capability. As we mentioned earlier, rule-based systems with exact reasoning that does not accommodate vagueness/variations in knowledge, lack detection capabilities in the presence of attacks with variable patterns.

As potential solution to the problem mentioned before, we propose to integrate fuzzy logic with FPGA based technology for intrusion detection. Fuzzy Logic, proposed by Lotfi Zadeh [6], works by mimicking the human way of reasoning, where everything is a matter of degree and nothing is absolute, as opposed to the Boolean logic that we commonly experience where everything down to the smallest unit of information is distinguished by a crisp 1 or 0. Fuzzy Logic has been shown to be effective in detecting attacks [9 and 12] but have not been integrated into FPGAs. The use of Fuzzy Logic embedded into FPGA-based IDS would add the benefit of detecting known and unknown attacks with variations, plus the inherited benefits of using FPGAs without sacrificing performance and detection rates.

2. Research Approach

FPGA based fuzzy IDS (FPGA–FIDS) offers a novel application of fuzzy logic integrated into FPGA based technology for intrusion detection with following features:

- Fuzzy Logic
  - Accounting for attacker actions with variations
  - Allowing intuitive reasoning accommodating vagueness in human perception
- FPGA based hardware providing
  - Parallelism
  - Reconfiguration
  - Low NRE costs

The basic framework of the system is shown in figure 1.

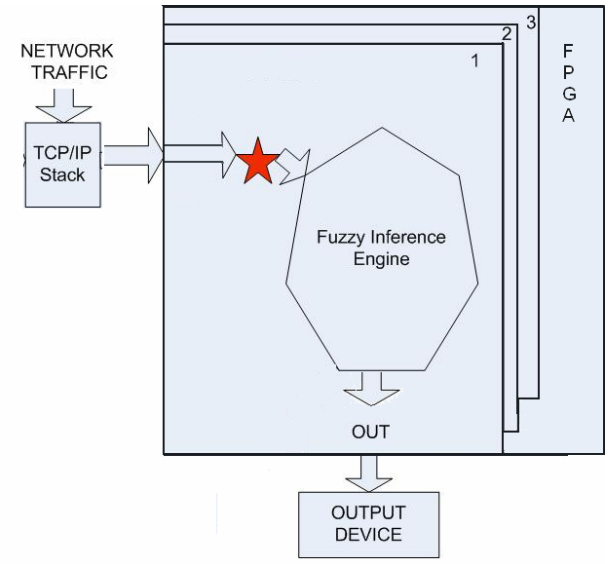


Figure 1. FPGA Based Fuzzy IDS Framework

The core of FPGA –FIDS is the fuzzy inference engine that performs the following functions, which are descriptive of any Fuzzy systems.

- Fuzzification of the Crisp Inputs
- Rule Evaluation
- Aggregation
- Defuzzification and Output

Fuzzification is the process of taking the crisp inputs of the fuzzy input variables, and determining to what degree the inputs map to the defined fuzzy sets. We then apply the fuzzified inputs to the antecedents of the fuzzy rules (e.g. If TCP SYN rate is *High* then SYN Flood attack is *High*). The remaining two steps perform the aggregation, which consists of unifying all the outputs of the rules and combine them into a single fuzzy set; and lastly, the defuzzification step, which takes the fuzzy output of the aggregation and converts it into a crisp output - meaningful to the network security personnel.

To use fuzzy logic, all input and output variables that describe system behavior are needed to be defined. Fuzzy linguistic variables are used as indicators of abnormal network behavior. The numbers of port scans, the total number of packets during a time window are examples of such variables. After selecting fuzzy variables, we narrow down their ranges into several boundaries that conform to fuzzy sets. These fuzzy sets can be defined as *Low*, *Medium* or *High*, according to an expert opinion, based on the metrics that describe these variables.

To support fuzzy set and rule definitions, we analyze network traffic described in the attack datasets provided by MIT Lincoln Labs during their DARPA Intrusion Detection Evaluation experimentation [4] using analytical tools such as SAS Miner [14].

For implementation of FPGA-based technology, we focus on a basic IDS module that would serve as a proof of concept and groundwork for a future more complete and finalized FPGA-FIDS. The FPGA development platform chosen is a fairly common board found in Embedded Systems Labs - the Altera NIOS II Development Kit board based on the Cyclone II EP2C35 device. This platform choice comes with several benefits. The inclusion of an 10/100 Ethernet physical layer/media access control (PHY/MAC) for network communications, an LCD module, 16MB of SDRAM, and from the software development standpoint, the inclusion of MicroC/OS-II real-time operating system (RTOS) and the NicheStack TCP/IP Network Stack, Nios II Edition. These last two items could be used to rapidly prototype an IDS system.

For initial experimentation, SYN Flood and/or Ping Sweep attack is considered for their widely reported use by attackers and for ease of implementation. The fuzzy inference engine of the IDS consists of a simple input, defined by the incoming SYN packet rate, a single output, defined by the output SYN Flood level, several rules and a Sugeno style defuzzification. The system is tested with a simulated SYN Flood attack (300 SYN packets per second). Although preliminary and immature, the results are promising. At variations of SYN rates (few hundred packets/sec), the system responds with outputs that correspond with the adjusted levels of attacks.

With our current general purpose FPGA testing platform (NIOS II Development Kit, Cyclone II Edition), the performance achieved so far requires improvements, which is our most immediate milestone. To achieve this, we are focusing on the inclusion of several parallel processing cores within the FPGA. Other objectives include the inclusion of capabilities to detect various types of malicious traffic or attacks, and the incorporation of better intelligence in the fuzzy inference engine. The inclusion of these features will allow a more in-depth benchmarking test, using among others, the MIT-DARPA 2000 IDS dataset against FPGA-FIDS platform.

3. Conclusions and Future Work

Based on the current status of research in the field of FPGAs toward Network Security and Intrusion Detection, there is no doubt that this technology is opening new ways for improving performance of network security devices and making their development a much easier task. Still this would not make any FPGA-based IDS more effective in detecting possible attacks or malicious traffic. From the security standpoint, there is need to address innate weakness of expert based IDSs which do not accommodate for uncertainty in real world in terms of vagueness or impreciseness of information. Inclusion of artificial intelligence based on fuzzy logic has the potential to help mitigate such problems by adding inexact reasoning capabilities to expert based IDSs.

Acknowledgement

Our sincere gratitude goes to Dr. Omar Elkeelany of Electrical and Computer Engineering Department at TTU for supporting this project by providing us with necessary hardware resources.

References

1. Automated Tools to Implement and Test Internet Systems in reconfigurable Hardware. J. Lockwood, C. Neely, et. al. *ACM SIGCOMM Computer Communications Review*. Volume 33, Number 3. July 2003.
2. Characterizing the Performance of network Intrusion detection sensors. L.Schaelicke, T. Slabach, B/. Moore, and C. Freeland. Proceedings of the *Sixth International Symposium on recent advances in Intrusion Detection (RAID 2003)*, Lecture Notes in Computer Science, Berlin-Heidelberg-New York, September 2003. Springer-Verlag.
3. DARPA Intrusion Detection Evaluation. Intrusion detection systems monitor network state looking for unauthorized usage, denial of service, and anomalous behavior. http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/index.html (current April 21, 2009)
4. Efficient Packet Classification for Network Intrusion Detection using FPGA. H. Song, J. Lockwood. Proceedings of the *International Symposium on Field-Programmable Gate Arrays.* Monterey, California, February 20-22.
5. Engineering News. The world is a matter of degree. EECS professor reflects on his pioneering theory, *Fuzzy logic*. October 31, 2005 Vol. 77, no. 10F.
6. Fash Hash Table Lookup Using Extended Bloom Filter: An Aid to Network Processing. H. Song, J. Lockwood, et.al. *SIGCOMM 2000*, August 21-16.
7. Fast and Scalable Pattern Matching for Network Intrusion Detection Systems. Sarang Dharmapurikar, and John Lockwood. *IEEE Journal on Selected Areas in Communications:* Oct. 2006, Volume 24, Issue 10, pp. 1781-1792.
8. Fuzzy Intrusion Detection. John E. Dickerson et. al. Joint 8th IFSA World Congress and 20th *NAFIPS International Conference Proceedings.* 2001
9. High Throughput Linked-Pattern Matching for Intrusion Detection Systems. Z. K. Baker and V. K. Prasanna. Symposium On Architecture For Networking And Communications Systems. Proceedings of the 2005 *ACM Symposium on Architecture for Networking and Communications Systems*. Pages 193-202. 2005.
10. Memory-Efficient Content Filtering Hardware for High-Speed Intrusion Detection Systems. S. Yi, B. Kim, et. al. *SAC 2007*, March 11-15.
11. MMDS: Multilelvel Monitoring and Detection System. D. Dasgupta, J. Gomez, et. al. In Proc. of the *15th Annual Computer Security Incident Handling* Conf. (FIRST), Canada, June, 2003
12. Rules-based Network Intrusion Detection using a Field Programmable Gate Array. Christopher Hayes and Tatin Singhal. 16.671 Advanced Computer Architecture (Final Project). UMASS Lowell. 2006.
13. SAS. Data Mining Software, SAS Enterprise Miner. 2009. http://www.sas.com/technologies/analytics/datamining/miner(current April 21, 2009)
14. SIFT: Snort Intrusion Filter for TCP. M. Attig and J. Lockwood. Proceedings of the *13th Symposium on High Performance Interconnects*. Pages: 121 – 127. 2005.

Developing Systems for Cyber Situational Awareness*

James Okolica, J. Todd McDonald, Gilbert L. Peterson, Robert F. Mills, and Michael W. Haas
Air Force Institute of Technology and USAF 711 Human Performance Wing
jokolica@afit.edu, jmcdonal@afit.edu, gpeterso@afit.edu, rmills@afit.edu, michael.haas@wpafb.af.mil

Abstract

In both military and commercial settings, the awareness of Cyber attacks and the effect of those attacks on the mission space of an organization has become a targeted information goal for leaders and commanders at all levels. We present in this paper a defining framework to understand situational awareness (SA)—especially as it pertains to the Cyber domain—and propose a methodology for populating the cognitive domain model for this realm based on adversarial knowledge involved with Cyber attacks. We conclude with considerations for developing Cyber SA systems of the future.

1. Introduction

On February 18th, 2001, Robert Hanssen was arrested for selling American secrets to Moscow for a period of 22 years[+].

On April 28th, 2007, distributed denial of service (DDOS) attacks began on media website in Estonia. These DDOS attacks would later spread to attacks on Estonia's critical infrastructure including banks, ministries, and police.

On August 8th, 2008, scant hours after shooting began between Russian and Georgian forces in South Ossetia, cyber attacks began on Georgia's government and bank websites.

The Department of Defense (DoD) NetOps strategic vision states that commanders, users, and operators (at all levels) need accurate and timely information when accessing the global information grid (GiG). Of course, the understanding of the health and mission readiness of the GiG remains vital for this goal to be achieved. At every level of the mission space, we need a coherent framework which translates events that occur in time and space to their (possible) deleterious effects on mission success.

What all of the above incidents have in common is that information was available that might have led to earlier detection and mitigation. Robert Hanssen had a password breaker program on his work computer [1]. Network probes and DDOS attacks were performed on Georgia's critical infrastructure as early as July 20, 2008[*]. What is needed is a means to increase awareness of what is happening in cyberspace—particularly from the viewpoint of attackers and malicious adversaries. What is needed is *Cyber Situational Awareness.*

With the advent of Cyber as a prominent operational concern and even a defined domain of operations in the U.S. Air Force, the DoD as a whole has come to realize that Cyber-based effects and defensive operations are integral to the overall success of air, land, naval, and space operations. Industry has also realized that vulnerabilities in this realm, including targeted

*The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

[+]John Markoff, Aug 12, 2008 NY Times, "Before the Gunfire, Cyberattacks", http://www.nytimes.com/2008/08/13/technology/13cyber.html

malicious attacks, have huge monetary consequences and carry losses in both productivity and public trust.

In this article we offer a definition for situational awareness for the Cyber domain and present an overview of the problem space within which it resides. We show how traditional definitions of SA may be adapted for Cyber specifically in a sense/evaluate/assess loop which provides correlation between real events, key system components, and their corresponding business/mission impact. We propose a notion of the adversarial narrative, which provides a ground truth view of SA which knowledge and data discovery techniques ultimately attempt to replicate and refine. We also propose a methodology for building an automated discovery engine that can build a useful, actionable Cyber SA picture for commanders at various levels.

2. Defining Cyber Situational Awareness

While there are several definitions of what is meant by situational awareness, one of the most accepted is by Dr. Mica Endsley [2]. It defines SA as "the **perception** of elements in the environment within a volume of time and space, the **comprehension** of their meaning, and the **projection** of their status in the near future". Endsley then extends his concept of SA to include a memory component and a decision/ action taken as a result of the SA. The decision / action is then considered to act upon the environment which produces a circular loop as SA begins again with a perception of the new environment (Figure 1).

Using Endsley's definition, there are three functions *any* SA system must perform: (1) it must sense its environment, (2) it must take its raw sense data and assemble it into a meaningful understanding of its environment, and (3) it must use its current understanding to predict the future. Figure 2 provides a specific Cyber example based on an attacker with inside knowledge and access to an organization (an insider threat).

First, the SA system senses elements of an individual's environment. Using an insider threat example, these sensors include emails sent and received by the individual as well as transaction logs from the applications the individual uses for his day-to-day activities. The SA system then assembles this information into a concept which matches its already known concept of "insider threat". At this point, the SA system has a suspicion that the individual might constitute an insider threat. The SA system than predicts that if the individual is an insider, he may (1) send information to computers outside of the local network and (2) possess password cracker programs. The SA system then decides to activate packet traffic and file locator sensors to determine if it is correct. When the results are positive, the SA system then combines the packet traffic and firewall information to determine what data vulnerabilities exist. The concept observed during this second pass is "data exfiltration." However, the SA system still only understands this concept in terms of data.

The final step is to incorporate an understanding of the relationship between business processes and data elements to determine the mission impact of the projected data exfiltration. It is this final step that is missing from many of the Cyber SA efforts to date. High level business processes must be broken down into detailed workflow steps performed by individuals within different organizations. Users and applications must then be associated with each functional responsibility and action respectively within each of the workflows. Once this association has been made, it is possible to relate data concepts to operational concepts. Then, when sensors extract user and application data and feed correlation tools that assemble it into a

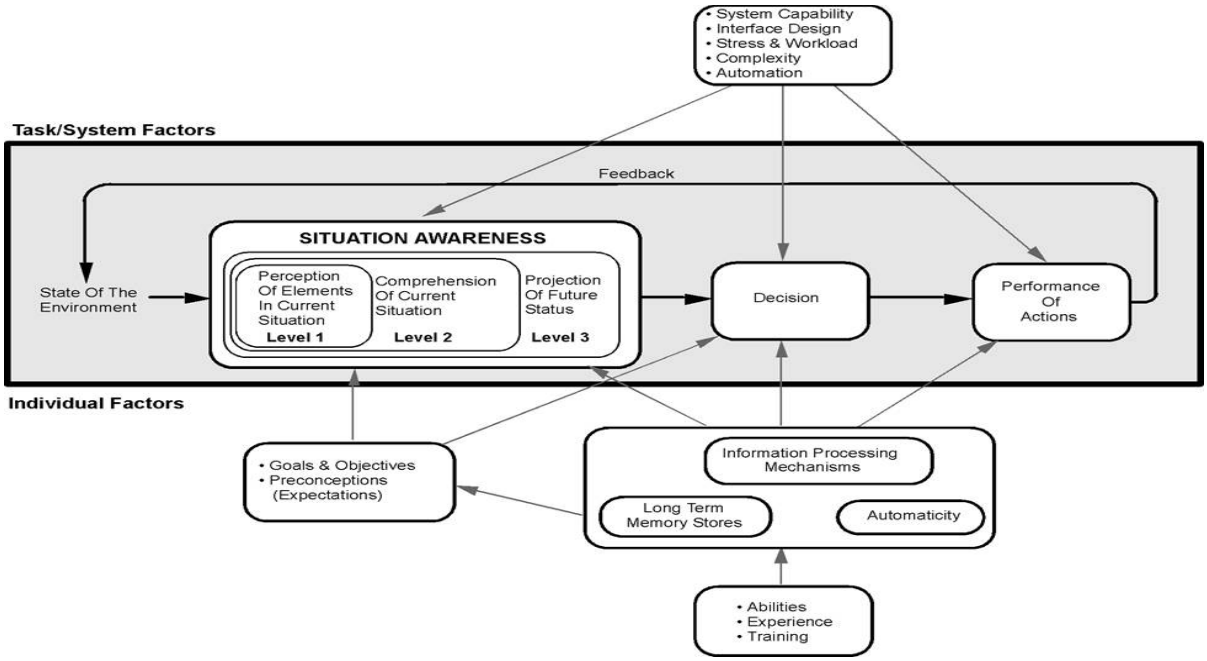


Fig. 1. Endsley's situational awareness model [2]. SA leads to decisions and actions which affect the environment itself. SA captures the environment state through perception, comprehension, and project (predictive analysis), forming a loop.

comprehensible picture of the data environment, business health assessment tools can then translate the data environment into an operational environment. This complete process (Figure 3) then provides a holistic Cyber Situational Awareness.

