2012

Cyberprints: Identifying Cyber Attackers by Feature Analysis

Benjamin A. Blakely
Iowa State University

Cyberprints: Identifying cyber attackers by feature analysis

submitted to Iowa State University

by

Benjamin A. Blakely

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Program of Study Committee:
Doug W. Jacobson, Major Professor
Thomas E. Daniels
Mani Mina
James M. McCormick
Steffen W. Schmidt

Iowa State University

Ames, Iowa

2012

ACKNOWLEDGMENTS

Reaching this point in my education is hardly something for which I can claim the credit. For the past twenty-six years I have been encouraged, cared for, counseled, and educated by many truly amazing people. To list them all would require much more space than can be allotted here. In that way, this is both the easiest and hardest part of this document to write. However, I wish to highlight the following people who have stood out as enduring influences and to whom I credit any past or future successes otherwise attributed to me.

First and foremost, I owe an eternal debt of gratitude to my parents, Bill and Denise, and all the rest of my family and extended family, who have been role models for everything I hope to become. Their generous support, whether it be emotional, financial, or just being willing to encourage me in all the many hobbies and activities I went through as a child, have instilled in me the eagerness to learn and willingness to do what it takes to reach for success. I have now been fortunate to add to this the wonderful extended family of my wife. My grandmother, Velma; grandfather, Paul; and uncle, Fred — all educators — are unfortunately not here to share this milestone with me. However, their example of the sanctity of education lingers with me. Additionally, my grandmother, Rose is not here to see the completion of this work, but her encouragement and interest in everything I've done all the way to the end will always stay with me.

Many educators have inspired me, and yes, put up with me, over the twenty-one years I've spent in school. From my kindergarten teacher, Janet Leonard, who gave me and so many others the foundation for a successful career, to teachers like Diane Silvers, Marian Houseman, Bruce Bennett, and Brenda Yoakum, who allowed me the latitude to be myself, tolerated my non-conformity in the classroom, and pushed me to succeed in whatever ways I chose. Had I

not had such caring and passionate people as a part of my educational career, I believe I would not have the insatiable curiosity that drives me today.

At Iowa State, I have encountered a number of instructors and friends who have kept me on the path that has led me to finish this work. In particular, I thank Doug Jacobson, for all of the opportunities he has allowed me to participate in and the many years of support; and Mani Mina, for his contagious passion for learning and all the moral support and guidance he has offered me through my most difficult semesters. Without them, I don't know that I would have finished even a Bachelor's in Computer Engineering, not to mention anything further. Additionally, I thank the other members of my program of study committee — Tom Daniels (whom I have shared many hours with in cyber defense competitions), James McCormick, and Steffen Schmidt — for all of their input and guidance during the completion of this work.

Many others have served as mentors and friends along the way. My friend and supervisor at the Krell Institute, Nazanin Imani, has always been willing to listen to me and lend her wise advice. My friend and colleague, Nate Evans, took me under his wing as a freshman and we've formed not only what I believe to be an unbeatable pair in the workplace, but a close and enduring friendship.

And of course, without the unending love, patience, and encouragement of my wife, Afton, I could not have even dreamed of dedicating the many hours to my schoolwork that would have otherwise been spent with her.

People sometimes ask why I sign my name with a lowercase "B" in my first name. I have signed it this way since I realized in high school that all of those things which I undertake and "leave my mark" on are only in small part due to any ability I might have. By signing my name 'benjamin A Blakely', I emphasize the significance of my family and all of those who have been like family to me in making me who I am.

Alor ergo sum.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

Deterrence, whether it be to counter cyber attacks, nuclear strikes, or playground bullies relies upon a solid and demonstrable understanding of where an attack originated. If an attacker knows his identity is unlikely to be determined, any threats made against him will fall on deaf ears. The current global state of cyber security is one that routinely makes the news as acts of cyber espionage, cyber crime, and cyber warfare close the gap in commonality with their conventional counterparts. In any of these domains, attribution is without a doubt a critical factor. This dissertation will lay the groundwork for this topic by analyzing the history of cyber warfare, and demonstrating why deterrence is the superior, and perhaps only, way to counter it.

Of course, great deal effort has been placed into discovering new ways to determine the origin of attacks in cyberspace. Particularly within the decade from 1995 to 2005, there was something of a genesis of this topic in the literature. While many of these methods proved useful, many failed to find applications beyond the laboratory. The same set of assumptions is used for many of these efforts — that a forensic analyst will be able to obtain, store, and analyze whatever information is necessary for attribution (omniscience), that she will also be able to place sensors in any place necessary to conduct attribution — perhaps even by covert means (omnipresence), and that the choice of location for such systems will be correct for attributing any attack (*a priori* positioning). A survey of the literature in this area is performed to show how these assumptions are built into existing methods and why a new strategy is needed.

The problem of determining authorship is not unique to cyberspace. It is also not a new problem. It has been known for at least a century that forensic analysis of documents can reveal behavioral and linguistic signatures of the author, that the author expressed without

realizing it. Whether it be handwriting, vocabulary, or even the choice of paper, many features can be used to say whether two paper documents share authorship. Such methods have been used extensively in literary analysis under the name Stylometry.

Similar methods have been applied to electronic mail, forum postings, software source code (including viruses), and other forms of electronic communication (Morse Code operators were even know to be recognizable by their cadence) with much success. The central question of this dissertation is whether similar methods can be applied to network-level feature analysis.

To this end, a number of network-level features are derived from the IPv4 and TCP headers of a dataset generated for this purpose. First, using a Multivariate Analysis of Variance (MANOVA), then a Principal Component Analysis (PCA) and visual inspection of feature distribution histograms, and finally a Kolmogorov-Smirnov comparison of the individual feature distributions, these features are evaluated for discriminatory power.

The major contribution of this work is a foundation for a new branch of network forensics - *Cyberprints*. Just as a fingerprint can identify a burglar, the materials in a bomb identify a bomb maker, and features of a fire can identify an arsonist, network-level features in traffic streams show promise for attributing actions in cyberspace. This is accomplished without making the assumptions of existing attribution methods, using relatively simple computations, and considering a small final list of useful features. It is hoped, however, this work will be extended to develop new methods to generate and analyze *Cyberprints*, and find new features to use in such analyses.

For the unfamiliar reader, a glossary of technical terms used in this dissertation is included before the bibliography.