Before proceeding, Endsley's term *comprehension* needs to be better framed. Specifically, we need a distinction between local comprehension and global comprehension. If it is possible for a single host to determine a concept, e.g., "I am under a DDoS attack", then we define that knowledge as a local concept. If the only way to determine a concept is to collect information from several hosts, e.g., "a worm is spreading across the network", then we define that knowledge as a non-local concept. For instance, a domain name server being singled out for a DDoS attack is a local concept.

Now, consider a non-Cyber example of this distinction. When a homeowner considers his water system, he thinks about the individual pipes, which rooms have faucets and whether the toilets are working. He also may give some thought to the water entering and leaving his home. However, when a city engineer considers his water system, the only parts of an individual's home that the engineer thinks about is the water entering and leaving a home. Not only doesn't the engineer care about the specific conditions in an individual home, he may not even use the same vocabulary, .e.g. faucets and toilets. This implies that the vocabulary used to describe local perceptions may not be needed to describe global perceptions. Furthermore, the vocabulary used to describe global data environment perceptions may not be used to describe global operational perceptions. At each level, the transformation from perception to comprehension changes the language used to describe the environment (Figure 4).

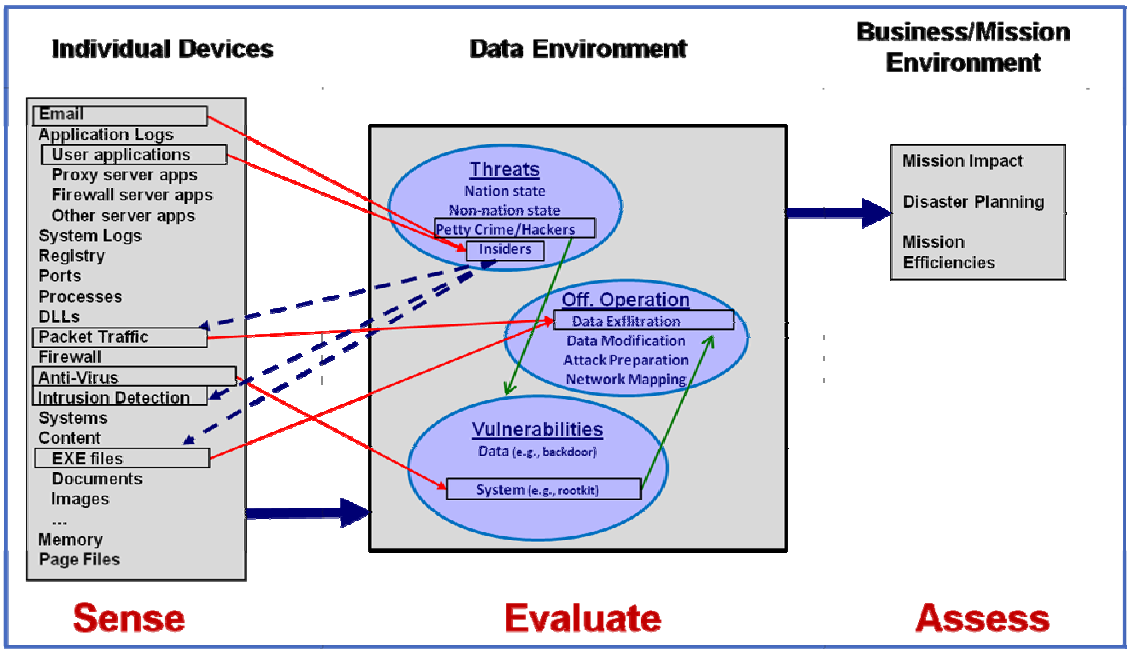


Fig. 2. Insider threat Cyber SA example. Sensors at lower levels on individual devices focus on specific information/data elements. Evaluation matches activities and patterns of data to known threat categories which spawn additional sensor / data collection activities. Determination of particular offensive operations and associated vulnerabilities that support the operations are distilled. The health of the overall mission and plans that mitigate effects of the projected evaluation are assessed.

3. Defining the Cyber SA Problem Space

Developing an infrastructure that provides operational cyberspace situational awareness requires successfully solving multiple problems. As Figure 5 illustrates, in addition to developing sensors (problem 1), correlation tools (problem 2) and visualization tools (problem 3), there are several embedded issues that must be resolved.

First, detecting a non-local (i.e., distributed) attack requires correlating information from multiple types of multiple sensors. For instance, there are sensors that track network traffic on a specific host and there are sensors that track program executions on a specific host. Only by combining the information from both types of sensors across multiple hosts can a "low-and-slow" attack be detected. At the heart of this issue is the need to evaluate information from multiple types of sensors that both view and describe the network environment in different ways. Developing an infrastructure for describing information from disparate sources in a unified way is defined as the environment description language (EDL) problem (Figure 5-P4). One subset of the EDL problem is describing data information that is either: (a) local to the Host (i.e. Host Data EDL (HDEDL) problem) or (b) descriptive of the entire network (i.e., Network Data EDL (NDEDL) problem).

Second, in addition to minimizing sensors' processor time on each individual host, correlating multiple sensors across hosts requires minimizing network traffic between hosts. If all sensor information from each host is transmitted across the network, the result would be a self-inflicted denial of service attack. Instead, some sensor fusion at the local (i.e. individual host) level needs to occur before transmitting a more abstracted state to other hosts. Determining methods for summarizing local data and transmitting it efficiently is defined as the scalability problem (Figure 5-P5).

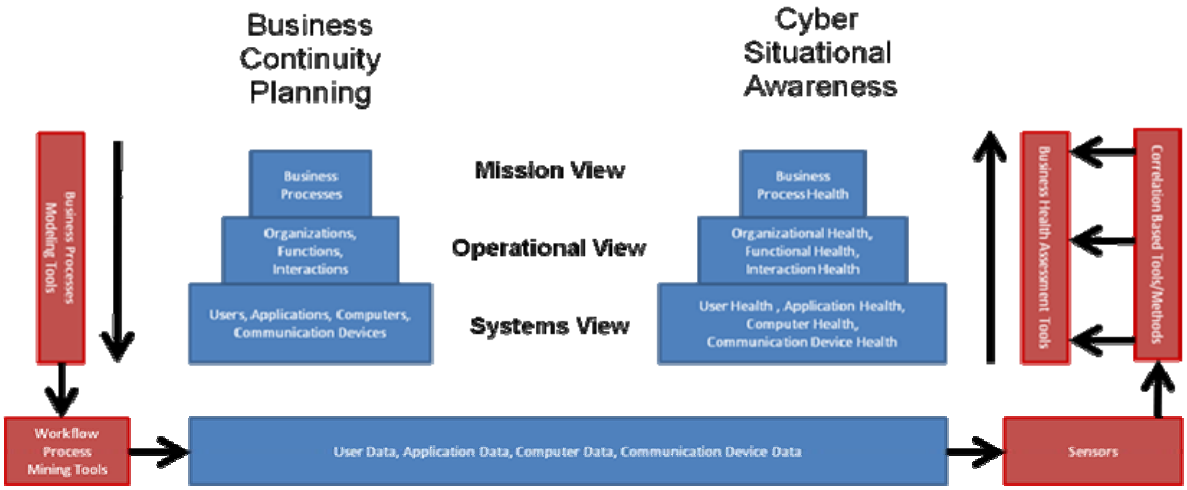


Fig. 3. Cyber situational awareness (SA) model. Business continuity planning (BCP) based on workflow processes and models allow top-down mapping of mission, operational, and systems functions/organizations/equipment to the overall business goals and activities. Data at various levels capture both BCP and Cyber SA data. Sensors and correlation tools provide bottom-up knowledge synthesis, filtering and fusing data to provide top-level business process health.

Third, an issue that emerges naturally from the first and second issues is identifying *what* to look at. Time and again, in the wake of an attack (cyber or otherwise), signs are uncovered that if they had been noticed and acted upon in a timely manner would have prevented the attack. Security professionals are left with the uncomfortable task of answering why they hadn't been looking for that particular sign. Unfortunately, the reality is that it is impossible, even in a cyber environment, to look at and evaluate everything. Instead, security personnel must select a subset of the data to collect and analyze. Determining what to look at is defined as the feature extraction problem (Figure 5-P6).

Our fourth concern deals with single points of failure. If correlation occurs in a central location and the adversary is able to neutralize that target, the security of the network is significantly degraded. In addition, if an adversary is able to subvert a host and cause it to send out erroneous sensor information, the security of the network will also be compromised. Addressing these twin issues of single point of failure and sensor corruption is defined as the resiliency problem (Figure 5-P7).

Finally, once we address these issues, it becomes possible to develop correlation tools to determine when the network, or its hosts, is/are under attack and what the implications of this attack are to the health of the data network.

While the four embedded problems listed above address determining whether the network is under attack, the issue still remains whether we can adequately communicate this information to security professions and senior management. While the problem of visualization is more of a human effects issue than a technological one, unless this problem is solved, efforts on the problems above are wasted. Related to the Visualization problem, is the "so what?" factor. While a good visualization tool can provide a Chief Information Officer with the relative health of her network, it does not address the Chief Operations Officer's (COO) question of "can the operation fulfill its mission?"

To answer this question, network health must be translated into business process health. In the same way that the data EDLs addresses disparate types of sensor information, an operational environment description language (OEDL)

would allow business process engineers to describe the relationship between the data environment and the operational environment. Thus the EDL problem has three sub-problems: HDEDL, NDEDL, and OEDL (seen in Figure 4). With this information, visualization tools can be developed to provide the COO with the answers to her questions.

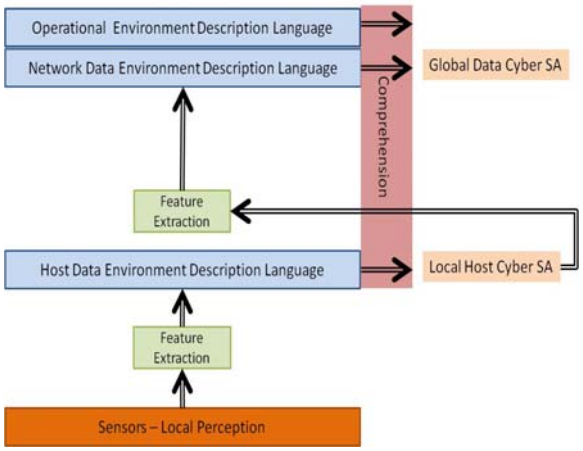


Fig. 4. The Cyber SA environment. Environment description languages exist at three different levels, providing both local and global SA comprehension and expression of Cyber SA.

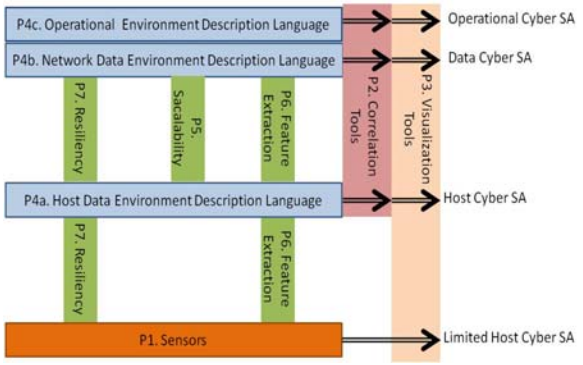


Fig. 5. The Cyber SA problem space. Six different problem areas are delineated, capturing the primary research space for accomplishing successful Cyber SA.

While IT specialists think of visualization tools as red light/green light monitors, this awareness represents only one type of visualization. Another involves providing a narrative description of the offensive operations being perpetrated on the organization.

Consider the following example: several network sensors identify that Bob's machine has a rootkit on it. The tools further identify what the rootkit is trying to hide. What senior management wants to know is who installed the rootkit and for what purpose. A successful Cyber SA monitoring system might provide senior management with parts of the real-life story that involves Mallory, the employ who actually perpetrated several malicious actions that led to the rootkit installation and operation. Though the true, real-life narrative of the events would detail the underlying social, political, or personal motivations (i.e., Mallory targeted Bob out of personal vendetta related to a work-place affair), the Cyber SA narrative would determine that Mallory used social engineering.

The awareness would include pertinent pre-exploitation details such as the fact that Mallory sent Bob an email with a link to a website that has a cross site scripting (CSS) vulnerability (which he clicked on), subsequently giving Mallory administrator privileges on his machine. She then used those privileges to install a rootkit and a backdoor so that she could access his machine. She started small by changing his Outlook schedule and removing important meetings; however, she quickly moved on to sending emails (and trying to remove the evidence so Bob wouldn't notice) to other employees with racial and sexist jokes in order to have him fired for inappropriate conduct – she was extremely vindictive in her dismissal. Although this sort of narrative may be considered science fiction today, by defining the vocabulary and languages and developing the appropriate sensors and correlation tools, this sort of visualization may be commonplace in the business world of tomorrow.

Once all of the problems described in the problem space are addressed, there is still the

issue of obtaining realistic data in order to test the Cyber SA systems and build confidence that projection accurately characterizes threats, offensive activities, and vulnerabilities.

4. Developing a Cyber SA System

We believe three overlapping activities are needed to develop a Cyber SA system: (1) developing a test environment that provides sensor data that can be correlated and fused, (2) developing one or more languages that can describe the cyber environment at different levels of abstraction, and (3) integrating the adversarial narrative into the abstraction space.

4.1. Developing Cyber SA System Test Environment

The purpose of a Cyber Situational Awareness system is to report on the health of an operational network. Therefore, an ideal dataset would provide data that duplicates an operational network. Some of the desirable characteristics include:

(1) "Real" data including normal baseline traffic and attempted/successful malicious attacks. While the percentage of normal to malicious data may be modified to provide sufficient exemplar data, the percentages should be explicitly stated so that a realistic baseline can be defined.

(2) "Timely" data from a time period long enough to model all activity expected on the operational network. This includes (a) peak usage data as well as off-peak (e.g., nighttime and weekend) data; (b) end of month/quarter/year usage data as well as day-to-day usage data;

(3) "Functional" data of many different types of users including technical, clerical, operational, and management users.

(4) "Scaled" data for an operational network of appropriate size. While this varies depending on where the operational Cyber SA system is intended, it is likely that the network data should include data from several hundred, if not several thousand, hosts.

(5) "Heterogeneous" data that covers all of the possible inputs that an IDS might desire. While it is impossible to enumerate all possible inputs, representative data includes network traffic, operating system logs, application transaction data, and temporal operating system process data.

Unfortunately there is currently no publicly available dataset that satisfies all of these requirements. However, there are several datasets available that satisfy some of them. In 1998, MIT Lincoln Laboratories under Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratories developed the first dataset for evaluating intrusion detection systems [3]. They added to this dataset with additional datasets in 1999 and 2000 [4]. Although there have been several criticisms of the representativeness of the data [5], they still remain one of the most used datasets.

While DARPA has since sponsored a 2002 Cyber Panel Correlation Technology Validation effort, the datasets used are no longer publically available. Instead, there are several datasets from other competitions that have been made available for public use. For instance, DEFCON is an annual convention for security professional and hackers. One of the principal events at DEFCON is its 72 hour Capture the Flag (CtF) contest where teams attempt to protect their own network while invading other teams (thus capturing their flag). The event traffic from DEFCON 8 CtF and DEFCON 10 CtF was recorded and made available by the Shmoo Group at http://cctf.shmoo.com/. Lastly, the 3rd International Knowledge Discovery and Data Mining Tools Competition focused on network intrusion and it has made its dataset available as well [6]. Unfortunately, what all of these

datasets have in common is a lack of a baseline. While the DEFCON and KDD Cup data are real data, they were developed in an artificial contest environment and consequently contain unrealistic amounts of attack data with little or no baseline data.

Recognizing the issues inherent in synthesized IDS data, several organizations have developed testbeds as more realistic environments for measuring the success of intrusion detection systems. We describe four such environments which have representative features consistent with Cyber SA and development and sensor data analysis.

Originally built from Utah's EMULAB software, the cyber-DEfense Technology Experimental Research (DETER) testbed has been configured to "provide stronger assurances for isolation and containment" [7]. Its goal is to specifically test network defense against attacks including distributed denial of service attacks, worms and viruses. DETER was developed to provide a medium-scale (approximately 300 nodes in two clusters) environment for "safe, repeatable, security-related experimentation to validate theory and simulation". It is run by Information Sciences Institute, University of California at Berkeley funded by the National Science Foundation and the Department of Homeland Security. More information can be found at http://www.isi.deterlab.net/.

Netbed, also a descendant of EMULAB, is "a software system that provides a time- and space-shared platform for research education, or development in distributed systems and networks" [8]. It uses both local, dedicated nodes, geographically-distributed shared nodes and emulated Dummynet nodes. Researchers access these nodes via a virtual topology which causes Netbed to configure a physical topology. Netbed provides an experimentation facility that integrates these approaches, allowing researchers to configure and access networks composed of emulated, simulated, and wide-area nodes and links. Netbed's primary goals are "ease of use, control, and realism, achieved through consistent use of virtualization and abstraction". Netbed is run by The Flux Group, School of Computing, University of Utah. More information can be found at http://www.emulan.net/.

The Protected Repository for the Defense of Infrastructure against Cyber Threats (PREDICT) is a repository for current computer and network operational data accessible through a secure web-based portal and is made available to qualified cyber defense researchers located in the United States [9]. It is run By RTI International, a not-for-profit research institute funded by the Department of Homeland Security. More information can be found at https://www.predict.org/.

Finally, System Administrator Simulation Trainer (SAST) is a software simulator which artificially generates internet/network traffic and superimposes actual exploits on it. SAST provides a safe simulator for DoD security and personnel and system administrators to hone their capabilities by providing thousands of real world exploits and an environment that can mimic an organization's information infrastructure. It is run by the National Center for Advanced Security Systems Research. More information can be found at http://www.ncassr.org/project/ .

4.2. Describing the Cyber Environment

Language is "a systematic means of communicating by the use of sounds or conventional symbols" [10]. It must, at a minimum, contain names of items (e.g. John, George, Andrew, hit, smack, beat) and may also contain classifications of items (e.g., person, president, attack, and strike). Additionally, adding grammar enables communication of relationship between items (e.g., without a grammar {George beat Bill}, {Bill beat George} and {Bill George beat} are equivalent). As a

result, language is generally considered to be composed of vocabulary (possibly containing classifiers) and the elements to manipulate them.

In order to (1) describe data that a Cyber SA system senses and (2) fuse that data into comprehensible concepts, a Cyber SA system requires a language. Relevant vocabulary may include (1) devices connected to a network, (2), users of the network, (3) application software run on the network (4) user missions/operations enabled by the network, (5) actions performed by devices, users, and applications (6) communications between devices, users, and applications, (7) actions performed on devices, users and applications.

There are two distinct ways of communicating relationships. The first, and most obvious, is via grammar (e.g., "George beat Bill"). The second defines vocabulary such that a single item contains this information (e.g., attack(source=George, target=Bill, time=12-Jan-09;21:23:00, method=stick)). There are benefits to each technique. Formal deductive methods, e.g., predicate logic, benefit greatly from the explicit relationships between objects that grammars provide. On the other hand, since the formalization of relationships limits the expressiveness of language, knowledge from data discovery can benefits from the lack of grammars, allowing for unconsidered relationships to emerge.

Two primary application areas that are related to Cyber SA are intrusion detection and cyber forensics. While the authors know of no Cyber SA-specific language, there are several languages related to intrusion detection and cyber forensics that apply. The Intrusion Detection Message Exchange Format (IDMEF) was developed by an Internet Engineering Task Force (IETF) working group and sent out with a Request for Comments (RFC) in March, 2007. IDMEF uses extensible markup language (XML) to facilitate the multitude of sensor vendors. It provides for sensor input from network devices (e.g., switches and routers), O/S audit logs, and application transaction logs as well as alerts to be sent back to operators and actions to be taken in response to sensor input.

The IDMEF data model (RFC4765) shown in Figure 6 is an object-oriented representation of a space which includes source data with very little information (e.g., origin, destination, time, and name/description) and source data with too much information (e.g., application transaction logs with hundreds of fields in them). The IDMEF-Message entity is the top level class. All other entities are sub-classes of it. Currently the two subclasses of IDMEF-Messages are alerts and heartbeats. Alerts correspond to analyzer (i.e., sensor) alerts or events and occur asynchronously. There are several sub-classes within the alert class including tool alerts (to describe attack tools), correlation alerts (to describe previously grouped and correlated alerts), and overflow alerts (to describe buffer overflow attacks). The heartbeat class defines messages sent out at regular intervals from analyzers to managers (centralized tools used by operators to configure sensors, analyzers, data consolidators, etc.). Lastly, the object-oriented representation provides both flexibility and extensibility.

While the IDMEF model requires the implementer to define the relationships between classes, Pinkston *et al.* [11] have developed ontology, shown in Figure 7, which defines both the classes and the relationships between them. Although as described, TCO focuses on network attacks but might be easily extended to incorporate exfiltration or modification of host data. Furthermore, despite the fact that TCO cannot describe distributed attacks affecting multiple hosts, it can detect them through the use of generic queries.

In addition to IDMEF and TCO, the National Center for Forensic Science and the University of Central Florida Department Of Engineering Technology have proposed the digital evidence

markup language (DEML) as a method to model digital evidence [12]. Unlike IDEF and TCO, DEML is more focused on characteristics of a specific device, e.g., hard disk model, partition size, O/S revision and uptime, etc. While DEML may not be expressive enough to be used to describe a large scale network-wide environment, its specificity makes it's a good choice for describing a detailed host-level environment.

Although not specifically a language, MITRE has compiled the common vulnerabilities and exposures (CVE) list [13] to provide standardized names for different attacks and vulnerabilities. CVE has since received widespread adoption by a number of organizations and individuals.

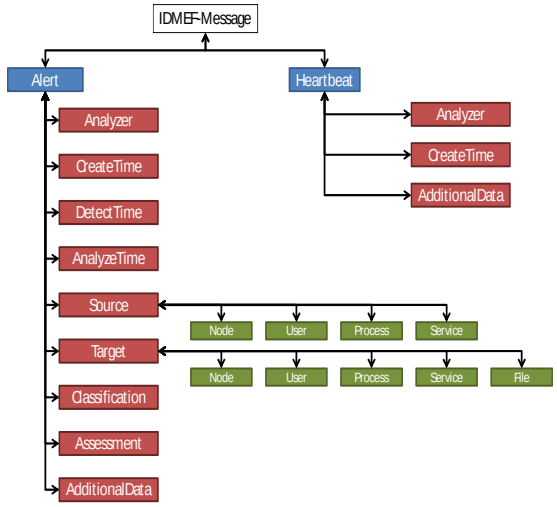


Fig. 6. IDMEF data model. Alerts and heartbeats define all sub-classes of IDMEF messages, covering both asynchronous and continuous monitoring data.

4.3. Measuring Aggressor Cyber SA

A crucial final element needing integration into Cyber SA systems is the ability to accurately describe or measure what is actually happening in reality. We consider that for the most basic of Cyber SA questions (whether a Cyber attack is underway, imminent, or in preparation stages), only the attacker possesses ground truth situational awareness and only the attacker can define the ground truth narrative which describes who, what, why, when, and where. Unless an attacker acts for no reason at all (purely psychopathic motivations), the underlying reasons and goals of an attack can help us identify patterns of behavior. Likewise, the actual steps taken in a malicious attack are known by the attacker perfectly, though execution of them may not be perfect. This perspective helps shape the way we design and test systems for Cyber SA.

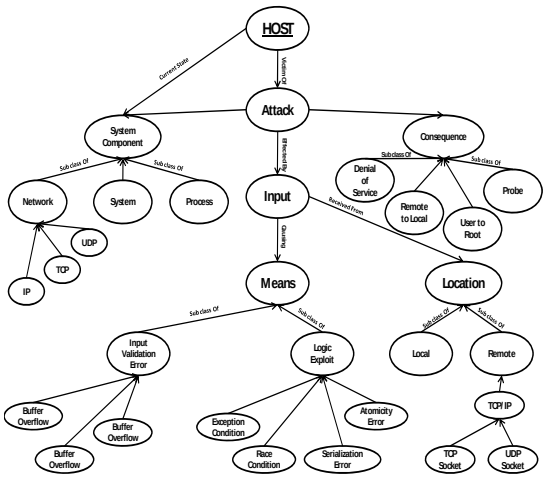


Fig. 7. Target Centric Ontology (TCO).

One way to describe Cyber SA then is how close assessment may come to the attacker's ground truth SA. Successful detection, identification, and differentiation of various malicious activities may be compared only rightly to the actual activities. Our methodology for resolving this question also forms a basis for refining a domain model that supports information fusion from bottom data/correlation tools to high-level Cyber SA abstractions (using environment descriptions and ontology). We envision test environments that involve use of real-world attacks (ARP cache poisoning, data exfiltration, social engineering, malware

deployment, etc.) executed in the backdrop of configured sensors and data correlation tools. Such attacks give the bottom-layer data elements which may be fed to correlation tools and engines.

What prevents accurate, high-level Cyber SA in many cases is not knowing which data elements to look for and which data elements to keep. It is those missing data elements and correlation hints that prevent the high-level picture from being adequately created. By executing known attacks in an iterative manner, we expect that candidate domain models may be refined that capture a "middle" layer of knowledge conducive for populating our high level SA expressions. Our current research efforts focus on developing this middle layer of domain ontology and finding appropriate fusion algorithms with favorable predictive behaviors.

5. Conclusion and Future Work

While the above steps bound the work of developing Cyber SA systems, we expect continued progress by researchers in the problem space areas will help candidate systems mature over the next decade. The co-problem of adequately defining the business mission space remains an open problem with a different and active research community. Without this fuller context of how Cyber may affect business process health and lower levels of correlation, Cyber SA systems may not find prominence in operational use. Our future work aims at developing adequate intermediary domain models that facilitate generalized fusion of lower-level correlation data with higher level SA statements..

6. Acknowledgement

This material is based upon work supported in part by the U.S. Air Force Office of Scientific Research under grant number F1ATA09048G001.

7. References

[1] Wise, David (2003). *Spy: The Inside Story of How the FBI's Robert Hanssen Betrayed America*, Random House Publishers, ISBN 0375758941.

[2] Endsley, Mica (1995). "Toward a theory of situation awareness in dynamic systems". *Human Factors* 37(1), 32-64.

[3] Lippmann, R., Fried, D., Graf, I., Haines, J., Kristopher, J., *et al.* (2000). "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation," *DARPA Information Survivability Conference & Exposition - Vol 2*., pp.1012.

[4] Haines, J., Rossey, L., Lippman, R., and Cunningham, R. (2001). "Extending the 1999 Evaluation", In the *Proceedings of DISCEX 2001*, June 11-12, Anaheim, CA. Datasets available at http://www.ll.mit.edu /mission/communications/ist/corpora/ideval/data/index .html.

[5] McHugh, J. (2000). "Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory." *ACM Trans. Information System Security* 3(4), 262-294.

[6] Online: http://kdd.ics.uci.edu/ databases/ kddcup99/kddcup99. html.

[7] Benzel, T., Braden, R., Kim, D., Joseph, A., Neuman, C., Ostrenga, R., Schwab, S., and Sklower, K. (2007). "Design, Deployment, and Use of the DETER Testbed". In *Proceedings of the DETER Community Workshop on Cyber Security Experimentation and Test*, August 2007.

[8] White, B., Lepreau, J., Stoller, L., Ricci, R., Guruprasad, S., *et al.* (2002). "An Integrated Experimental Environment for Distributed Systems and Networks". *Proceedings of the Fifth Symposium on Operating System Design and Implementation*, Dec 2002, 255 - 270.

[9] Online: https//www.predict.org/Portals/o0/files/ Documentation/MANUAL%20OF%20OPERATIONS /PREDICT_Overview_final.pdf.

[10] Available online: wordnetweb.princeton.edu.

[11] Undercoffer, J., Pinkston, J., Joshi, A., Finin, T. (2003). "Target-Centric Ontology for Intrusion Detection," *IJCAI Workshop on Ontologies and Distributed Systems (IJCAI'03)*, August, 2003.

[12] Online at: http://www.ncfs.org/digital_evd.html.

[13] Online: http://www.cve.mitre.org/cve/cve.html.

Protecting Reprogrammable Hardware with Polymorphic Circuit Variation*

J. Todd McDonald
Air Force Institute of Technology
jmcdonal@afit.edu

Yong C. Kim
Air Force Institute of Technology
ykim@afit.edu

Michael R. Grimaila
Air Force Institute of Technology
mgrimail@afit.edu

Abstract

Cyperspace is constantly threatened by attackers and malware that focus their attacks on a set of known vulnerabilities. When a sequence of software code or hardware structure is exposed, it can reveal new vulnerabilities and weaken embedded protections. Attacks on existing code sequences or hardware structure will be less effective if we can provide sufficient protection. Though software protection is an open problem with known theoretical limits, practitioners seek to find ways of expressing time or cost metrics induced by various techniques on malicious reverse engineers and adversarial analysis. In this paper we consider the nature of circuit transformation algorithms that operate on programmatic logic using iterative sequences of probabilistic and deterministic transforms. We consider such algorithms from the perspective of the kinds of information relative to circuits we are interested in hiding or protecting and experimental results along those lines.

1 Introduction

One approach to protecting software or circuits from reverse engineering is *obfuscation*: obscuring programmatic logic or original source code information so that an adversary may not subvert, copy, or understand some original version [4]. We observe that general programs typically have collections of straight-line logic (no loops and discrete input/output relationships) and basic programs are themselves abstractions of Boolean primitives [8]. Accordingly, we may represent an interesting class of programmatic syntax as Boolean logic circuits. We also note that reprogrammable hardware environments such as Field Programmable Gate Arrays offer possibility for software-like configuration in a wide variety of modern embedded systems. This provides great context for the Cyber realm and gives us motivation to understand the limits of circuit variation because more and more cryptographic operations and critical technology now find their way into reprogrammable environments.

Leveraging this correlation, we present in this paper an experimental environment that gives insight into the fundamental nature of whitebox variation where functional semantics of a circuit are preserved. Namely, at what point does a polymorphic circuit[1] variant exhibit a hiding property of interest, or obfuscation? We consider this question by analyzing the effect of systematic and iterative changes (variation) to small parts of a circuit

*The views expressed in this article are those of the authors and do not reflect the official policy or position of the Unites States Air Force, Department of Defense, or the U.S. Government

[1] Other established definitions of polymorphism in virology refer to multifunctional circuits that perform two or more functions under different conditions. We use the term polymorphism to highlight the fact that functionally equivalent circuits have *many* (poly) different *forms* or *kinds* (morph) that are all semantically interchangeable.

where we allow large variability within the design of specific experiments. Such experiments allow us to introduce large numbers of user-driven goals, random/probabilistic choices, and criteria-based deterministic options. Thus, we can consider end-to-end effects of small syntactic level changes that manifest not only as whitebox structural variations, but possibly protection metrics of interest.

2 Background

As a measure of security, circuit obfuscation has theoretical boundaries if we desire to prevent *all* leakage in the information theoretic sense [1] or if we want to obtain a best possible alternative [6]. However, if we allow transformations that change blackbox behavior but use a recovery function to return the intended output, other possibilities exist. If we have small input-size functions, we can combine canonical minimization and encryption function composition to fully hide the intent of intermediate gate logic [12]; likewise, if we have circuits with behavior that falls into special classes such as rational functions, we may use homomorphic transformation schemes to provide the hiding [14]. If we limit our measurement scope to specific properties such as side-channel analysis [9, 20, 15] or topology hiding [19], several heuristic and theoretical models come into view as well.

Cohen [3] was one of the first researchers to link Shannon's concepts of confusion and diffusion with programmatic transformations. Most modern obfuscation algorithms use one or more of the program evolution techniques suggested by him: equivalent instruction sequences, instruction reordering, variable substitution, jump addition/removal, call addition/removal, garbage insertion, program encoding, redundancy, program interleaving, and anti-debugger mutations.

More recently, researchers have appealed to formal software models to express certain properties related to obfuscation. Term rewriting systems [2, 16], abstract interpretation [5, 13], and program encryption [17, 11] have all been used to analyze and characterize the effect of structural variation and syntactic changes. These frameworks may either characterize the difficulty of finding and normalizing malicious transformations or attempt to measure the strength of friendly protection schemes based on variation.

Table 1: RPM Notations

Variable	Meaning
C	A combinational Boolean circuit
C'_i	Original circuit C after i iterations of randomization
C', C'_n	Original circuit C after n-iteration randomization is finished
Ω	circuit *basis*. Ω is a set of Boolean functions such that $\Omega \subseteq$ {AND, NAND, OR, NOR, XOR, XNOR, NOT}
$C_{X\text{-}Y\text{-}S\text{-}\Omega}$	the *class* of a circuit, indicating inputs (X), outputs (Y), size (S = maximum number of gates), and basis (Ω)
δ, $\delta_{X\text{-}Y\text{-}S\text{-}\Omega}$	circuit *family*, i.e., the set containing all circuits $C_{X\text{-}Y\text{-}S\text{-}\Omega}$
δ_C	family of circuits semantically equivalent to C ($\delta_C \subset \delta$)

The hardness of reverse engineering or its suitability to hide some original program information is normally linked with *unintelligibility* or *understandability*. The use of these terms has unfortunately not promoted robust theoretical discussion of actual/practical obfuscating transformations because intelligence and understandability remain human-centered concepts. Collberg and his colleagues [4] use metrics that in almost all cases correlate larger size and numbers of artifacts to the specific cost in time or resources of various software reverse engineering tasks. We prefer the ability to measure *indistinguishability* and *randomness* as more precise since both terms have context in traditional cryptography and information theory. In order to understand the fundamental/mathematical nature of heuristic-based syntactic transformations, our experimental environment considers the effects of large numbers and large classes of random choices applied to the structural level of a circuit. As a motivating context, we probe assertions of one obfuscation definition known as the *random program model* [17], which we review next.