CHAPTER 2. CYBER DETERRENCE

2.1 History of Cyber Warfare

Exploitation of vulnerabilities in cyberspace is hardly a twenty-first century concept. As long as nodes on the Internet have been able to communicate with each other, there have been those willing to take advantage of them. What was perhaps the oldest known widespread cyber attack was not even intended as an attack. In 1988, Robert Morris, a student at Cornell University, wrote a program to count the number of nodes on the Internet. To do so, it used known system vulnerabilities to replicate itself from machine-to-machine. In this regard, it was largely successful, having reached an estimated 10% of the Internet - 60,000 nodes. However, an unintended side effect of its exploitation method caused it to also crash the systems it penetrated, leaving a trail of destruction in its path. Morris resulted in the first conviction in the U.S. under the 1986 Computer Fraud and Abuse Act, despite its benign intentions (Moore, 2008). The concept of self replicating software has since developed into the malicious codes now referred to as worms.

In 1994, one of the first international cyber incidents to gain notoriety occurred at the Air Force Rome Lab. At least 150 intrusions were detected there by system administrators, eventually traced to an Israeli. Unlike Morris, no damage occurred and there were no Israeli laws at the time that made the action a crime, so the perpetrator escaped punishment (Beidleman, 2009). This highlighted the need for international coordination of criminal law and agreements for cooperation to apprehend and prosecute offenders, a topic that will have a more thorough treatment later in this dissertation.

To keep from being caught completely by surprise, good system administrators must view their systems from an adversarial perspective. It was just such an exercise by the U.S. National

Security Agency (NSA) Red Team in 1997 that led to an embarrassingly effective lesson in over-confidence. During operation Eligible Receiver, simulated attackers were able to take control of computers in the command center of the United States Pacific Command (PACOM), and in power grids and 911 systems of nine major U.S. cities (Beidleman, 2009).

In a 1998 event, later named "Solar Sunrise", over 500 U.S. Department of Defense (DOD) computers were found to have been compromised by an unknown attacker (Beidleman, 2009). A year later, an event named "Moonlight Maze" resulted in the breach of hundreds of computers at the U.S. National Aeronautics and Space Administration (NASA), the Pentagon, the Department of Energy (DOE), and other universities and laboratories. Information stolen included technical research, contracts, encryption techniques, and information on war-planning systems (Adams, 2001; Moore, 2008).

In 2003, what were believed to be Chinese attackers began infiltrating classified U.S. networks at an alarming rate in a operation named "Titan Rain" that is still under a veil of secrecy (Moore, 2008; Thornburgh, 2005). In 2007, a penetration so great it has been called "our electronic Pearl Harbor" compromised computers in the DOD, DOE, Department of State (DOS), and Department of Commerce (DOC). The amount of information exfiltrated was sufficient to fill the Library of Congress (Habiger, 2010). That same year, a trove of classified data was found on peer-to-peer (P2P) networks such as Limewire. This included a diagram of the U.S. Secret Internet Protocol Router Network (SIPRNet), password change scripts for the Pentagon's SECRET network, encryption certificates allowing access to contractor systems, the Pentagon's information technology (IT) threat response plan, and threat assessments for multiple U.S. cities (among other data) (Habiger, 2010). In 2008, an attack suspected to originate from Russia crossed into a classified network, creating enough alarm that the President of the United States was personally briefed (Habiger, 2010).

The worst-case scenario in cyber warfare is an adversary that has the capability to paralyze a victim in cyberspace, especially if the victim has a high dependency upon it. In April 2007, the small country of Estonia was perhaps the first victim of an all out cyber barrage. While one of the smallest former Soviet-bloc countries, Estonia has a technological base to rival many

larger countries, and is one of the most "wired" nations in Europe (Beidleman, 2009). This net-centricity became an Achilles' heel when Estonia sustained what was calculated to be a 90 megabit per second (Mbps) onslaught of traffic for over ten hours (Moore, 2008). This overwhelmed the networks and systems of the Estonian government and finance sectors, and brought the country to a standstill. The attack originated from a network of software robot implants (botnet) spread across compromised systems in seventy-five countries, but appeared to be controlled by nodes in Russia. In fact, Estonia (and much of the rest of the world) blamed Russia (Beidleman, 2009), but there was no solid evidence to link the attack to that country and, thus, insufficient casus belli for a retaliation.

Again in 2008, Russia took heat when a series of cyber attacks took out many of the government websites of the country of Georgia. These attacks occurred immediately before a Russian air strike and clearly created a favorable battlefield for the Russian invasion. A botnet was again found to be the method of attack. This time, instead of just choking out legitimate traffic altogether, all traffic to and from Georgia was redirected through Russia (Beidleman, 2009). This amounted to the first known cyber blockade and drew a great deal of international condemnation. But once again, it could not be shown conclusively that the Russian government was the perpetrator. Moscow placed the blame on a non-governmental pro-Moscow activist youth group.

2.2 Difficulty of Defense

Network administrators and information security professionals are faced with an extremely hostile environment. They are unable to classify the attackers, or even if they are attacking, with any great accuracy. The nature and identity of attackers and attacks are constantly in flux. A large number of vulnerabilities are unknowable to the administrator, and a single attacker can cause damage disproportionately large compared to the resources available. On top of all of this, systems often cannot be secured to the level necessary to fend off all attacks because to do so would be to reduce their usability. Regardless of one's opinions about "cyber-security hype" or a "cyber-industrial/military-cyber" complex, these facts are axiomatic.

2.2.1 Lack of clear borders

Cyber territories do not always align with geo-political borders. In fact, many times they do not even have clear boundaries. The U.S. DOD maintains an extensive global network named the Global Information Grid (GIG). This network carries sensitive information required for the successful execution of military operations around the world and clearly represents a valuable asset. However, to maintain a global presence is to cross over many geo-political boundaries. The GIG makes uses of strong encryption to ensure confidentiality, but often the links and nodes that comprise it cross jurisdictions and are controlled by non-governmental parties.

One might make the case that the U.S. DOD has a clear interest in claiming cyber territorial privilege over any link or router that carries information for the GIG. After all, in a large part the Internet consists of information — a purely abstract concept. However, if an attack on the GIG originates from a foreign country, traverses one, or terminates in one, does this fall into U.S. jurisdiction?

Take a simpler case, a trans-Pacific submarine cable carrying this same information. It is difficult to argue that while traveling on networks inside U.S. territories, this traffic is not under U.S. jurisdiction. But how about a cable crossing from Los Angeles to Tokyo? At what point does the traffic leave U.S. jurisdiction? Do we discount the link because of its high speed and merely classify either endpoint as a basis for jurisdiction? What if a submarine vessel places a tap on the cable deep under international waters (as the U.S. did during the Cold War). Does jurisdiction now depend upon the origin of the traffic? The destination?

Since the UN Laws of Armed Conflict base their rationale for retaliation largely upon territorial integrity (Beidleman, 2009), this is more than just a technicality. If a state cannot prove that its sovereignty was threatened by an aggressor, a retaliation risks running afoul of international law regarding the use of force (if, indeed, cyber attacks constitute a use of force).

Similarly, network administrators who wish to secure their networks by only allowing access from trusted states will have quite a difficult time doing so. Even without any intent to disguise his identity, an attacker's Internet Protocol (IP) address might be difficult to track to a specific country or city. It is likewise difficult to generate a list of "all addresses from Great Britain"

versus “all addresses from Iran”.

The Internet Corporation for Assigned Names and Numbers (ICANN) is responsible for delegating IP addresses to regional Internet registries (see Figure 2.1), who assign blocks of addresses to internet service providers (ISPs). A multinational ISP might have a network that spans geo-political borders or occasionally move network blocks between locations. The regional authorities might release blocks of IP addresses, allowing them to be reassigned to a different region entirely!

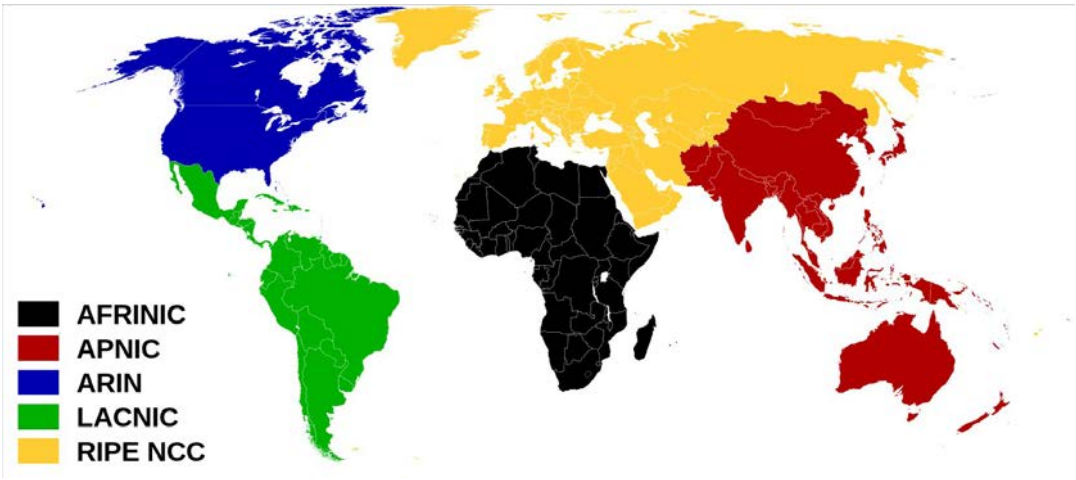

Figure 2.1 Regional Internet Registries

Granularity more fine than country level is even worse in many places. Since an ISP might span the entire country and keep poor records (or none) of which address blocks are used where, it might simply be impossible to determine an accurate location for an IP address. Third parties, such as Google, have made use of open wireless access points to catalog IP addresses they can detect in various areas. This information is sold into large databases, such as MaxMind, and made available to security professionals, but must be constantly updated. Access to these databases is often cost-prohibitive to administrators of smaller networks, leaving them with only coarse estimates of geographical locations (such as the location of an ISP’s corporate headquarters).

2.2.2 Asymmetry

The interaction between defenders and attackers in cyberspace is highly asymmetrical. A network administrator must be constantly on guard, and make a large investment in software and appliances to secure the network from attack. Information systems are highly complex, and understanding of their internals and configuration is often highly compartmented. In many large organizations, a Chief Information Security Officer (CISO) must delegate security monitoring of various systems to subordinates who are experts in those subject areas. In larger networks, it might not be possible for a single network administrator, or even a small team, to keep up with the constant stream of possible vulnerabilities to defend against.

For this reason, cyber defense requires a large investment of time, resources, and manpower to achieve an acceptable level of security. On the other hand, if even a single flaw is left unpatched - it is an Achilles' Heel. If 1 out of 1000 servers did not receive the latest patch to its web server software, that might be all it takes for a lucky attacker to gain entry into a company's systems and do further damage. Conversely, an attacker needs only a cheap computer and Internet connection, and some time. Tools to automate the detection and exploitation of vulnerabilities are widely available, often in the public domain (system administrators need access to these to test their own security).

Simply put, a seemingly insignificant attacker can overcome huge investments in defense by finding a single flaw - which is not difficult. Defenders must be constantly on guard, monitoring every entry point to the network, keeping all systems up-to-date, and keeping abreast of new developments in both offensive and defensive technology. Short of 9/11-type events, this is the only domain in which an attacker has such extreme leverage.

2.2.3 Dynamic nature

Keeping abreast of new developments in offensive and defensive technology is literally a full time job in many organizations. This is in addition to the fact that the very fabric of the cyber domain is constantly in flux. The "consumerization of information technology" has employees introducing entirely new classes of devices and applications to networks that

security professionals must account for in the organization. The ongoing transition to IPv6 (the next version of the backbone protocol of the Internet) has dragged on for over a decade, largely because system administrators are not comfortable with the security ramifications of this switch. This is to say nothing about the constant evolution of the skills of attackers.

Indeed there is an arms race between cyber defenders and attackers. One need only to open his email inbox to see this playing out in real time — network administrators are waging a continuous battle against email spammers, who constantly find new ways to evade filters. A system that was "secure enough" yesterday might become the "low hanging fruit" of today. All it takes is for someone to release a tool or exploit code into the wild (information security jargon for widespread public distribution), and anyone with a compiler can now break into a system. A recent example of this phenomenon was the Debian SSL vulnerability in 2008 (DSA-1571-1, 2008). It was disclosed that the random number generator used to create the certificates used to encrypt web traffic (among other things) was predictable. The very protocol system administrators thought was protecting them turned out to be a vulnerability!

2.2.4 The unknown

A fact life for network administrators is that one simply cannot know what one doesn't know. For this reason, best practices documents and configuration guides run amok — at least that way well-read network administrators can say they tried. But the brutal fact of the matter is that no matter how much effort one puts into defending against the attack vectors of yesterday, new attack vectors constantly appear. Anti-virus vendors are aware of this, and thus do not rely solely on signature-based detection, but also use heuristics. They make an attempt to classify nefarious executables not by what they look like, but by what they do. Security professionals even have a special category set aside for attacks that make use of a previously unknown vulnerability — zero-day exploits. This term is derived from the idea that one can count the days from when a vulnerability is known until when a method of taking advantage of the vulnerability (an exploit) is made available.

Zero-day exploits are those that occur on the "zeroth" day - computer science parlance for

the first index in a list. By their definition, zero-day exploits cannot be foreseen. Usually, these make use of a flaw in a piece of software running on a system. For example, many programs have been found to be vulnerable to buffer overflows, whereby an attack is able to overflow the memory allocated to hold user input. This overflow allows the attack to place malicious code outside of the heap and in the stack; in other words, outside the data of a program and in the program itself. By using techniques such as NOOP-sleds, a series of instructions that tell the processor to simply continue to the next instruction, an attacker can maximize the chance of tricking the vulnerable program into running the malicious instructions instead of its normal execution sequence. Recent programming languages have better bounds checking on user input and memory access, but these flaws continue to be found at an alarming rate.