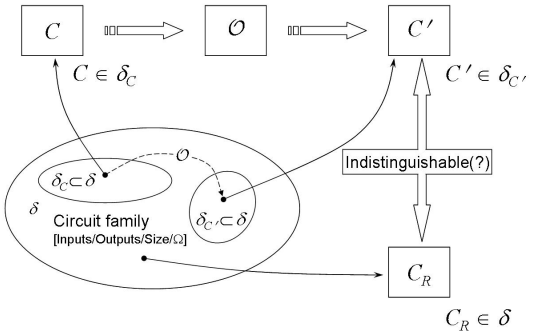


Figure 1: Random Program Model (RPM) [17]

2.1 Notation

In our context, we model programs specifically as Boolean circuits. A circuit over Ω is a directed acyclic graph (DAG) having either nodes mapping to functions in Ω (referred to as *gates*) or having nodes with in-degree 0 being termed *inputs*. We also distinguish one (or more) intermediate nodes as *outputs*. The basis is complete if and only if all functions f are computable by a circuit over Ω. The basis sets {AND, OR, NOT}, {AND, NOT}, {OR, NOT}, {NAND}, and {NOR} are all known to be complete. One example of a complete 6-gate basis is Ω = {AND, OR, NOR, NAND, XOR, NXOR} which has basis size $|\Omega| = 6$. We summarize our notational style in Table 1.

2.2 Randomness as an Obfuscation Metric

When considering circuits, we typically use two primary analysis paradigms to describe them: how they *behave* and how they are *constructed*. We rightly consider software "behavior" as the blackbox functional characteristics (denotational semantics) of a circuit reflected by all possible input/output pairs while we can define circuit "construction" as the representation of its whitebox internal structure (the collection of language statements that define its topography).

We may define an obfuscating transformation $O(\cdot)$ as an efficient, terminating program which takes a circuit C as input and returns another circuit C': $O(C) = C'$. Of this assertion, all theoreticians and practitioners (that we are aware of) would agree. Beyond that, the majority of theoretical and practical models for obfuscation have at least two other requirements for the obfuscating program $O(\cdot)$, where $O(C) = C'$: *semantic equivalence* and *security*.

- **Semantic Equivalence**: $\forall x \in \{0,1\}^n : C(x) = C'(x)$, where n is the input size of C and $C' = O(C)$.
- **Efficiency**: There is a polynomial l such that for every circuit C, $|O(C)| \leq l(|C|)$.
- **Security**: A property that expresses some notion of information "hiding" or security guaranteed by $O(\cdot)$ for every possible circuit under consideration. The expression and measurement of the property varies from model to model: black-box [1], indistinguishability [1], best-possible [6].

In [18, 17], a theoretical and practical understanding of obfuscation based on the *random program model* (RPM) is given. RPM posits that an *intent-protected* circuit when compared with any other circuit randomly chosen from a similar family (i.e., the same $\delta_{X\text{-}Y\text{-}S\text{-}\Omega}$, where $C \in \delta_{X\text{-}Y\text{-}S\text{-}\Omega}$) are indistinguishable as possible variants of the original circuit. Figure 1 gives our visual understanding of RPM. Intent protection itself is expressed as adversarial software exploitation for three main purposes:

1. Tampering with code in order to get specific results
2. Manipulating input in order to get specific results
3. Correlating input/output with environmental context

Compared to other theoretical understandings, RPM differs in the requirement for semantic equivalence and its definition of security. For its security property, RPM posits that if 1) the behavioral (blackbox) information gleaned from the obfuscated circuit C' has no correlation with the original circuit's behavior and 2)

the structural (whitebox) topology of C' has no *more* correlation with the original circuit than any randomly chosen circuit of similar kind, then the intent of the original circuit has been protected. RPM also allows a different input/ouput semantics in the obfuscated circuit, as long as the intended, original output is recoverable.

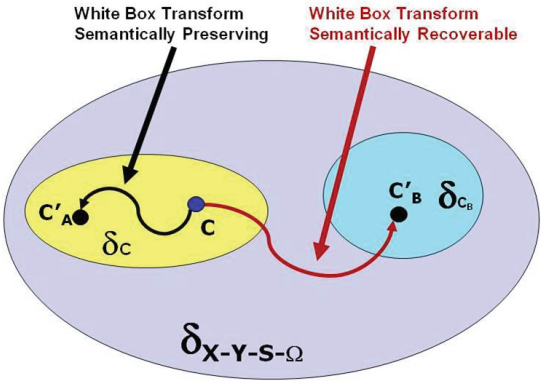


Figure 2: Obfuscation as Set Selection

To achieve this effect, RPM uses both semantically preserving whitebox and semantically recoverable blackbox transformations. In general, an obfuscating function has *only* two possibilities: whitebox changes which induce a blackbox transformation on the input/output and whitebox changes which preserve blackbox semantics. An obfuscator may change the whitebox structure of a circuit so that blackbox input/output relationships of the original circuit C are changed. Likewise, an obfuscator may change whitebox structure in such a way so that semantic equivalence with C is preserved. We illustrate this distinction in Figure 2 and note that we can alternatively view obfuscation as a set selection process.

2.3 Uniform Set vs. Iterative Selection

We design a framework that supports both semantic preserving/semantic recoverable transformations. For sake of brevity, we limit our discussion in this paper to the *whitebox, semantic-preserving* component. In other words, we only consider experiments where algorithms are sequenced, semantic-preserving structural transformations based on random or deterministic choices arranged in some random or deterministic manner. As Figure 2 illustrates, we can view an obfuscator as a program that selects programs from a set of functionally equivalent variations (i.e., polymorphic versions).

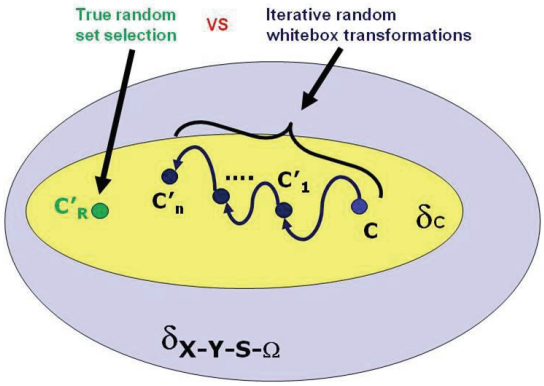


Figure 3: Random Uniform Set Selection versus Iterative Random Selections

For example, all semantic-preserving obfuscators that produce a variant of circuit C, where $C \in \delta_C$ and $\delta_C \subset \delta_{X\text{-}Y\text{-}S\text{-}\Omega}$, will select some (other) element of δ_C, regardless of the theoretical model we choose to describe its security. We may conceive of one obfuscation goal and measurement criteria as whether we have maximized the randomness between the intermediate gate structure of C and the intermediate gate structure of its variant (C'_A in Figure 2). This translates to the goal of creating the best variant (in terms of confusion) that still accomplishes the same function as C.

RPM assumes that the best-possible obfuscator under this criteria would be one that chooses a circuit variant C' from the entire set of functional equivalents (δ_C in Figure 2) in a random, uniform manner. This random choice would represent our best attempt at producing a variant with random properties, or saying it another way, our best attempt at producing a variant that has confused and diffused the topological structure of the original circuit C. Even if we bound the size of the circuit family for (which is

the primary factor in determining the set size of $\delta_{X\text{-}Y\text{-}S\text{-}\Omega}$ and thus the subset size of δ_C), enumerating all possible circuits with such a configuration is super-factorial in running time and storage requirements. However, if the circuit size is reasonably small, enumeration is feasible and we can select functional alternatives in a random, uniform manner. We leverage this fact in the construction of one half of our experimental framework (see Section 6) which deals with finding replacements for very small subcircuits.

As Figure 3 depicts, we summarize an ideal obfuscation selection process under RPM compared to achievable, practical obfuscation processes that we can build currently. RPM posits that we can do no better than an obfuscator which chooses an element in a uniform, random fashion from the set of semantically equivalent alternatives (i.e, δ_C). In Figure 3, C'_R represents such a choice. Current obfuscation techniques that perform iterative forms of confusion and diffusion, at best, only produce variants that are structurally close to each other. We are interested in how far we may alter an original circuit structure through small changes before it becomes indistinguishable from a truly random variant. After performing some sequence of small transformations, we focus on how much intermediate gate information of the original C is revealed by the final variant (C'_n in Figure 3).

One motivating reason for developing a whitebox variation environment is to explore whether random iterative selections might eventually approach a truly uniform selection from a large circuit family. If it is possible, we would expect the distribution of obfuscated circuits that come from an iterative random selection sequence obfuscator to be indistinguishable from a random uniform set selection obfuscator. In either case, we base the ideal variant to be one that has least correlation (according to some definable metric) with an original circuit.

Since we are not concerned with function hiding itself, we limit our concern to measuring how effective we can create a randomized variant of some original circuit. Our framework provides us a way to represent circuits and design experiments with a large option space in how small random alternatives are created. We present next the environment itself with a description of how we carry out experiments.

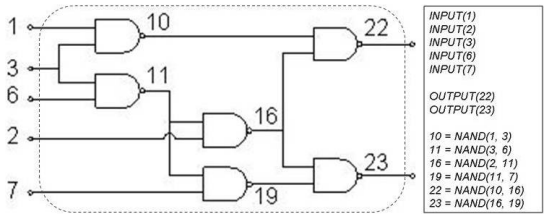


Figure 4: ISCAS Benchmark Circuit c17

2.4 Circuit Representation

Circuits have several manifest properties. We let $SIZE(C) = n{-}m{-}s$ denote that circuit C has input size n, output size m, and intermediate gate size s. For example, in Figure 4, $SIZE(c17) = 5{-}2{-}4$. Output gates are distinguished intermediate gates and together with the inputs define the denotational semantics of the circuit. We let Ω(C)={`NAND`, `XOR`} denote that circuit C has basis $\Omega = \{\texttt{NAND}, \texttt{XOR}\}$. For example, in Figure 4, Ω(c17)={`NAND`} and we would refer to $c17$ as a `NAND`-*only* circuit. For notational purposes, let Φ represent the set of all gates (intermediate or output) in a circuit C and let g_x represent a gate $g_x \in \Phi$. For example, in Figure 4, $\Phi = \{10, 11, 16, 19, 22, 23\}$.

In addition, we indicate the level of a gate g within a circuit by $level(g)$, which is synonymous with the trace level of a gate within a signal propagation hierarchy, assuming that every output signal has a final level of 0 and some virtual level where its Boolean logic signal is computed. Also, we let $|level(g)|$ represent the number of gates belonging to a particular level within the circuit. For example, in Figure 4, all inputs ($\{1,2,3,6,7\}$) are at level 3, $level(10) = level(11) = 2$, $level(16) = level(19) = 1$, $|level(16)| = |level(19)| = 2$. Each node within the DAG of a circuit C constitutes a specialized node with an associated Boolean logic function, derived from $\Omega(C)$.

We define any subset of gates $\alpha \subseteq \Phi$ as a

subcircuit of circuit C, and we use C_{sub} to help identify algorithmic selections. We designate an k-gate subcircuit as a selection by $[a_1, a_2, ..., a_k]$. As a final property of interest, a circuit (and by definition, any subcircuit) readily express its blackbox behavior by enumeration of all inputs, subsequent evaluation and propagation of signals on all intermediate gates, and recording of the corresponding output.

We refer to the full list of input/output pairs of the circuit as the *truth table*. The blackbox behavior of such a circuit may be succinctly expressed by the output signals corresponding to a canonical ordering of the 2^n inputs, which we refer to as the circuit *signature*. For instance, the signature for a 2-input and 1-output Boolean logic gate with AND functionality has a signature $< 0001 >$ while a 2-input OR logic gate has signature $< 1110 >$.

3 Experimental Configuration

We derive experiments based on textual descriptions of Boolean logic in BENCH format [7] and utilize a Java-based graph library package to support graph-based manipulation of the associated circuit DAG. Using this common DAG form, we compute a variety of graph-based, circuit-based, and semantics-based metrics. Our variation algorithm incorporate Kerckhoff's principles of cryptographic systems design [10]: namely, we give every possible choice made by the obfuscator as public knowledge while keeping only the precise set of steps used for a given obfuscation secret (much like the only secret part of a secure cipher should be the encryption key).

To perform whitebox transformation, we use a two-step *iteration* process which includes subcircuit *selection* followed by subcircuit *replacement*. Figure 5 illustrates the general notion, in two different views, of how we take an original circuit C and apply iterative changes to it that produce intermediate versions, C'_i. Each intermediate version, C'_i becomes the starting point for the next iteration which will produce the intermediate version C'_{i+1}. When we complete some n iterations of selection and replacement, the final variant becomes our candidate obfuscation variant, C'.

The large number of experiments which we may create using this approach derives from the nuance of each selection and replacement component. We say that a selection or replacement activity is *random* if we leave the choices of the algorithm completely open to a probabilistic dice-roll made by the algorithm (pseudo-random number generators suffice for this purpose). We say that a selection or replacement activity is *smart* if some criteria or user preference is used to guide or replace a probabilistic choice made by the algorithm. In the case of our selection/replacement algorithmic framework, the *obfuscation key* consists of the combined composition of all random and smart choices made during an experiment.

1. *Random selection*: Select a subcircuit $C_{sub} \subset C$ at random.

2. *Random replacement*: Select a replacement circuit $C_{rep} \in \delta_{C_{rep}}$ at random.

3. *Smart selection*: Only select subcircuits which have a particular property. If the subset of allowable selections contains more than one subcircuit, then one may be selected at random or based on another user-specified criteria.

4. *Smart replacement*: Similar to smart selection, only select replacement circuits from the library which have a particular property. If the subset of allowable selections contains more than one subcircuit, then one may be selected at random or based on another user-specified criteria.

We define a deterministic obfuscation *experiment* to be an 5-tuple: $(C, n, \xi, \sigma, \tau)$. We define the tuple as follows: C is an original circuit, n is the number of iterations, ξ is a set of selection algorithms with cardinality $|\xi| = n$ where $s_i \in \xi$ indicates the selection algorithm performed during iteration i, σ is a set of selection algorithms with cardinality $|\sigma| = n$ where

$r_i \in \sigma$ indicates the replacement algorithm performed during iteration i, and τ is a set of gates that are are given selection priority during the incremental execution of the experiment. The *trace* of an experiment records all pertinent information and metrics across all iterations of the experiment as well. It would, for example, indicate for each iteration, which specific set of gates or subcircuit was chosen for selection and which specific set of gates or subcircuit was chosen for replacement. It is possible, for example, that no suitable replacement could be found for a given selected subcircuit and given the constraints of the replacement criteria. Thus, some iterations of an experiment may return the same original circuit.

Since we design each selection/replacement iteration as independent, atomic operations, we use τ to represent the notion of a global experiment state where we may target some gates of interest in the original circuit. For example, we may have a smart (criteria) based *experiment* that stipulates at least one original gate be considered by every selection algorithm, until all original gates are replaced. This criteria would, over some number of iterations close to the original circuit size, guarantee that all original gates of a circuit are replaced at least once. If, after accomplishing such criteria, we reset τ to be all gates in the current iteration variant, we would then guarantee (after some number of iterations) that all original gates with their newly introduced gates would be replaced at least once as well.

4 Subcircuit Selection

Given an *experiment* defined as the tuple $(C, n, \xi, \sigma, \tau)$, ξ represents a set of selection algorithms and $s_i \in \xi$ indicates the selection algorithm used during iteration i. We define a subcircuit selection operation $C_{sub} = s(C, x, \gamma, \tau)$ with several characteristic attributes. The input to the algorithm is a circuit C, the (intermediate gate) size of the selection subcircuit x, the particular strategy $\gamma \in S$ (whether *smart* or *random*), and an optional set of gates τ that provide limiting criteria for the selection strategy itself. The set S of possible selection strategies (described below) must have all members defined a priori and we use the following set currently: S = {RandomSingleGate, RandomTwoGates, RandomLevelTwoGates, LargestLevelTwoGates, OutputLevelTwoGates, FixedLevelTwoGates, RandomAlgorithm}. The output of the algorithm C_{sub} is a circuit whose signature and $SIZE(C_{sub})$ forms the basis for functionally equivalent alternatives and replacement. As an example, iteration i that uses the `RandomSingleGate` strategy would be delineated as $s_i = s(C, 1,$ `RandomSingleGate`, $\emptyset)$, if we assume no experiment level criteria for selection/replacement.

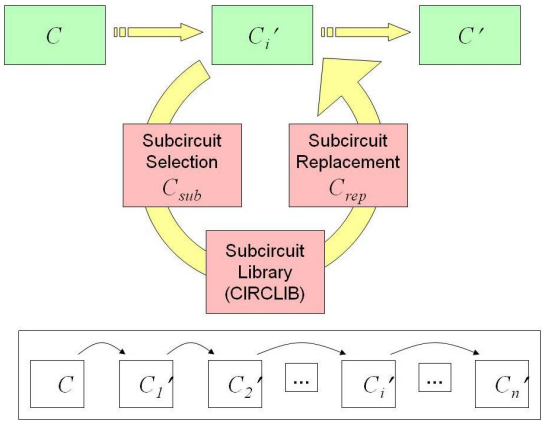


Figure 5: Iterative Substitution and Replacement

4.1 Selection Strategies

In terms of selection approaches, we presently experiment with six different subcircuit-selection strategies. The `RandomAlgorithm` strategy chooses any possible selection strategy for a single iteration of the experiment and we define five as follows:

- `RandomSingleGate` $\longrightarrow$ Choose $g_1 \in \Phi$ in a random, uniform manner.
- `RandomTwoGates` $\longrightarrow$ Choose $g_1 \in \Phi$ in a random, uniform manner. Choose $g_2 \in \Phi$ where $g_2 \neq g_1$.