A network administrator cannot possibly know the intricate workings of every system she operates. Just using a web browser means having faith in millions of lines of code written by a number of other individuals. An administrator who is in charge of a myriad of systems, each with their own unique software applications and configurations (not to mention the operating systems and infrastructure) has no choice but to trust the authors of the chosen software. Therefore, it easy to see the great chasms in the understanding of the knowledge of the workings of systems under an administrator's care.

Assuming one does trust the authors of the software, which one has little choice but to do (save federal agencies which require extensive code review for applications placed onto sensitive networks), one must monitor the many channels of information providing notices of new vulnerabilities and patches to each and every piece of software. Clearly, this can become overwhelming. Add to this the fact that entirely new classes of attack vectors are being discovered. For example, there has been much research lately into hardware trojans (e.g., Chakraborty et al., 2009; Tehranipoor and Koushanfar, 2010; Jin et al., 2009) - malicious agents that run not within the operating system of a computer, but within its very brain! Until this class of malware began being researched, network administrators did not know this was something they needed to worry about. However, one is given to wonder to what extent more advanced cyber attackers, such as nation-state level forces, were aware of and making

use of these technologies.

Network infrastructures are also of greater concern than ever. While there has always been much trepidation about questionably legal law-enforcement monitoring of communications (and, it should be said, this is not just limited to the digital age), questions have begun to surface about larger scale, more clearly nefarious monitoring. While it is general knowledge that China censors and monitors much or all of the Internet traffic of its citizens, recent events have raised questions about whether they might be going even further. In 2010, a large share of global Internet traffic was re-routed through China due to an incorrect modification of Internet routing tables (Hijacking, 2010). While China claims this was unintentional, it shows that if a state really desires, it can exert great influence over the communications of millions (and not just those within its borders).

2.2.5 Third-parties

In addition to the reliance upon software engineers and programmers outside of one's own organization, an increasing unknown for network administrators comes in the form of partnerships with service providers. As more and more information moves "into the cloud", network administrators have less and less control over the security of their intellectual property. Google has been widely criticized (though most vocally by Microsoft) for not taking seriously enough the security concerns of its clients. To avoid this same mistake, companies, such as Verizon, have created purpose-built data centers for federal, and even classified, information. However, concerns about the integrity, confidentiality, and availability of data still are a stumbling block for many organizations wishing to gain the many benefits of cloud-based services. A network administrator must ultimately place her faith in these service providers, and trust that they will protect information and services to the same extent she would herself.

Third-party relationships are not just limited to cloud-based services, however. Any time a packet is sent across the network, it traverses the systems of a number of independent network operators. While availability concerns are minimized by the survivable nature of IP (specifically, the transmission control protocol - TCP), one still must place a great deal of trust

in these providers. As the "network hijacking" mentioned above illustrates, even the biggest network providers are susceptible to error, if not intentional manipulation. One can make use of transport encryption to make it difficult to intercept transmissions, but an attacker with sufficient resources or knowledge of a vulnerability, such as the Debian SSL weakness mentioned above, might still be able to gain information from these ostensibly protected traffic streams.

2.2.6 Usability

Most security professionals will attest that one of the biggest challenges of securing systems is doing so while maintaining a user-friendly interface. A statement so common in security circles it has become somewhat trite is "the only secure system is one that is unplugged and locked in a safe". Indeed, by making a system accessible in any way, one potentially opens it up to attack. Security professionals must fight the stereotype that they exist only to make the lives of "ordinary employees" more difficult. This author has heard pejorative terms such as "profit prevention department" applied to information security departments on more than one occasion. Indeed, it seems any action taken to secure a system must necessarily make it more difficult to use. Security adds layers of authentication and authorization that users must navigate. A system designed purely for usability would be one in which a user simply directly performs the actions required to achieve the task at hand, without any ancillary interactions. One might think of this as the polar opposite of the computer in a safe. Such a system would have no way to ensure a user is who he claims to be, and is authorized to perform the requested action.

Biometrics, or authentication based upon biological traits of the user, are a potential solution to this, but accuracy and affordability of such systems have yet to make these feasible for everyday use. It is now more common to see laptops with built-in thumb print scanners, but anyone who has used one will attest that they often go unused because of the frustration of having to swipe one's thumb on the sensor several times before it is correctly identified. Even if perfect, such systems still require extra interaction from the user. There are potentially transparent methods, such as facial recognition or embedded identification transmitters, but

such systems raise concerns about privacy that many users are not willing to ignore.

Information assurance is usually defined as protecting the confidentiality, integrity, and availability of information (and there are a number of suggested "fourth elements" such as non-repudiation). However, it might better be defined as working to find a balance between these requirements and usability. While information security position advertisements often list requirements of knowledge of applicable regulations and systems, it is increasingly common to also see them mention knowledge of business operations. Without a practical awareness of what users require to perform their jobs efficiently and pleasantly, security is a lost cause. Post-it notes with passwords, copying data to external drives, emailing sensitive documents, Internet web proxies, users will always find a way around security measures if they are not able to use the system in the way they desire. Therefore, it is self-defeating for an information security professional to foist draconian policies upon system users. The more restrictive they become, the more users will try to find ways around them. But without strict adherence to security principles, network administrators potentially leave open avenues of attack against their systems.

2.2.7 Detection

Another large unknown for network administrators is knowledge of if and when they are being attacked. In all other domains of warfare, it is (painfully) obvious when one is under attack. Radar systems can provide preemptive warning of many types of aircraft, missiles, ships, and satellites; port authorities and border patrols can monitor for unauthorized territorial incursions. If an actual attack occurs, one immediately shifts into triage and recovery mode. To date, cyberspace attacks are largely more analogous to espionage than armed combat. However, as will be discussed later, the transition between the two can happen instantaneously in cyberspace. Whereas a certain tolerance and looking the other way goes on in the traditional game of espionage, this is less tolerable in cyberspace. Therefore, automated systems able to detect incoming attacks are required. Humans simply cannot observe, analyze, and react at the speed required for a cyber skirmish.

Despite years of research, intrusion detection and prevention systems are still far from perfect. They require a huge investment of time to set up and calibrate to avoid constant false alarms, but with every reduction in false alarms comes the potential for missing an actual attack. Methods to detect attacks in-progress have evolved from simple IP address blacklists, to signature-based analysis, to heuristic analysis (which mirror, in some ways, the detection of viruses on workstations). Most intrusion detection systems rely upon signature-based analysis, where certain sequences of network traffic trigger alarms. This does little to detect zero-day exploits. It will also miss many of the tell-tale signs of an impending attack — network scans. But to allow detection of these scans is to invite a deluge of alerts, as the "background radiation" (Zimmerman et al., 2005) of the Internet is rife with automated scanning traffic.

2.3 Why Deterrence?

One might ask why deterrence is necessary at all. After all, there is no cyber equivalent of a nuclear attack, where the target finds itself utterly unable to mount any effective defense. Cyber warfare is much more subtle and nuanced than that. As discussed elsewhere in this dissertation, it is often more comparable to espionage than to armed combat. However, after having explained many of the reasons cyber defense is so hard above, hopefully this question is somewhat clearer.

2.3.1 Justifications

It has been explained how one cannot possibly hope to defend every potential avenue of entry by his adversaries. Even if one has an unlimited budget to hire security professionals and purchase security software and appliances, there will still be the obstacles of proprietary confidentiality, poorly-written code, unforeseen consequences of even well-written code, and of course the human element (Evans, 2009).

Put simply, eliminating potential vulnerabilities is not an option. This is not to say that network administrators should not take every precaution and implement best practices such as least-privilege, isolation/compartmentalization, regular update cycles, and regular vulner-

ability assessments. If it were not for these practices, network administrators would likely find themselves constantly ensnared in constant low-skill dragneting by every teenager with an Internet connection. Therefore, if one cannot close all doors to potential exploitation, one can at best hope to deter the enemy from walking through them.

One cannot enumerate all potential adversaries in the cyber realm. It is then not possible to calibrate defensive systems to even detect, let alone prevent, all possible attacks. It is impossible to avoid talk of the Advanced Persistent Threat (APT) in security circles. Such actors are well-funded, well-trained, and determined. Often they take the form of a unit within the armed forces of a state.

While most intrusion detection systems are designed to look for patterns of traffic known bad within single traffic streams, or perhaps within the last few hours or days of traffic, these attacks could entail years of research and development by the attacker. The infiltration of the Bushehr nuclear centrifuge facility in Iran is suspected to have undergone years of highly-funded, highly-collaborative, and yet highly-secretive development to tailor it precisely for the cyber environment of Bushehr. Network administrators at all levels are simply not capable of detecting and mitigating attacks of this level of sophistication.

Even if one can detect an attack and attribute it to an attacker, the chances of successfully bringing prosecution against the attacker, let alone winning a verdict, are hopelessly slim. In the best case scenario, an attacker directly attacks a well-prepared domestic target with little or no attempt to hide her identity. In this case, domestic law enforcement can use the audit information from the victim to look up the IP address of the attacker, subpoena the ISP for the identity of the user attached to that address at the time of the attack, and bring a sufficiently convincing civil or criminal case to win a guilty plea from the attacker. However, there are a number of assumptions here.

First is the assumption of a well-prepared defender. While cyber defenses have likely improved among network administrators at all levels in recent years (due to heightened awareness, and enrollment in training programs and degrees), there are still undoubtedly large numbers of network administrators who simply "plug and play". This is reflected in the Internet Crime

Complaint Centers' 2010 Annual Report (Center, 2010). While regulations exist for the protection of health care information, records of publicly-traded corporations, financial information, military secrets, and payment card information; much of cyberspace remains a "wild west". Without regulations, and audits to enforce them, many network administrators never have cause to investigate the security of their networks. Thus, these networks remain only as protected as the default installation parameters allow. Such network administrators, if they even have audit logs of activity on their network, might not be able to access them, or might not keep sufficient history to be able to track down an attacker after the fact.

It was entertaining to many in the security industry to learn that the attacks perpetrated by the Anonymous group against the likes of Visa and Mastercard in the wake of their termination of accounts related to Anonymous funding were in fact not in the least bit anonymous (Pras et al., 2010). The tool of choice for this group was the Low Orbit Ion Cannon (LOIC), a tool which accepts commands from an Internet Relay Chat (IRC) network (or can be run manually) to work in concert with other machines running the software to attempt to overwhelm the network connections and servers of targets. However, the tool does not make any attempt to mask its IP address, making it trivial for targets to block and attribute attackers. But this is the exception, not the rule. Most tools come with at least one way to disguise one's identity, and entire applications are built to do just this (e.g., Tor: The Onion Router). While there are legitimate uses of such tools, in particular bypassing the censoring mechanisms of oppressive regimes, they are often used by those who wish to hide their identity online.

In fact any cyber attacker worth his salt will not make the mistake of directly attacking a target. He will first compromise lower-sensitivity targets on which he is unlikely to be caught, and build a chain through the Internet to obfuscate his true identity. This is called the stepping-stone method of anonymization and is the same basic premise used by Tor. In these cases, due to jurisdictional or logistical issues, it might be simply impossible to unravel the chain of systems to make it back to the original attacker.

Jurisdiction is an issue in and of itself. Even if one is able to readily identify the identity of an attacker, she might be unable to prosecute due to the nature of the territory in which the

attacker resides. The foreign state might be simply covering for a state-sponsored attack, but more often privacy, legal, or national identity roadblocks prevent collaboration from taking place. For example, while China might be unwilling to cooperate on the grounds that it does not trust the United States to prosecute one of its citizens, Germany is legally unable to release this information because of domestic legislation regarding the privacy of its citizens. Some countries might simply have no laws which would make the attack illegal, and thus it is not subject to prosecution. One can see how transnational organizations can quickly become entangled in such bureaucratic tugs-of-war, let alone an organization without a presence in the foreign country. Especially if one is simply trying to unravel a chain of stepping-stone's where he cannot even say the information requested is in fact of an actual criminal!

Assuming a cooperative territory and ISP, and a rock-solid attribution to an IP address, one must still account for the fact that the ISP might not keep records of sufficient detail to identify the user of a particular IP address. As most ISPs make use of dynamic addressing for their clients, it is entirely possible that an IP address has been shuffled among a number of different individuals by the time the request for its owner is received. The dynamic host configuration protocol (DHCP) does not have any tracking history, so it is up to the particular implementation of the protocol to extended it to add this functionality. Even if this is done, the information might not be kept very long because of the storage space required.

If all of this is resolved, one must then hopefully elicit a confession from the attacker, and find an applicable statute under which the action is worthy of prosecution. Without an applicable statute, there is no basis for further repercussions and the victim is simply out of luck. As law often lags technology, there are constantly new permutations of cyber attacks which fall into gray areas. Some countries simply have no laws applicable to cyberspace. But even if they do, without a confession the case must still go to court. In this case, the attorneys are saddled with explaining the highly technical details of Internet communications, attribution, cyber attacks, and digital forensics through the use of expert witnesses. Most juries are likely to struggle to understand these facts, and some might even be simply hostile to any mention of cyberspace. Under U.S. statutes a jury verdict must be reached by the

preponderance of evidence (civil trial) or proof beyond a reasonable doubt (criminal trial). To remove even 51% of doubt from the minds of jurors in such cases is no doubt quite difficult, raising the bar for conviction.