- `RandomLevelTwoGates` $\longrightarrow$ Choose $g_1 \in \Phi$ in a random, uniform manner. Choose $g_2 \in \Phi$ where $g_2 \neq g_1$ and where $level(g_2) = level(g_1) \pm 1$ or $level(g_2) = level(g_1)$.

- `LargestLevelTwoGates` $\longrightarrow$ Choose $g_1 \in \Phi$ such that $|level(g_1)| = \ell_{max}$ where ℓ_{max} represents the maximum size of all levels within the circuit: $\ell_{max} = \sqcup\{|level(g_x)| \mid g_x \in \Phi\}$. Choose $g_2 \in \Phi$ where $g_2 \neq g_1$ and where $level(g_2) = level(g_1) - 1$ or $level(g_2) = level(g_1)$.

- `OutputLevelTwoGates` $\longrightarrow$ Choose $g_1 \in \Phi$ where g_1 is a distinguished intermediate gate (i.e, an output of the circuit). Choose $g_2 \in \Phi$ where $g_2 \neq g_1$ and where $level(g_2) = level(g_1) - 1$ or $level(g_2) = level(g_1)$.

- `FixedLevelTwoGates` $\longrightarrow$ Choose $g_1 \in \Phi$ where, for some user-provided level criteria k, $level(g_1) = k$. Choose $g_2 \in \Phi$ where $g_2 \neq g_1$ and where $level(g_2) = level(g_1) - 1$ or $level(g_2) = level(g_1)$.

- `RandomAlgorithm` $\longrightarrow$ Choose any selection strategy $\gamma \in S$ in a random, uniform manner.

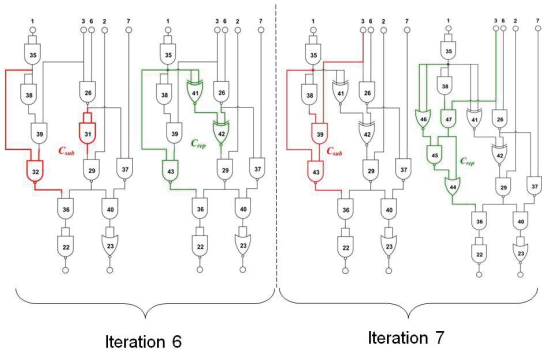


Figure 6: Iteration Example

Every random or smart selection strategy may be guided by criteria-based rules at the experiment level. When $\tau \neq \emptyset$, we modify the strategies listed by limiting the possibilities of at least the first gate chosen by the strategy. For example, an experiment that guarantees replacement of all original circuit gates would provide $\tau \subseteq \Phi$ to each iteration selection, which is to say that the strategy would make its first gate selection from the subset. If we used a `RandomSingleGate` strategy, the algorithm would instead choose $g_1 \in \tau$ in a random, uniform manner. Depending on the result of the replacement operation, if we *effectively* replace an original gate $g_x \in \tau$ (i.e., change fan-in, fan-out, or gate type), then we remove that gate from the set of possible first choices for the next iteration: $\tau = \tau \setminus \{g_x\}$.

4.2 Smart Strategy Limitations

A number of future, possible "smart" subcircuit selections can lead to NP-complete problems in generating the appropriate set of selectable subcircuits. For instance, a smart selection strategy based on subgraph isomorphism creates an NP-complete search, which is too computationally involved for large circuits. We may also develop selection strategies that look for specific Boolean logic functions (adders, multiplexer, decoder, comparator, etc.) for replacement. These would introduce greater than polynomial complexity to the obfuscator and would warrant heuristic options for the search.

5 Subcircuit Replacement

Given an *experiment* defined as the tuple $(C, n, \xi, \sigma, \tau)$, σ represents a set of replacement algorithms and $r_i \in \sigma$ indicates the replacement algorithm used during iteration i. We define a subcircuit replacement operation $C_{rep} = r(C_{sub},\ z,\ \psi, \Omega)$ with several characteristic attributes. C_{sub} is the circuit chosen for replacement, z is the requested gate size of the replacement circuit, ψ represents criteria that governs how we generate the replacement circuit library (described in Section 6.1), and Ω represents the basis choice of the replacement circuit. Given access to a selected circuit, we can derive the key characteristics that determine a replacement circuit library. $SIZE(C_{sub})$ gives us input size n, output size m, circuit (intermediate gate) size s. Combining this with

knowledge of the basis, Ω, we have enough information to *create* or *query* a circuit family.

As we mention previously, the replacement component of our experimental environment actually accomplishes for small subcircuits what we would desire to do for large circuits. Recalling Figure 2, the subcircuit library generator (seen as `CIRCLIB` in Figure 5) first creates a set of circuits $\delta_{n\text{-}m\text{-}s\text{-}\Omega}$. From this set of circuits, we choose randomly and uniformly an alternative variant for C_{sub} from the functionally equivalent subset $\delta_{C_{sub}} \subset \delta_{n\text{-}m\text{-}s\text{-}\Omega}$. Therefore, $C_{rep} \in \delta_{C_{sub}}$ and, ideally, $\delta_{C_{sub}} \neq \emptyset$. Based on the circuit selected and the criteria for replacement, there are a countless number of configurations in which there are no alternative replacements and thus $\delta_{C_{sub}} = \emptyset$. For example, there are no [2-1-1-{`NAND`}] circuits that implement the `AND` Boolean logic function with signature $< 0001 >$. Likewise, we could also design many experiments that, when given a circuit C, only return the original circuit C.

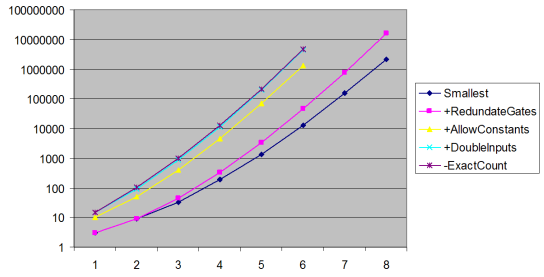


Figure 7: Circuit Library Sizes

Figure 6 illustrates two iterations (6 and 7) from an experiment with the `c17` circuit from Figure 4. The figure shows that in iteration 6, `CORGI` uses a two-gate selection strategy to choose the subcircuit $C_{sub} = [32, 31]$ and then, once it removes the subcircuit from the original, replaces it with $C_{rep} = [41, 42, 43]$. Both of these circuits belong to the C_{4-2-X} family. We note that the replacement increases the gate size of the overall circuit by one and increases the levelization also. Other effects of replacement may include changes to fan-in, fan-out, link length, unique input/output paths, unique paths through node, average paths per node, nodes per level, largest level, link length per node, average link length, and average nodes per level. We also note that, in iteration 6, gate 43 of C_{rep} is essentially the same gate 32 of C_{sub}: though renumbered, the gate has the same logic function, fan-in, and fan-out relationship. Figure 6 also demonstrates how the next iteration (7) of the experiment use a two-gate selection strategy to choose the subcircuit $C_{sub} = [39, 43]$, which resides in the C_{3-1-2} family and replaces it with a functionally equivalent $C_{rep} = [44, 45, 46, 47]$ which belongs to the C_{3-1-4} family. This example illustrates that we may grow the circuit size by virtue of replacing a circuit of size s with one of $s + 1$, $s + 2$, and so forth.

We note here that, if viable replacements were possible, we could easily replace size s subcircuits with functionally equivalent versions of size s, $s - 1$, or $s - 2$. It should make sense, that there are no single-gate replacement circuits for single-gate selection circuits, and there are some, but few, numbers of single-gate replacement circuits for two-gate selections. We discuss some of these relationships next, but point out that gate size and basis type drive the size of potential library classes. Currently our primary experimentation centers on single and two gate selection strategies, thus limiting our ability to report on optimizing or identical-size replacements at this time.

5.1 Library Generation Algorithm

Currently, we define only one iterative replacement algorithm but provide several basis-transforming operations (`NAND`-*only* , `NOR`-*only*) and structure-transforming operations (decompose multiple fan-in gates to dual fan-in, convert to sum-of-minterms form, convert to product-of-maxterms) that work at the whole-circuit level or do gate-by-gate replacement for all gates within the circuit. We focus currently on the use of purely random replacement choices versus smart options and describe next our recursive algorithm that enumerates circuit possibilities to produce a characteristic circuit fam-

ily. We conceptually view this as the creation process for the circuit family $\delta_{X\text{-}Y\text{-}S\text{-}\Omega}$ seen in Figure 2, where we start with the knowledge of input size, output size, gate size, and basis.

We begin with a discuss of what constitutes a "legal circuit", because the generation algorithm must enumerate all possible graphs which conform to a set of combinational logic constraints. Assuming that all circuits consist of inputs and a set of one or more gates with exactly two inputs each (2-input/1-output logic function gates), some of which we treat as outputs, there are still a few questions to ask. We characterize these questions as true/false queries that form a Boolean 6-tuple, which we define as ψ in the description of a replacement operation: $r(C_{sub},\ z,\ \psi, \Omega)$. We may vary these options for every replacement opportunity in an experiment, but typically choose a set of options ψ that remain constant for the entire sequence of iterations. Each option determines also how many circuits are produced, and we show in Figure 7 the exponential growth of library sizes (based on intermediate gates), based on different generation options for the C_{3-1-X} family as reference.

- `SymmetricGates` $\longrightarrow$ Are gates symmetric?
- `RedundantGates` $\longrightarrow$ Should we allow gates that are identical to other gates based on the inputs?
- `AllowConstants` $\longrightarrow$ Should we allow the circuit immediate access to the constants $True$ and $False$?
- `DoubleInputs` $\longrightarrow$ Should we allow both inputs to a gate to originate in the same place?
- `ExactCount` $\longrightarrow$ Does the set contain all circuits within a certain size bound or only all circuits of an *exact* size?
- `SimpleOutputs` $\longrightarrow$ Which gates may be outputs?

```
generateAll(gateNum)
{
   for each gate type:
      for each enumerateInputCombinations()
          add gate type with specified inputs
          if (RedundantGates and truth table of new gate is
              not equal to another gate's truth table)
              if (legalCircuit())
                  output circuit
              if(gateNum<size bound)
                  generateAll(gateNum+1)
}

enumerateInputCombinations()
{
   if(AllowConstants)
     include the constants True and False with the inputs
   select all gates and inputs g (1..n)
   if(SymmetricGates)
      deselect all combinations (a,b) such that a<b
   if(not DoubleInputs)
      deselect all combinations (a,b) such that a=b
   return remaining combinations
}

legalCircuit(){
   if (ExactCount and circuit does not contain
      the maximum number of outputs)
      return false
   else if (SimpleOutputs and any but the last numOutputs
      gates contain open outputs)
      return false
   else if (the circuit contains more than numOutputs
      number of open outputs)
      return false
   else
      return true
}
```

Figure 8: Circuit Enumeration Algorithm

These options are the primary way we may make smart choices about the libraries that we choose to make random selections from. Our first initial generation algorithm was very basic. However, by accounting for the six creation options listed above, we present a final refined version of the recursive algorithm in Figure 8. Several of the creation options govern what we refer to as *practical* versus *theoretical* constraints on circuit construction. For example, it is highly unusual for real-world logic circuits to have gates with inputs both coming from the same source (the `DoubleInputs` option). The `SimpleOutputs` option also gives ability to preclude circuit replacement options that have dangling intermediate gates that are never actually used. We observe from running many (5000+) experiments with varying number of iterations that randomly chosen alternatives of two, three, and four gate size typically are considered "bad" from the perspective of normal VLSI/ASIC circuit design. As a first goal, we want to consider the effect of purely

Table 2: Transformation Library Size

Transformation Library	DB Size
1 to 2 Gates	23.7 KB
2 to 3 Gates	53.6 MB
3 to 4 Gates	166.9 GB
4 to 5 Gates	934.9 TB

random replacements while learning what metrics best reflect either hiding properties of interest for reverse engineering purposes.

Once the enumeration algorithm generates (or locates) a circuit library with the appropriate circuit typology, it can find circuits within the family that match a particular (functional) signature. For our current experiments with 1 and 2 gate selection using 2, 3, or 4 gate replacement, we discover that it is more efficient to enumerate such libraries in memory versus access them from persistent data stores. As expected, we find that generation and retrieval of replacement candidates remains constant regardless of the circuit under consideration or the number of experiment iterations.

Table 3: Library Efficiency

	Usable Circuits	Total Circuits	Efficiency
1 Input - 1 Output	1,512	1,512	100.00%
1 Input - 2 Outputs	3,240	3,240	100.00%
2 Inputs - 1 Output	9,720	9,720	100.00%
2 Inputs - 2 Outputs	22,468	27,216	82.55%
3 Inputs - 1 Output	4,752	33,696	14.10%
3 Inputs - 2 Outputs	26,820	116,640	22.99%
4 Inputs - 1 Output	0	86,400	0.00%
4 Inputs - 2 Outputs	5,184	356,400	1.45%
Total	73,696	634,824	11.61%

5.2 Library Creation and Size

Because of the recursive nature of the algorithm, we can see the factorial blowup in Figure 7 of possible circuit numbers, using the 3-1-X family as an example. We also note that there are orders of magnitude in size difference based on the creation options. We have found from numerous experiments that ψ = (`SymmetricGates` = *true*, `RedundantGates` = *false*, `AllowConstants` = *false*, `DoubleInputs` = *false*, `ExactCount` = *true*, `SimpleOutputs` = *true*) produces circuits most like those we expect to see in traditional VLSI designs. We have discovered that certain option combinations produce gates which may be degenerate (all 1 or all 0), easily optimized away by a linear search algorithm, or produce redundant copies of either inputs, intermediate gates, or output gates. By using this approach, we also see the intractability of efficiently producing variants of larger circuits in a truly random way (if we want to use larger gate selection with full enumeration of the replacement alternatives). Table 2 illustrates the recorded disk space or memory requirements for several typical selection/replacement requests to the `CIRCGEN` library. To support 5 gate circuit replacements, we need almost 1 Petabyte.

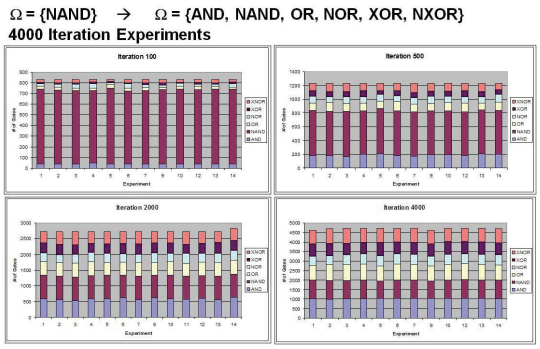


Figure 9: Uniform Gate Distribution Experiment

To improve efficiency further, we have the ability to cull out from a library circuits that have no expectation of every being used. This ability comes as an artifact of the way in which we select subcircuits to begin with and with the particular library creation options available to us. In particular, choosing a certain number of gates will result in variance between the actual circuit classes that contain equivalent circuits. For example, choosing two gates might result in a circuit with one input, two inputs, three inputs, or four inputs. In Table 3, we show the efficiency of choosing a subcircuit containing 2 gates and replacing it with a subcircuit containing 3 gates. We show the percentage of the gen-

erated subcircuits containing 3 gates which participate in transformation rules, meaning those which the algorithm would actually use in a replacement query. We note that the cost of storing only those circuits which can be used for replacements would be significantly less than the cost of storing all sub-circuits. We also note that a large body of future work remains to cull out circuit replacements which are by nature easy to find and reverse, though we leave the valid discuss of circuit reduction and logic minimization for another time. We are currently integrating various optimizing algorithms into the experimental framework as part of the variation process.

6 Obfuscating Measures of Interest

In thousands of experiments in our environment, we have ran various types of single and two gate selection and replacement experiments. Most of our experimental circuits come from ISCAS-85 Benchmark set or custom designed variants of comparators, carry-look-ahead adders, ripple-carry adders, multiplexors, decoders, and randomly generated circuits. Our maximum iteration run is 10000 currently, our largest *effective* selection and replacement size choice is 2 gates replaced with 2/3/4, and our largest real-world circuit for consideration has 3500 gates (we have processed randomly generated circuits with 10000 gates as well).

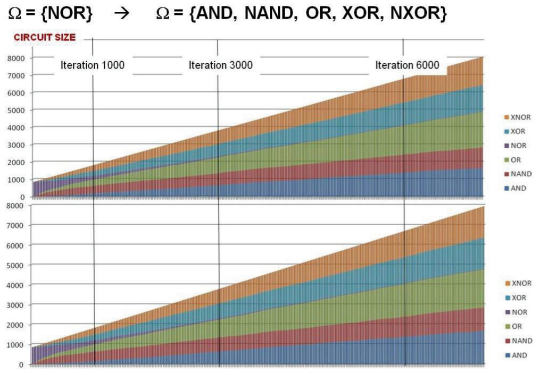


Figure 10: Full Replacement Experiment

Besides understanding basic metrics that we may collect from circuits undergoing structural change, we find interest in properties of the circuit that point to effective information hiding or beneficial mutations that foster real-world circuit protection goals. There are several information hiding properties of interest if we focus on the hiding of intermediate gate signals. We observe that the power of a random iterative algorithm with small selections size (1 or 2) to accomplish signal hiding is very small: mainly because 1-gate and 2-gate selections cannot physically or logically support hiding (regardless of any replacement we may use). Single-gate circuits, for example, will never hide the original signal because there must be a gate in the replacement circuit (regardless of gate size) that keeps the output behavior of the original gate. Two-gate circuits will only provide opportunity for hiding when gates are arranged in more than level (within the virtual circuit create by the selection itself, not their level within the circuit). If two gates chosen are independently related, then on average, random and criteria-based random selection strategies will not on average choose the structure that is suitable for signal hiding.