2.3.2 Alternatives

It is worth asking if there might be more sure-fire alternatives to deterrence. Deterrence after all, is merely a formal process of hedging one's bets or changing the perceptions of the odds by those being bet against. Like any risk assessment or mitigation process, it is full of uncertainty. One can make it so that only an insane individual would take the chance of attacking, but there are no shortage of adversaries with questionable grips on reality. What then are other ways to overcome the difficulty of defense and why are they not preferable?

First, we return to the unplugged server in the safe. Joking aside, isolation is a practical strategy. Granted no one takes it quite so far, but it is used in many other ways. For example, the U.S. DOD makes use of a number of different networks for different levels of classification. There is the NIPR-net (the non-classified IP routing network), SIPR-net (secret), and a number of networks at higher security levels, such as top secret and sensitive compartmented information. These networks are nominally "air-gapped" from one another, meaning that ideally there is no communication channel between different levels of networks.

This principle almost never holds for practical reasons. Often unclassified databases must be used on a classified network. Perhaps some information has been declassified and must be moved to a less sensitive network. Perhaps automated intelligence gathering systems needs to operate agents in environments not secure enough to operate a sensitive network, but need to feed information into classified systems. Whatever the case, one at a time, breaks are made in this air gap. The most common is "sneaker net", the physical transfer via removable media, photograph, or transcription of sensitive information. This is how Bradley Manning allegedly exfiltrated a large number of secret U.S. DOS cables. But, even users who have no ill will can become frustrated when they are trying to "just do their jobs" and are unable to do something as simple as copy an unclassified document onto their classified system, leading to breaches

of isolation measures. Trying to prevent users from doing so by removing external interfaces, locking down applications, or other technical measures merely escalates the amount of effort users will exert to bypass them.

Critical infrastructure operators claim their networks are entirely "off-grid". After all, it is hard to imagine that a nuclear power plant's control rod mechanism having a web presence will lead to anything good. But increasingly, it is being discovered that Internet links to such "isolated" systems exist, even unbeknown to their operators. Often, this takes the form of a backdoor the vendor installed to perform remote maintenance, but increasingly the networks used in the front office and the control systems are interconnected. In power distribution systems, this potentially exposes a large number of substations and other remote systems to Internet exposure. To return to the example of Bushehr, U.S.B thumb drives were used to trick employees of the plant into loading malicious software onto their network. Isolation is bound for failure.

So, if it is accepted that at some point all systems will have exposure to hostile agents, what other options are there? The most obvious is to return fire. Certainly this is the most dramatic alternative. When under attack, one could counterattack in an attempt to take out the adversary before she can do any harm. However, this is unlikely to be successful for several reasons. Most practically, it is unlikely the defender will become aware of the ongoing attack in time to mount an effective response against it. In the case of an APT, the attack might be a marathon instead of a sprint, but there is likely to be little to set off alarm bells until damage is already in process. Most cyber attacks happen in an instant, and most are not worthy of a response. If one were to respond to every unauthorized attempt for entry into his network, it would quickly consume all of his time (and then some) and, as discussed momentarily, be unlikely to have much success.

The probability of identifying the attacker (including removing any layers of anonymization), enumerating an exploitable weakness in her system, and exploiting it within the time span of an attack is nearly zero. Considers that an APT spends months or years identifying exploitable vulnerabilities and assume that an attacker secures his system against all threats

known to him. This leaves little chance that an exploitable vulnerability is even present, let alone readily identifiable. This requires the defender to discriminate retaliation-worthy targets from harmless annoyances. An SSH server (a remote login service allowing console access to a system) might see thousands of intrusion attempts in a day, coming from hundreds of hosts (speaking from this author's own experience). To retaliate against each and every attacker is certainly impossible, especially considering that the number of intrusion attempts scales with the size of the network.

So, a defender is unlikely able to successfully isolate himself and would waste his effort in attempting retaliations. What is left? Simply put, sit there and take it, but do so in an intelligent way. First, one can make her network sufficiently resilient to attack that even if an attacker is successful in taking down or damaging systems, the organization continues to function. This significantly increases the cost of one's network, but resiliency is likely to be the case anyway due to the best practice of avoiding single points of failure. Resiliency does much to take care of availability concerns and potentially some integrity concerns, and can act as a deterrent in and of itself by signaling attackers that attempts to take out systems are unlikely to be successful. However, it does nothing to address concerns of confidentiality. Even if a database of social security numbers is stolen from only one of ten servers, this does not lessen the impact of it being stolen. In fact, maintaining ten servers actually increases the potential attack surface and requires network administrators to ensure that ten times as many systems are appropriately secured. One must also ensure that replication of data between systems is done in a controlled and reversible way, so if the integrity of data on one system is compromised, this does not also pollute all replicants of that data.

Finally is the attempt to divert attacks from one's systems. The most common method is to set up a "honeypot" or "honeynet", a system or network of systems which serve no purpose other than to attract attackers. By makings these systems sufficiently enticing (but not ostentatiously so), an administrator can encourage an attacker to spend time attacking a decoy system instead of the actual corporate network. Additionally, such systems can be specially instrumented to provide information that can be used to reveal the methods of the

attackers. This can help network administrators stay ahead of the attacker and proactively secure production networks.

While this all sounds good, there are several snags. First is that one cannot be sure attackers will take the bait. Assume that a decoy network will, at best, be as enticing as the real network, and the administrator sets up one decoy network. Then at best there is a 50:50 chance the attacker will pick the decoy to target. While perhaps these networks can be made more enticing by opening additional vulnerabilities and planting fake data, sophisticated attackers are likely to see through such ruses, which might intensify the "attention" the production network receives. If attackers have done their homework, meaning open source intelligence gathering, they will be able to quickly see behind the false front. Something as simple as identifying which network a company's web server is actually running on can defeat the entire decoy.

There also remain legal questions in some jurisdictions related to wiretapping and entrapment. If the decoy network is run by a law enforcement agency (or perhaps a contractor of one), it might be construed as enticement to commit a crime, which might not only exempt the attacker from prosecution, but could land the network operator in hot water. Even if the network is operated by a private network operator, evidence obtained might lack a solid footing in a legal proceeding. Perhaps the attacker knew it was a decoy and, therefore, knew that he wouldn't cause any actual damage? Maybe he wouldn't have attacked it if it was a real network. Perhaps the enticement to attack by intentionally leaving open vulnerabilities with the intent to gather intelligence is sufficient to constitute authorization by the network operator of the otherwise illicit use. Additionally, using information obtained from such networks without explicit consent of the user might run afoul of wiretapping laws. While corporations usually have little restriction on the monitoring of their internal networks, it is usually advisable to have users consent to this ahead of time to avoid any legal issues. If the attacker is not made aware that his actions are monitored, the operator might violate the Federal Wiretap Act, which permits observation of activities on one's own networks only with consent from those being monitored.

Therefore, while deterrence is far from a sure-fire method of preventing cyber attacks, it might be a more effective defensive tactic than the alternatives. Isolation is not likely to last, retaliation is not feasible, resiliency does not cover all the bases, and misdirection seems like somewhat of a minefield. The only option is to prevent the adversary from becoming the attacker — to stop the attack before it begins.

2.4 Classical Deterrence

2.4.1 Objectives

There is an extensive body of literature that can help shed light on the problem of deterrence as a result of the Cold War. However, this idea is much older. In the Peloponnesian War writings (431-404BCE), Thucydides describes ways in which enemies sought to entice each other from starting or expanding a war (George and Smoke, 1974). As with cyber war, the risk of nuclear war presents a terrifying specter for which no one can exactly calculate the outcome. However, in many ways nuclear deterrence is much easier.

The fundamental problem of deterrence is simple and anyone who experienced (or was) a playground bully can understand the basic idea. Deterrence policies seek to affect the behavior of an adversary with the use of calculated threats (Achen and Snidal, 1989; National Defense Strategy of the United States, 2008). This might be used to prevent a direct attack, to secure strategic access and freedom of action globally, to comfort allies and strengthen relationships, to establish favorable security conditions as a hedge in case a need for military action should arise (National Defense Strategy of the United States, 2005), or to protect allies from attack (National Defense Strategy of the United States, 2008).

It is common to think of deterrence as a kind of mental calculus performed by both the aggressor and the defender. The benefits of aggression (or retaliation) sit on one end of a hypothetical seesaw, with the consequences of aggression (or retaliation) on the other. The consequences of restraint (e.g., appearing weak or risking an attack from the intended victim) and risk-taking propensity of the aggressor act as a fulcrum in this balancing act and can be moved closer to one end or the other by external factors (Beeker, 2009; DOJOC, 2006). For

example, a country might know there is a resource they very much desire within a neighbor's borders that could be obtained through an invasion (benefit of aggression), but it also knows that international sanctions are likely to follow if it proceeds with the attack (consequences of aggression). If the international body that would sanction them appears distracted with another matter, it might push the fulcrum towards the consequences, meaning the benefits now appear to outweigh the risks and the country is likely to proceed with the invasion. Conversely, if the neighboring country has recently developed a nuclear strike capability, the fulcrum is shoved the other direction.

2.4.2 Methods

Deterrence relies on the ability to raise the perceived costs of aggression or lower the perceived benefits (Beidleman, 2009). A number of methods can be used. The primary method of raising perceived costs is to make a clear statement and demonstration of retaliatory capability (Beidleman, 2009). To be credible, the threatened retaliation must be both proportional to the deterred action, and clearly within the capabilities (politically as well as logistically) of the defender (Taipale, 2010). The greater the certainty of retaliation, the greater the deterrent effect (Taipale, 2010). It must also be obvious to the aggressor that if they exercise restraint, the resulting outcome will be desirable for them (DOJOC, 2006; Jervis et al., 1989).

Another method of deterrence, which had limited utility in nuclear war, is to demonstrate the futility of attack (Beidleman, 2009). If the aggressor believes that her actions will butt up against strong defenses or otherwise have an insignificant effects, they are less likely to invest time and resources developing the attack. Unfortunately in the Cold War, there was little defense for a nuclear explosion. Civil defense drills, underground shelters, and attempts to shoot down missiles before they detonated were the extent of it, and none were particularly effective.

In nuclear deterrence, the credible threat of retaliation was the primary factor keeping a lid on the pot. Assurances from both the Soviet Union and the United States that any attack would result in severe, swift, and incontestable retaliation kept the peace for over thirty

years (Achen and Snidal, 1989; National Defense Strategy of the United States, 2008). The Nuclear Deterrence Triad — the term for the primary deterrent threats — consisted of long-range bombers, Intercontinental Ballistic Missiles (ICBMs), and Submarine-Launched Ballistic Missiles (SLBMs) (Chilton, 2008). The combination of these three prevented any notion of a single attack able to completely disable the enemy.

Today, deterrence has evolved into multi-actor deterrence in the modern world. However, it is not entirely removed from Cold War-era strategies and does not fully mesh with cyberspace realities. It is a much more complex process, requiring an integration of military powers with diplomatic, informational, and economic (DOJOC, 2006). Global strike capability lives on, but makes more use of forward deployed units in "force projection" missions than on long-range devastation (though it remains an option) (DOJOC, 2006). A large degree of global situational awareness is required to keep tabs on the objectives and values of the many potential adversaries that must be deterred. This requires a high-speed global network (such as the GIG) to enable forward deployments in adversary networks and communication between distant forces working on the same mission (DOJOC, 2006). It must not only deter specific aggressive behaviors, but specific targets of aggression and outcomes of aggression (Taipale, 2010).

Deterrent threats must also be tailored to specific adversaries (DOJOC, 2006), as the threats that influence one aggressor might have no effect on another. For example, the threat of being caught in the act might be a serious concern for a state involved in espionage against an ally, but have no effect upon a state with which relations are strained. Similarly, the threat of lethal force loses its edge when dealing with a suicide bomber. Even within a particular state, deterrence must be targeted to many different levels of decision-makers to ensure that deterrent signals (communications between a defender and aggressor of a retaliatory threat) are properly heard and interpreted (DOJOC, 2006). However, while threats must be sufficiently clear, precisely targeted, and credible to present a real threat, sufficient ambiguity must remain to preserve flexibility of retaliation and prevent precisely calibrated challenges from the aggressor that approach cross retaliation thresholds without crossing them (Taipale, 2010).

A good summary of current deterrence thinking is present in the U.S. Deterrence Operations

Joint Operating Concept (DOJOC, 2006). Here, there is an outline for integrating deterrence into military planning. One must specify the deterrence objectives and strategic context, assess the decision calculus of adversaries, identify desired effects on adversary's decision calculus, develop and assess tailored courses of action for deterring the adversary, and finally execute and monitor the methods for effectiveness. A similar outline applies in cyberspace, but assessing the decision-making calculus of adversaries becomes incredibly difficult when dealing with the small timescales, the large variety of adversaries, and the highly anonymous nature of the Internet.