As another facet of information hiding, we have particular interest in whether the algorithm effectively replaces gates of the original circuit itself. We note that hiding an original *signal* is only one possible side effect of replacing an original *gate*: other possibilities include copying the original gate signal (redundancy), inverting the original gate signal, copying an input/output signal, inverting an input/output signal, or introducing degenerate gates that always produce either 1 or 0. As an example, an verb"AND" gate that has dual inputs originating from the same source will always duplicate the input signal (producing a buffer) while an `XOR` gate that has dual inputs from the same source will always output a 0, (producing a degenerate gate). We focus here simply on the nature of the algorithm to completely remove an original gate from the final version and leave for future analysis the reversibility properties of

the replacement.

6.1 Measuring Replacement

We report on three forms of experiments designed to measure gate replacement. We define gate replacement as the case where a gate chosen for selection does not appear in the replacement circuit in some renumbered form. This means that there is no gate in the replacement circuit with the same logic function (gate type), fan-in, and fan-out. We leverage the ability of our algorithm to choose the basis type of its replacement circuits to measure replacement. Given a replacement operation $r(C_{sub}, z, \psi, \Omega)$, we can vary Ω and thus guide the types of Boolean gates within the circuit over the course of the experiment. If we count the gate types of all gates within the circuit, over each iteration, we can tell when one type of gate no longer appears. If, for example, a circuit were a `NAND`-*only* circuit and we design an experiment where for all $r_i \in \sigma$, $\Omega = \{$`NOR`, `OR`, `AND`, `XOR`, `XNOR`$\}$, then the we know when we find the iteration where the number of `NAND` gates in the variant circuit $= 0$, we have replaced all the original gates of the circuit.

Figure 9 illustrates the first type of experiment where we begin with a `NAND`-*only* circuit of around 700 gates. We set the replacement basis Ω to be all six possible types: $\Omega = \{$`NAND`, `NOR`, `OR`, `AND`, `XOR`, `XNOR`$\}$. What we expect to see is that the circuit will manifest a fairly even distribution of gate types, assuming the selection/replacement operations are uniform as we expect. After conducting 14 separate 4000-iteration experiments using a `RandomTwoGates` selection strategy and 3-gate random replacement, Figure 9 shows the relative distribution of gate types for each experiment at iteration 100, 500, 2000, and 4000. What we observe are uniform distribution of gate types. Even for those gates that are `NAND`, they may not be original`NAND` gates either, but we do not account for those in this experiment.

Figure 10 illustrates a second experiment using a `NOR`-*only* circuit of around 850 gates

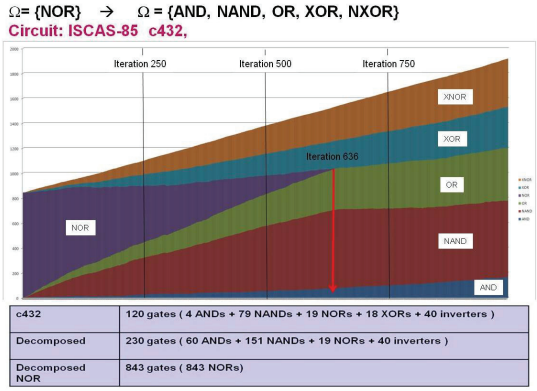


Figure 11: Smart Experiment/Full Replacement Experiment

(the decomposed, `NOR`-*only* variant of the ISCAS-85 `c880` circuit referenced in Figure 11). We show two (typical) results from 15 separate 7400-iteration experiments using a `RandomAlgorithm` selection strategy (weighted 75% towards `RandomTwoGates` selection strategy) and 3-gate random replacement. We use $\Omega = \{$`NAND`, `OR`, `AND`, `XOR`, `XNOR`$\}$ and expect to see an asymptotic decrease in `NOR` gates over time. In all of our experiments, purely random selection with no smart criteria at the experiment level always leaves some small number of original gates. This of course can be attributed to the fact that as the circuit grows in size, the small remaining original gate types become less likely to be chosen for replacement.

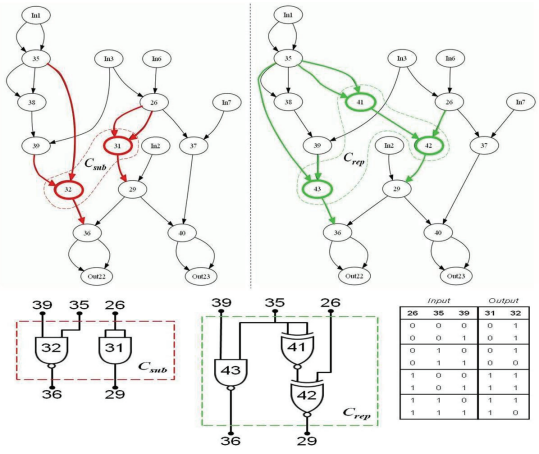


Figure 12: Crossover Example Context

Figure 11 uses the same NOR-*only* circuit as a starting point and illustrates the results of a (typical) single experiment out of 25 where we chose a smart selection approach at the experiment level (reference Section 4). In this case, we set $\tau = \Phi$ and indicate that selection algorithms should favor original gates as their first selection choice. Each experiment was a 1000-iteration experiments using a smart `RandomTwoGates` selection strategy and 3-gate random replacement. We use $\Omega = \{$`NAND`, `OR`, `AND`, `XOR`, `XNOR`$\}$ and expect to see all `NOR` gates to be removed from the circuit over time. As expected, in every experiment we saw all original gates removed from the circuit, on average, around iteration 630. In Figure 11, we know that the variant at iteration 636 contains no original gates. This illustrates the usefulness of smart-based variants of strategies which may be affected at the experiment level.

6.2 Control Diffusion and Redundancy

We conclude with a brief discussion of another circuit artifact of interest in both reverse engineering and mission assurance. By virtue of the two-gate selection strategies we specify, when gates in independent control paths are chosen for selection and replacement, the replacement circuit induces a control flow or diffusion within the circuit that did not exist before. Using the old adage that one man's trash is another man's treasure, the criticisms we give for small two-gate, cut set selection/replacements to provide signal hiding do on the other hand foster the ability to duplicate signals. When signals become duplicated in new control flows, this property may further goals such as fault tolerance or open up new methods of producing modular redundancy. Figure 12 illustrates this behavior, which occurs in nearly 95% of all `RandomTwoGates` selection strategy experiments. We highlight in this iteration example, $C_{sub} = [32, 31]$, which resides in the C_{3-2-2} family. Gates 31 and 32 have no dependency or control flow between them before the selection and replacement operation. Once chosen for selection, however, their replacement induces a new control flow when $C_{rep} = [41, 42, 43]$. We leave for future work and results more extensive analysis of this phenomenon, but note here that the current set of our experimental strategies for two-gate selection create this behavior with high probability. We also leave for future analysis the resilience of such constructions to detection or removal.

7 Conclusion

In this paper we present a framework for white-box circuit variation and describe our efforts to understand the effect of random and deterministic subcircuit selection and replacement on hiding properties of interest. We show the value of the framework for answering questions related to randomness as an obfuscation metric in considering circuit variants that may be used in reprogrammable hardware environments such as FPGAs. We give results of initial experimentation in support of specific questions such as gate replacement and gate diffusion/crossover. For brevity, we do not discuss all initial findings here but do expect in future work to report results related to a wide variety of questions: larger gate selection strategies, alternate possibilities for circuit library generation, impact of reduction or reversal algorithms, attempts for larger circuit library generation and storage, optimization and steady-state circuit replacements, and measurements of physical characteristics with real-world circuits.

References

[1] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. *Electronic Colloquium on Computational Complexity*, 8, 2001.

[2] Mohamed R. Chouchane, Andrew Walenstein, and Arun Lakhotia. Statistical signatures for fast filtering of instruction-

substituting metamorphic malware. In *WORM '07: Proceedings of the 2007 ACM workshop on Recurring malcode*, pages 31–37, New York, NY, USA, 2007. ACM.

[3] Frederick B. Cohen. Operating system protection through program evolution. *Comput. Secur.*, 12(6):565–584, 1993.

[4] Christian S. Collberg and Clark Thomborson. Watermarking, tamper-proofing, and obfuscation: tools for software protection. *IEEE Trans. Softw. Eng.*, 28(8):735–746, 2002.

[5] Patrick Cousot. Constructive design of a hierarchy of semantics of a transition system by abstract interpretation. *Theor. Comput. Sci.*, 277(1-2):47–103, 2002.

[6] S. Goldwasser and G. N. Rothblum. *On best-possible obfuscation*, pages 194–213. 4th Theory of Cryptography Conference, TCC 2007. Proceedings (LNCS Vol. 4392). Springer-Verlag, Germany; Berlin, 21-24 Feb 2007.

[7] M.C. Hansen, H. Yalcin, and J.P. Hayes. Unveiling the iscas-85 benchmarks: a case study in reverse engineering. *Design & Test of Computers, IEEE*, 16(3):72–80, 1999.

[8] Michael Huth and Mark Ryan. *Logic in computer science: Modelling and Reasoning about Systems*. Cambridge University Press, 2004.

[9] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. pages 463–481. CRYPTO, 2003.

[10] Auguste Kerckhoffs. La cryptographie militaire. 9:5–38, January 1883.

[11] J. Todd McDonald, Yong C. Kim, and Alec Yasinsac. Software issues in digital forensics. *ACM Operating Systems Review*, 42(3), April 2008.

[12] J. Todd McDonald and Alec Yasinsac. Applications for provably secure intent protection with bounded input-size programs. In *Proceedings of the International Conference on Availability, Reliability and Security (ARES 2007)*. IEEE Computer Society, 10-13 April 2007.

[13] Mila Dalla Preda and Roberto Giacobazzi. Semantic-based code obfuscation by abstract interpretation. In *ICALP*, pages 1325–1336, 2005.

[14] T. Sander and C. F. Tschudin. On software protection via function hiding. *Information Hiding*, pages 111–123, 1998.

[15] Kris Tiri and Ingrid Verbauwhede. Design method for constant power consumption of differential logic circuits. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 628–633, Washington, DC, USA, 2005. IEEE Computer Society.

[16] Andrew Walenstein, Rachit Mathur, Mohamed R. Chouchane, and Arun Lakhotia. Normalizing metamorphic malware using term rewriting. In *SCAM '06: Proceedings of the Sixth IEEE*, pages 75–84, Washington, DC, USA, 2006. IEEE Computer Society.

[17] Alec Yasinsac and J. Todd McDonald. Tamper resistant software through intent protection. *The International Journal of Network Security*, 7(3):370–382.

[18] Alec Yasinsac and J. Todd McDonald. Of unicorns and random programs. In *Proceedings of the 3rd IASTED International Conference on Communications and Computer Networks (IASTED/CCN)*, 8-10 Nov 2005.

[19] Yu Yu, Jussipekka Leiwo, and Benjamin Premkumar. Hiding circuit topology from unbounded reverse engineers. In *ACISP*, pages 171–182, 2006.

[20] Yu Yu, Jussipekka Leiwo, and Benjamin Premkumar. Private stateful circuits secure against probing attacks. In *ASIACCS*, pages 63–69, 2007.

False Alarm Reduction in Automatic Signature Generation for Zero-Day Attacks

Daniel Wyschogrod
BAE Systems
Dan.Wyschogrod@baesystems.com

Jeffrey Dezso
BAE Systems
Jeffrey.Dezso@baesystems.com

Abstract

The techniques and supporting tools for signature based intrusion detection have reached a high level of maturity. They are well understood by the community and have hardware implementations capable of matching rules at high speed. Their major shortcomings involve handling "zero-day" attacks. Anomaly or protocol-adherence based sensors are capable of detecting zero-day attacks, but with high false alarm rates and at more limited speeds. The design proposed here combines the zero-day detection capabilities already supplied by anomaly detection front ends with the speed, hardware compatibility and mature infrastructure of signature based systems. A unique capability of this proposed technology is that false alarm rates of matched rules can be reduced to arbitrarily low levels by increasing the amount of training on benign traffic. A goal of future work would be to produce an efficient and secure mechanism to distribute automatically generated signatures with the goal of broadening the perimeter of protection and blocking attacks farther away from sensitive servers and hosts.

1. Introduction

Various commercial and open source systems currently exist for signature-based intrusion detection. Many of these systems are at a high level of maturity and are able to match large numbers of signatures at high data rates. Typically, signatures are generated by highly trained analysts who evaluate attacks by hand and perform manual extraction of byte sequences of interest. Once generated, signatures can be installed on existing "bump-on-the-wire" hardware appliances or coprocessors that provide line rate deep packet inspection capabilities. However, the principal shortcoming of existing IDS technology is the inability to handle attacks for which no known signature applies. These attacks, known as "zero-day" exploits, have the highest effective penetration during the vulnerability window between the time the exploit is unleashed and the time signatures are created and uploaded via traditional means. At present, this vulnerability window can persist over a significant time scale.

On the other hand, various sensors exist, many in the experimental stage, for the detection and analysis of zero-day attacks [1]. Typically, these sensors are either anomaly-based or protocol-adherence based. The primary difficulties with these technologies include high false alarm rates, sensitivity to the statistics of background traffic and system response times. Also, many of these sensors detect attacks only after a system has been compromised.

In this paper, we discuss a system which receives as input sets of packets identified as malicious and outputs signatures that can automatically be used as inputs to classical signature based matching systems. Of particular importance is our innovative mechanism for driving down the false alarm rate associated with the produced signatures via using readily available samples of benign, non-attack traffic.

2. Approach

In this section, we describe the approach and goals motivating our design.

Firstly, the ASG subsystem is not a sensor. It is not designed to perform the initial detection of an attack. It is intended to operate as a second stage to high quality front-end sensors. The desired result of our backend module is a set of signatures with a very low false alarm rate. High false alarm rates serve as a continuous denial of service and are therefore considered a major system risk. These signatures are designed to be compatible with classical signature matchers so that they can be easily disseminated to distant pattern matchers. The advantage of this approach is that signature matchers are widely available, are typically cheaper than other intrusion detection technologies, and have been implemented in ASIC and FPGA and can therefore keep up with network data rates. Ultimately, the purpose of signature dissemination is to automatically extend the perimeter of protection.

Finally, in the presence of polymorphic attacks (attacks that change from instance to instance), in order to find novel instances of an attack, one must produce signatures for attack invariants (parts of the attack that are constrained to remain constant). It is therefore important to produce signatures of minimum length that still also have low false alarm rates.

It should be noted that if any signatures are found for an attack and survive filtering, they will always match an identical repeat of the attack. In the presence of polymorphism, the probability of detection will depend on our success in generating signatures for critical attack invariants. Stated another way, any signatures derived from the attack will have a zero missed detection rate for repeated instances of the identical attack, while if they contain many relatively short signatures they have a good chance at detecting polymorphic or other variants. We hope to address the question of the effectiveness of the technique against polymorphic attack in a subsequent paper.

3. Benign traffic filtering

At the core of our innovation is the capability to drive down false alarm rates for signatures produced from instances of attacks. Existing signature extraction systems [2] use only multiple examples of exploits to generate signatures. These systems do not directly address questions of false alarm rates. In addition, instances of exploits are hard to come by while benign traffic examples can be easily acquired.

3.1 Outline of technique

The technique used by our system involves observing large numbers of network packets containing benign, non-attack traffic and extracting all substrings within a given range of lengths and storing them in efficient data structures. When presented with a set of attack packets, the system applies a technique involving benign traffic filtering where only substrings never seen in benign traffic are used to produce signatures.

3.2 Trie based filtering

The filtering operation can be performed efficiently using a *trie* data structure. Tries store large numbers of strings compactly and allow for straightforward implementation of intersection, subtraction and other operations [3]. During the benign traffic training phase, we create a *trie* for each TCP or UDP service of interest which contains all of the extracted substrings from all observed packets. A simple example of such a structure is shown in Figure 1 and is labeled “Benign Traffic Trie”. A double circled node indicates the terminal node of a string. Note that all substrings of all lengths can be included, but usually they are limited to substrings within a length range.

A smaller *trie* is produced for a set of packets corresponding to an attack. Subtracting the benign traffic *trie* from the *trie* of the attack packets leads to a difference signature *trie* which contains substrings which are used as signatures for that attack. A simple example is worked out in Figure 1. The Benign Traffic Trie in the example is extremely small for illustrative purposes.

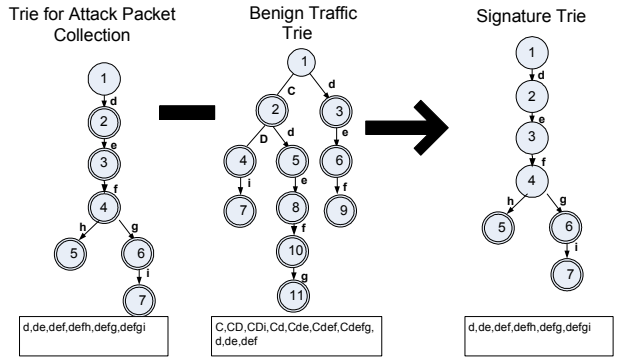


Figure 1: Signature trie created by subtracting benign traffic trie from attack packet trie.

A number of points should be noted with regards to this methodology. They include:

1) All substrings found in benign traffic which would lead to false alarm signatures are removed.
2) As more benign traffic is used for the production of the benign traffic *trie*, the system does a better job of filtering out less frequent occurrences of signatures that may occur in benign traffic (this is discussed at greater length below).
3) Since short signatures will be produced when possible, there is a better chance that attack invariants for polymorphic attacks will be captured.
4) If a service is highly structured, as time goes by strings will repeat themselves and the benign traffic *trie* will grow more slowly. If the service has random content, the *trie* will grow quickly.

4. Predictive benign filtering models

Predictive models are useful for comparing the observed performance of signature filtering schemes with expected results. The purpose of this section is to present a basic analytical model for relating the amount of benign traffic used for training to the expected false alarm rate for attack signatures from benign test traffic. We first develop a model for false alarm rates associated with attack packet substrings that have been filtered by benign traffic. We then present two specializations to characterize important classes of benign traffic.

4.1 False alarm rate derivation

Ultimately, we wish to install a set of signatures into a pattern matcher to detect repeat instances of an attack that has been seen previously and for which attack packets have been captured. Clearly, these signatures will have a certain false alarm rate when exposed to real world non-attack traffic. For a false alarm to occur, a number of events must conspire to set off the alert. They are:

1. In the initial training of the benign database, the string that ultimately will lead to the false alarm must be missing. If it were present, the signature leading to the false alarm would have been filtered out. The probability that the string is missing from the benign database is given by $\boldsymbol{P_{mdb}(s)}$
2. The false alarm string must have occurred in an attack packet. Attack packets are mixtures of unique attack strings surrounded by elements of everyday syntax and data. The average percentage of an attack packet that is really benign is given by $\boldsymbol{P_a(s)}$**.**
3. The false alarm string must occur in subsequent non-malicious traffic. The frequency of occurrence of the signature string in a benign traffic stream is given by $\boldsymbol{P_b(s)}$**.**

We now present expressions for the three probabilities based on combinatorics. They are:

$$P_{mdb}(s) = (1 - \frac{1}{N_s})^{S_c}$$

$$P_a(s) = P_{B/A} \bullet \frac{1}{N_s}$$

$$P_b(s) = \frac{1}{N_s}$$

where S_c is the number of bytes collected in the benign database and N_s is the mean number of input bytes between sightings of a benign string s.

If we now take the product of the three necessary factors for a false alarm to occur on subsequent network traffic and integrate over the various mean number of input bytes for each string we have:

$$P_{FA}(S_c) = P_{B/A} \int_1^{\infty} \frac{W(N_s)}{N_s^2} (1 - \frac{1}{N_s})^{S_c} dN_s$$

where $W(N_s)$ is the weighting function for occurrences of strings with equal values of N_s. In the following section, we calculate the value of the integral for various assumptions about $W(N_s)$.

4.2 Models for types of service

In this section, we build two simple mathematical models for different types of services. Real world services can adhere reasonably well to one of these models but can also be a combination.

4.2.1 Syntactic Data Services

In this model, we assume that text is linguistic in nature in that it is composed of words that repeat themselves with varying frequency. We approximate word frequency using Zipf's law [4] with the *s* parameter set to 1 which is characteristic of English text. Under these assumptions, the integral of section 4.1 can be evaluated with $W(N_s) = \frac{1}{\overline{W} K_Z}$ and thus constant and the probability of false alarm, from evaluating the integral, is given by:

$$P_{FA}(S_c) = \frac{P_{B/A}}{\overline{W} K_Z} \frac{1}{1 + S_c}$$

where S_c is the amount of traffic that has been collected for benign traffic training. P_{FA} is the false alarm rate per MB of test benign traffic seen. $P_{B/A}$ is the average percentage of each attack packet that contains benign traffic. $\overline{W}$ is the average word size. K_z is the Zipf normalization factor.

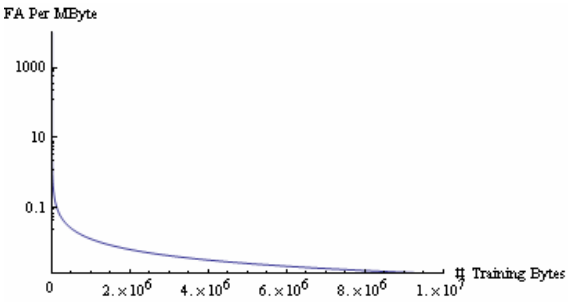


Figure 2: False alarms suppression as a function of training sample size.

This model predicts that the number of false alarms per MB of test data is inversely proportional to the number of MB of training data as illustrated in Figure 2.

Clearly, false alarm rates drop rapidly at first with reasonable amounts of training, but the tail requires more and more training to further lower the false alarm rate. This is reasonable, since we must wait longer and longer to capture the few uncommon strings that we haven't seen yet.

4.2.2 Random Data Services

In this second model, services can be modeled as streams of uncorrelated, random data. Encrypted services, such as SSH or SSL, fall into this category. Services which transport compressed data also have this behavior. This leads to a weighting function in the integral of section 4.1 given by:

$$W(N_s) = \delta(N_s - 2^{8|s|})$$

which is a Dirac delta function where $|s|$ is the length of the string.

The result of evaluating the integral is given by a model of the form:

$$P_{FA}(S_c) \approx \frac{P_{B/A}}{2^{16|s|}}$$

P_{FA} is the false alarm rate per MB of test benign traffic seen. $P_{B/A}$ is the average percentage of each attack packet that contains benign traffic. $|s|$ is the length of a false alarm string in characters and S_c is the amount of traffic that has been collected for benign traffic

training. Note that this result is not dependent upon the size of the training set.

The above proportionality expresses the fact that as a random string becomes longer, the probability of seeing it again becomes exponentially smaller. Thus, for a random data service, an acceptable false alarm rate can be "chosen" by selecting an appropriate minimum signature length. Further, the implication is that if a service is purely random, as long as we choose strings as signatures that are of sufficient length, we can reduce the false alarm rate arbitrarily and that we would not even have to train. On the other hand, training on random data over a sufficiently long time simply captures most of the shorter strings and a smaller and smaller percentage of longer strings so that effectively we filter only short strings from potential signature sets.

In the real world, services that are completely random, such as SSH or SSL are frequently encrypted making it impossible for an anomaly detector to find the initial attack. .In the case of "mixed" traffic such as syntactic traffic with embedded binary data, training is needed for the syntactic component. We aren't typically harmed by training on the random element since it will simply end up filtering out a good portion of the universe of short strings and a smaller percentage of longer strings.

The ultimate outcome is that random traffic as part of attacks may create large numbers of signatures which will not be filtered out since they are unlikely to occur again in benign traffic, but the fact that they won't occur in benign traffic makes them unlikely to be a source of false alarms. Generally, short string sequences reoccur more frequently and are thus filtered out, while long random signatures survive since they occur most infrequently.

5. Experimental results

In this section, we examine a number of experimental results. We first examine some statistics describing the rate at which the number of novel N-grams in the benign traffic *trie* grows. We then examine the number of false alarms per MB of test data as a function of the number of MB of training data for a number of attacks launched using the Metasploit framework [5].

5.1 Benign Traffic Training Data Analysis

Figure 3 illustrates benign traffic database size in terms of number of unique inserted N-grams as a function of number of bytes of training data. Here, the N-gram size is fixed at 5. The data was taken from a representative sample of actual network traffic. The illustration shows a number of different types of services. The growth of the size of the SSH database, which is an encrypted service and therefore random, is explosive. On the other extreme, port 631 which is the Internet Printing Protocol, is syntactical in nature with a small vocabulary and therefore flattens out very quickly. Another interesting finding involves http, port 80. Two plots are displayed. The first presents results for all packets to port 80. At one particular point, the plot rises dramatically. On examination of the data, it turned out that an image was being transmitted. The second plot removes the packets containing the image data to illustrate that pure protocol packets demonstrate a less dramatic increase in the number of unique N-grams.

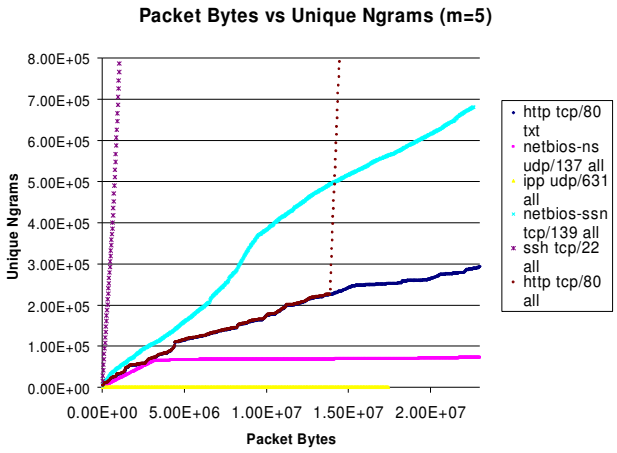


Figure 3: Number of unique N-grams as a function of number of bytes of training data. N-gram size is fixed at 5.

5.2 Pattern matching false alarm rates

Figure 4 illustrates the false alarm probability on test traffic of a set of signatures for a Port 80 Apache attack [6] for various amounts of training data. We compare this to a best fit syntactic model curve. The fit is quite close with the exception that the actual data's false alarm rate decreases in discrete steps as various signatures are removed with more training. It should be noted that for this data set, with sufficient training, the false alarm rate actually reaches zero.

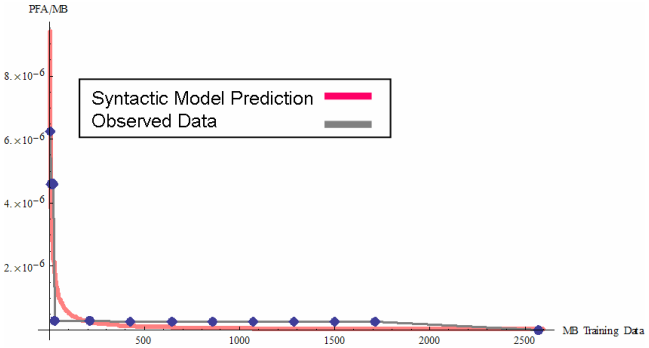


Figure 4: Apache attack signatures and FA rates as a function of benign traffic training. Test set contains 856 MB of packet payload.

In Figure 5, we show a port 139 Samba attack [7]. The Samba attack requires less training to achieve a zero false alarm rate. For the port 80 case, we require a training set about three times the size of the testing set to achieve zero false alarm, while with 139 the factor is only two.

5.3 Characterization of surviving signatures as a function of training

Figure 6 illustrates the number of surviving signatures after various amounts of filtering both for the Apache (port 80) and Samba (port 139) attacks. As can be seen from Figure 6, significant numbers of signatures are produced for both attacks. Two important points can be extracted from this plot. The first is that the number of signatures produced for the Apache attack is much higher than for the Samba attack. The second is that most false alarms are due to only a few patterns since we reach zero false alarms by removing only a few rules.

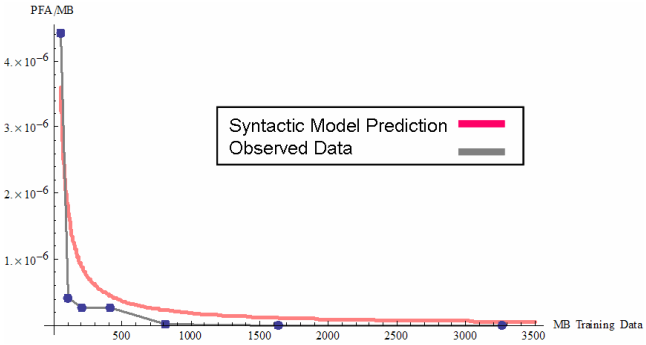


Figure 5: Samba attack signatures and FA rates as a function of benign traffic training. Test set contains 3,265 MB of packet payload.

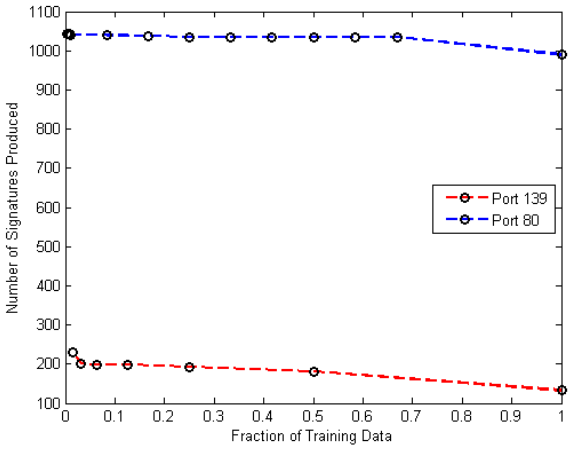


Figure 6: Number of rules that survive filtering as a function of the fraction of training data needed to get to zero false alarms.

In Figure 7 and Figure 8 we examine how false alarms are distributed among surviving signatures at different points in training. Clearly, we can see that the Apache attack's false alarms are more distributed over multiple rules and survive until quite late in the training, while the Samba attack narrows the false alarms to just a few rules very quickly.

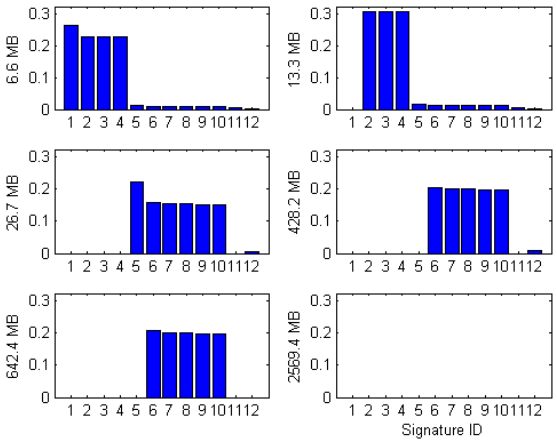


Figure 7: False alarm distribution among surviving signatures for the Apache (port 80) attack. Signature IDs are consistent across histograms.

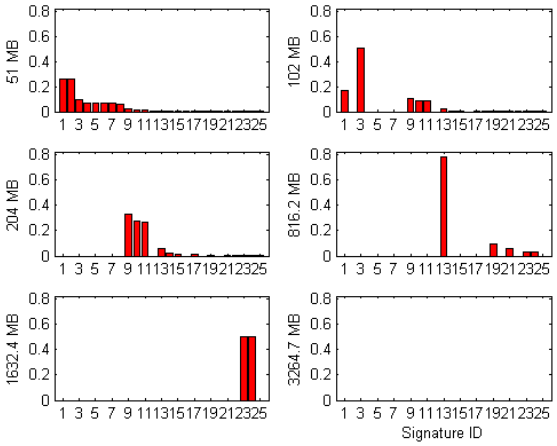


Figure 8: False alarm distribution among surviving signatures for the Samba (port 139) attack. Signature IDs are consistent across histograms.

Our current hypothesis with regards to the difference in false alarm behavior between port 80 and port 139 is that port 80 benign traffic has more of a random component and therefore filler elements in the attack will be filtered so that false alarm signatures produced from filler in the attack and are distributed and survive randomly. The port 139 traffic, in contrast, is highly syntactic, and the offending signatures are quickly filtered out. However, additional analysis needs to be performed to verify this assertion.

Finally, we consider the issue of survival of short signatures. Short signatures are beneficial in that if they survive, they can better capture attack invariants which may be short and for which we want to avoid "bleeding" into neighboring polymorphic elements. Unfortunately, but perhaps understandably, far more signatures of length 7 survive (the longest strings allowed) than of any other length. An important item in follow up work would be to determine if discernable attack invariants survive our filtering for a variety of attacks. Figure 9 and Figure 10 summarize our results for the two exploits studied.

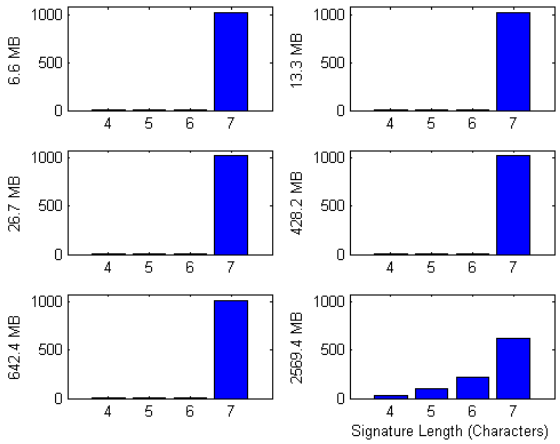


Figure 9: Distribution of pattern lengths as a function of training for the Apache (port 80) exploit. We believe there is a software bug in the last histogram.

6. Enterprise-wide application

The current ASG system has been combined with both anomaly based and honeypot type front-ends and has run in prototype environments. The produced signatures were tested for false alarm rate performance and behaved as expected.

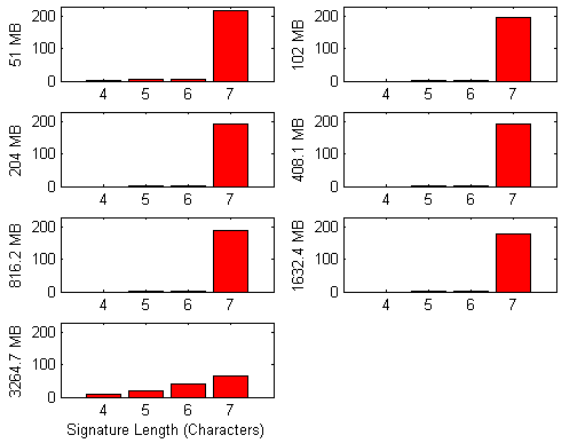


Figure 10: Distribution of pattern lengths as a function of training for the Samba (port 139) exploit. We believe there is a software bug in the last histogram.

An important future direction is the integration of Automatic Signature Generation technology with advanced sensors and high speed pattern matchers throughout the enterprise and out into the cloud. This is motivated by the simple observation that signatures produced locally can be distributed globally in order to extend the perimeter of protection. High quality but expensive and slow sensors can be used to capture instances of attacks from which signatures can be extracted. These signatures can then be distributed to enterprise gateways and to other installations (see Figure 11). In many cases, systems that are elaborately instrumented with anomaly detectors may end up being compromised as part of the attack process, with the net benefit that the attack is detected and its corresponding network traffic captured. The network pattern matchers, however, would be at some distance from sensitive hosts and servers and would block instances of re-infection far from their targets.

While the signatures produced by ASG would be more numerous and less tailored than those produced by a human analyst, they would serve as initial zero-day protection. In addition, within reasonable limits, string matchers don't degrade in performance with large numbers of patterns, particularly those implemented in hardware so that the large number produced would not be too great a concern. As time went on, the automatically generated signatures could be used as aids to human analysts in producing refined, hand-tuned versions which would presumably also decrease their number.

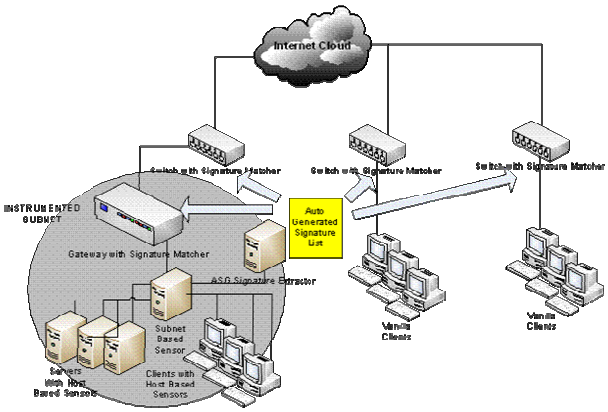


Figure 11: Distribution of signatures to other subnets.

Secure, efficient automated signature distribution would thus constitute a significant step towards the goal of perimeter of protection expansion.

A more distant goal would be to propagate signatures at a rate sufficient to contain a propagating threat, such as a worm. This would require high-speed, real-time responses on the part of various system components.

7. Conclusions

We feel that using Automatic Signature Generation as an add-on component to various zero-day exploit detection systems has a number of important advantages:

1) As a separate component, it can integrate with a variety of sensors. As sensors improve their sensitivity, signatures will benefit.

2) Improvements in false alarm rates are achieved through training on benign traffic. Additional attack instances are not needed. Benign traffic is typically readily available.

3) Existing high-speed signature matching capabilities are leveragable for use with produced signatures.

4) Random data services may produce more signatures but do not increase false alarm rates.

5) Perimeter of protection capability allows the user to move defenses farther away from vulnerable assets.

6) Future possibility of protection outstripping propagating threats.

7) Ability to assist human analysts in drawing attention to attack portions of packets involved in an exploit.

8. References

[1] Parekh, J. J., Wang, K., Stolfo, S. J. 1999. Privacy-Preserving Payload-Based Correlation for Accurate Malicious Traffic Detection.

[2] Newsome, J., Karp, B., Song, D. 2005 Polygraph: Automatically Generating Signatures for Polymorphic Worms. 2005 IEEE Symposium on Security and Privacy

[3] Rieck, K., Laskov, P. 2006. Detecting Unknown Network Attacks using Language Models. Third International Conference on Detection of intrusions and malware & vulnerability assessment, Berlin, Germany

[4] Manning, C.D., Schutze H. 1999. Foundations of Statistical Natural Language Processing. MIT Press

[5] http://www.metasploit.com.

[6] http://www.osvdb.org/838

[7] http://www.osvdb.org/4469

Does your analysis platform accept the following data sets?

LOGICAL TOPOLOGY – GLOBAL INTERNET AND ENTERPRISE LEVEL
- Border Gateway Protocol (BGP)
- Internal and external routing protocols
- Regional Internet Registries (RIRs)
- Internet Routing Registries (IRRs)
- Global Traceroute
- Network flow

PHYSICAL TOPOLOGY
- Geo-location
- Physical asset location (building, room, owner, etc)
- Internet Protocol (IP) address to country mapping
- AS and CIDR to country mapping

TRANSPORT MEDIUM
- Fiber links
- Satellite links
- Wireless links
- Trusted Internet Connections

THREAT WEATHERMAP
- Malicious activity data (Botnet, phishing, etc)
- Distributed Denial of Service (DDoS) information
- Environmental Data (outage, weather, and natural disaster overlay)

LOOKINGGLASS WHITEPAPER

The Cyber Intelligence Mecca: Ten Rules for Achieving Cyber Situational Awareness

INTRODUCTION

One thing is for certain, network analysts are overwhelmed with the amount of data available, and current analysis tools are not designed for the rapidly increasing data sets or demands created by modern networks. *Identifying an emerging threat, identifying the nature and extent of the threat, and gaining perspective on its possible impact requires complete visibility into vast Internet pathways and real-time data integration*. In order to achieve true cyber situational awareness – and be in a position to maximize defenses and minimize business risk and exposure to cyber attacks of any kind – ten key rules must be followed.

RULE #1: MOVE BEYOND MANUAL

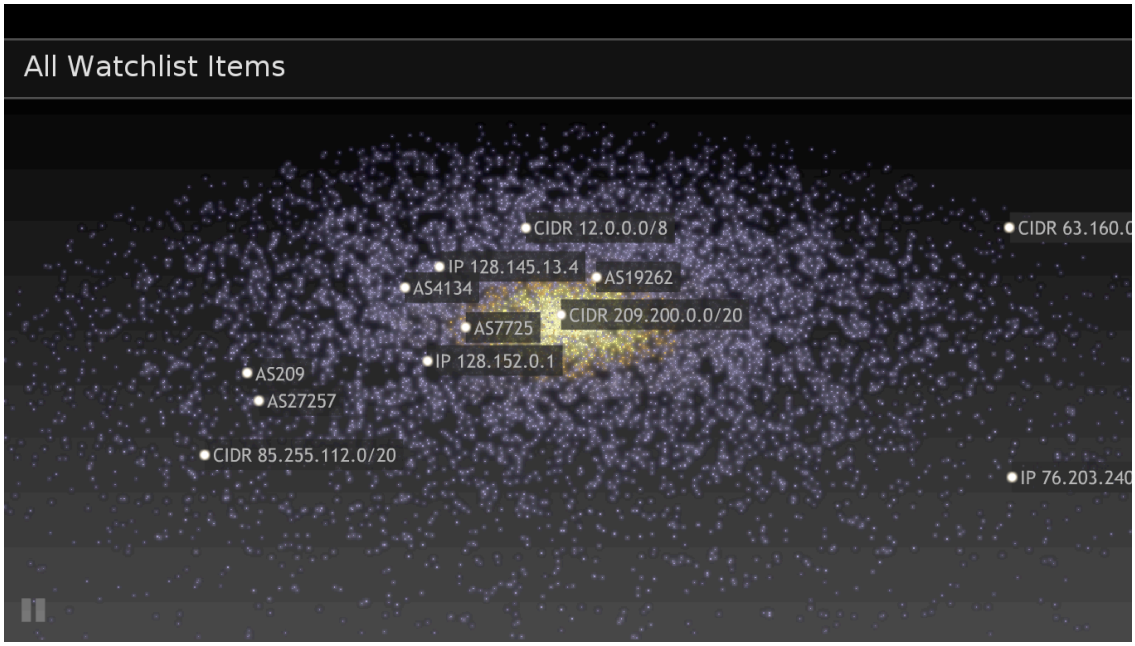
Most rudimentary tools involve labor intensive, text-based manual analysis and patchwork – therefore missing the benefits of visual analysis. In order to evolve with all types of analysis, automation is the key to your cyber intelligence platform. Automation will allow you to focus on tasks where human intervention is necessary, and improve the overall speed and accuracy of analysis. As the amount of data being processed and the reliance on the system to produce accurate information increases, an automated platform will soon become the cornerstone of your analysis.

RULE #2: GET HOLISTIC

There are a number of network monitoring applications available today that tend to focus on one specific area of concern or insight; however, these traditional monitoring tools are not enough. What is needed is a holistic view of activities, patterns, and connections that provide insight into hidden associations beyond the enterprise network. *Presenting data relationships in a meaningful way improves analysis, by allowing you to see patterns in the data you might otherwise have missed.*

RULE #3: VISUALIZE THE NETWORK

Most visualization solutions that currently exist are renderings of limited data sets pulled from text-based tools - failing to provide useful results. As a network analyst, you need a next-generation, flexible analysis solution that provides you with actionable information and adapts to the speed and growth of the Internet. By selecting a platform that contains both textual and visual components, you'll be able to measure quantities, map similarities and identify hidden relationships. Patterns become more obvious and integration with other systems becomes easier. For example, overlaying logical and physical routing and infrastructure data with traffic flows and attack data can reveal emerging threats and their location in cyberspace (and quite possibly the real world).



RULE #4: THINK BEYOND THE ENTERPRISE

Remember that network cognizance goes beyond the enterprise view. The right platform will combine the visualization, identification, and analysis capabilities that you need to patrol cyberspace beyond your boundaries. By looking at the big picture and matching events across disparate data sets, you gain the ability to see beyond your own company's perimeters, enabling you to better monitor incoming and outgoing traffic, locate and mitigate previously unknown or unseen risks, and respond to potential attacks faster.

RULE #5: KEEP YOUR DATA OPTIONS OPEN

Any information that is useful for establishing network awareness is a possible data source. Unlike traditional tools that required time-consuming code changes to incorporate new data formats, the platform you select should have the ability to handle new data sets quickly and easily. By combining data such as BGP, network flow, packet capture, and IDS/IPS data into one complete picture, connections can be made in seconds instead of weeks. A system that allows virtually any data type opens new avenues of analysis never before realized, and provides a more complete network picture.

RULE #6: DEMAND YOUR FREEDOM

You need freedom to work on new and unique methods of network analysis to keep ahead of evolving threats. The ultimate goal of your analysis platform should be to improve the response to all types of threats, pinpoint troubleshooting areas and manage defensive asset placement. The only way to improve threat response with the huge amounts of data is to automate the ingesting, processing, merging, correlating and presentation of the data.

RULE #7: KNOW WHAT YOU DON'T KNOW

An often-overlooked part of a well-built analysis system is a method of reporting the completeness of consumed data sets. A common phrase within the intelligence community is applicable here: “Know what you don't know.” Simply acknowledging that there is missing information is a helpful part of analysis. Any system should have the capability to compare data sets, report their completeness and what data might be missing. For example, information about IP address allocation by region, unregistered autonomous system announcements, conflicts between regions where prefixes are registered and where they are being announced make up valuable information sets. Commonly an analysis tool is so focused on data loading and querying that it overlooks the importance of analyzing how much value the data sources add.

RULE #8: LOOK FOR CORE BUILT-IN FUNCTIONALITY

A well-designed system will have many features to help you quickly determine how vulnerable your network is to cyber attacks and allow you to take steps to mitigate or automatically respond. Make sure your platform offers:

- Simple, accurate navigation of IP networks
- Automated analysis and intuitive interface
- Monitoring and visualization of global Internet access, connection points, routing and topology
- Tracking Botnets, phishers and spammers, worm and virus propagation, and other malicious traffic
- Displaying network flow and other network connection information

- Collecting and analyzing global network data from multiple sources
- Understanding of the enterprise footprint in relation to the Internet
- Optimizing security and network management tools
- Correlating global and local events
- Continually improving analytics and sharing

RULE #9 – MONITORING AND ALERTS MUST BE IN REAL-TIME

Responding quickly to emerging threats can be difficult with large data sets, simply because of how much data has to be processed. Networks connected to the Internet are producing and processing billions of packets of data daily. As the rate of traffic increases, systems doing analysis need to be able to process these large data sets quickly, while simultaneously delivering results. Providing alerts in real time is critical for a system that addresses threats as they happen, and allows analysts to stay on top of network problems and changes.

RULE #10 – LEARN TO EVOLVE

A useful analysis system should evolve with the changing cyber landscape. It's essential that this system be designed with flexibility in mind and a modular architecture that allows it to scale quickly and easily. You'll be freed up to improve analysis techniques and keep up with the growth of the Internet. The ideal application enables analysts to discover new cyber security events and analysis methods and feed those techniques back into the system. Planning for the unexpected helps define how a system is designed and prevents it from becoming obsolete upon deployment. In order to stay relevant, an analysis system has to be flexible so that new data types and relationships can be integrated quickly without impacting analysts.

CONCLUSION

Network security challenges can by summarized by the four V's – Volume, Velocity, Variety, and Veracity.

An effective analysis platform should handle the volume of data by organizing, aggregating and handling duplicate data efficiently. It should manage the velocity of data by consuming it rapidly, allowing for quick queries and providing related change alerts in near real time. It must handle the variety of data by avoiding the trap of building a system that can only handle a limited range of data types. And finally, your system should manage the veracity of data by automating as much analysis as possible, while allowing you to augment data sets with manual analysis results.

Networks of all sizes are presented with these challenges daily, and the exploding growth of the Internet has made effective and timely human analysis impossible. A platform that addresses

these core challenges is best positioned to handle all aspects of data analysis, and in turn, help you achieve complete cyber situational awareness.

FUTURE CAPABILITIES FOR SITUATIONAL AWARENESS – SCOUTVISION™

Lookingglass ScoutVision™ provides a dynamic view of the world's enterprise and Internet activity. By fusing data from various proprietary sources and partnerships, ScoutVision™ is the only solution that provides analysis and visualization of logical (IP routing), physical (geo-location) and transit medium (fiber, satellite) topology - enabling cyber professionals to accelerate analysis, and improve responsiveness and effectiveness.

ScoutVision™ Features:

- Advanced Monitoring and Visualization
- Internet / Enterprise Routing and Infrastructure Analytics
- Malicious Activity and IP Threat Intelligence
- Geo-Selection and Network Health / Stability Dashboard
- Collaboration, Alerts, and Warnings
- Extensible Data Fusion and Analytics
- Flexible and Scalable Solution

ScoutVision ™ Benefits:

- Monitor Internet and Enterprise topology, routing and connection points
- Navigate Internet-to-Enterprise perspective
- Predict impact from Natural Disasters, Outages, and In-climate Weather
- Track assets infected and controlled by global malicious activity
- Optimize assets and leverage existing security / network management tools
- Improve analytics and sharing
- Find results faster and make accurate decisions

In addition to core functionality, ScoutVision offers features unique to Enterprise Security Management, Critical Infrastructure Protection, Cyber Defense, Cyber Intelligence, and Investigative Analysis applications:

ScoutVision™ Enterprise Security Management

- Enhancement of IT governance and risk management compliance efforts
- PCI assessment preparedness, HIPAA and Sarbanes-Oxley compliance
- Link to Security Information & Event Management (SIEM)
- Prevent security breaches and data leakage

ScoutVision™ Critical Infrastructure Protection

- Weather overlay
- Natural disaster overlay

- Major outages overlay
- Network weather/health reporting

Cyber Defense

- Botnet, malware and phishing monitoring
- Link to Security Information & Event Management (SIEM)
- Threat reporting

Cyber Intelligence

- Intelligent management assets
- Intel access point overlay
- Identification of when major routes of interest change to effect management assets or access points
- Oversight compliance reporting
- Access status reporting
- Attribution assistance

Investigative Analysis

- Taps and tap access point overlay
- Identification of when major routes of interest change to effect management assets or access points
- Warrant compliance reporting
- Access status reporting
- Forensics evidence reporting
- Attribution assistance

ABOUT LOOKINGGLASS

Lookingglass delivers the industry's first Internet-to-Enterprise network intelligence platform, offering a real-time virtual, physical and contextual view of the global Internet extending into the enterprise. Our solutions support a wide range of government and commercial applications, including critical infrastructure protection and network security.

Lookingglass empowers cyber professionals to gain insight into potential threats and makes it possible to accelerate analysis, improve decision-making and inform correct action in real-time. Learn more at www.LGScout.com.

Lookingglass © 2009 All rights reserved.