2.4.3 Problems

Deterrence theory comes with a set of caveats, mostly based in the psychology of deterrence. After all, the reality is that humans are not perfectly logical beings and the world is not as simple as a game of chess. Decision-makers are faced with complex situations with many interwoven factors, including emotions, beliefs, and biases on the part of both the aggressor and the defender (Jervis et al., 1989). The amount of effort needed to fit the mold of a completely rational and logical decision-maker is unlikely to be met. An alternative description is that of Cybernetic Decision Making, whereby decision-makers only marginally adjust immediate actions without a complete picture of the long-term consequences, much as a thermostat makes a nearsighted decision as to whether or not to turn on the air conditioner (Steinbruner, 1976). In any case, deterrence theories must attempt to approximate the thinking of decision-makers, and the effects of deterrent threats upon them.

Compounding this uncertainty is the difficulty of "getting in the head" of potential adversaries. After all, there are a wide range of adversaries in the post-Cold War world — state, criminal, extremist, state-affiliated or encouraged, fundamentalists — all with different customs and behaviors. This makes it very difficult to accurately predict how an adversary might perceive a deterrent threat (DOJOC, 2006). The challenge is to gather enough intelligence about an adversary's perceptions and intentions to at least approximate the process of deterrence calculus, but this is very difficult and it is hard to find cases of even mild international

conflict where both sides accurately made such assessments (Jervis, 1984). When considering the case of irrational actors, such as psychopathic or extremist actors, it becomes nearly impossible to predict how an adversary will respond to a deterrent threat.

Understanding the threat an adversary might present is another challenge. Certain adversaries might make unsubstantiated threats, have a history of a particular kind of violence, or be suspected of ties with known aggressors. However, decision-makers must be careful not to fall prey to the biases of availability and representativeness when assessing these threats (Jervis et al., 1989). The former is the psychological term for the logical fallacy of determining an event more likely based upon experience or knowledge, but not necessarily upon the actual probability of the event occurring. The latter is essentially a term to mean making an error in judgment due to stereotypes or prejudices. For example, assuming an Islam fundamentalist presents a threat of suicide attack regardless of whether he has any history of violent behavior. In fact, the most successful attacks are those for which the defender has not adequately prepared. Attacks, such as these, that come as a surprise usually do so because they are unlikely to succeed and have been downplayed by the defender (Jervis et al., 1989).

Presenting a credible threat to the adversary (or "signaling") is equally difficult. It must be made explicit to the adversary what the stakes are, without providing sufficient information about retaliation capability for the adversary to counter it. It might be believed by the defender that an aggressor has received a deterrent signal, but, due to cultural misunderstandings, the message was lost. Such communication breakdowns can lead to situations where a retaliation is judged as unjustified by the aggressor, because the deterrent threat was not understood (Jervis et al., 1989).

In summary, there is a four fold problem to determine likely attacks. Decision-makers are not likely to analyze a situation in a logical and calculated way, are likely to make biased or misguided estimates of the true threats, run the risk of a signaling miscommunication, and are likely to ignore the threats most likely to turn into an attack. Enemies who believe their stake in a successful attack outweighs the risk (or certainty) of a retaliation, or are otherwise sufficiently determined, will likely ignore deterrent threats (Achen and Snidal, 1989; DOJOC,

2006).

2.5 Cyber Deterrence

With a basic analysis of both cyber warfare and classic deterrence completed, it is important to now examine how cyberspace contorts the ideas of deterrence. Traditional deterrence focused upon a universal deterrence policy to deter all threats (Morgan, 1977). Cyberspace will require deterrence to be tailored for new classes of attackers and attacks, and require new ways of thinking about deterring attackers. Blotzer (2000) characterizes the necessary new deterrence paradigm as focusing on the carrot and not the stick. That is to say that the U.S. can no longer rely solely on the threat of overwhelming, precisely targeted force to deter a single adversary. Instead, the U.S. must focus on what adversarial leadership values most and seek ways to develop enticements for continued peace. The National Defense Strategy of the United States (2008) emphasizes the use of non-military and defensive tactics in this way:

> The same developments that add to the complexity of the challenge also offer us a greater variety of capabilities and methods to deter or dissuade adversaries. This diversity of tools, military and non-military, allows us to create more plausible reactions to attacks in the eyes of opponents and a more credible deterrence to them. In addition, changes in capabilities, especially new technologies, permit us to create increasingly credible defenses to convince would-be attackers that their efforts are ultimately futile.

2.5.1 Attackers and attacks

One of the most difficult parts of cyber deterrence is identifying the correct threats to deter. Often, a threat is unknown until a successful attack has occurred. There is no way to see the enemy preparing for cyber battle, to track the development of cyber arms (without an explicit declaration by the adversary), and no way to know all of the attacks that have been tried but failed (or just have not been detected) (Habiger, 2010). Conventional deterrence relied upon early detection of attacks to give time to mount a retaliatory strike — in cyberspace

there is no such luxury (Beeker, 2009). Small groups of cyber-savvy attackers can amass a disproportionately large attack that would not be possible in traditional warfare (Habiger, 2010) in no time at all.

Considering state-sponsored cyber aggression alone, there are over 100 potential adversaries (Habiger, 2010). Some of the scariest names in U.S. foreign policy are already on the bandwagon. The Russian military has incorporated cyber warfare into its deterrence strategy and recognized the role of information superiority (Sinks, 2008). China is preparing at least a battalion of cyber warriors (Sinks, 2008), has openly discussed goals to dominate the U.S. in cyberspace by 2050 (Habiger, 2010), and in 2008 successfully hacked the White House email (Habiger, 2010). Hezbollah has outlined a strategy of cyber attacks to disrupt Israeli cyber systems (Sinks, 2008). Al-Qaeda computers seized in Afghanistan were found to contain electronic schematics and simulation software of dam control systems (Beidleman, 2009), as well as logs indicating research into U.S. critical infrastructures. Iraqi insurgents were known to use off-the-shelf software costing $26 to hack into the live feeds from U.S. aerial drone's and intercept the video streams (Habiger, 2010). This is not to mention that anyone with access to some cash can hire a botnet to do the dirty work (Moore, 2008). Andrew Palowitch, a former Central Intelligence Agency official now a consultant for the U.S. Strategic Command and head of the U.S. Joint Task Force for Global Network Operations, recently stated that the U.S. Department of Defense has been the subject of 80,000 cyber attacks, some of which have clearly reduced its operational capacity (Habiger, 2010). Cyber attackers should not be taken lightly.

Taipale (Taipale, 2010) places attackers into three primary groups: 1) those who can be deterred directly, 2) those who can be deterred indirectly, and 3) those who are not likely to be deterred. The first group is what classical deterrence is built upon. A single enemy to whom threats of retaliation can be credibly made. The second group is increasingly important in the international spider web of alliances and skirmishes, and non-state groups of attackers. Such groups might be difficult to pin down to threaten directly, but by placing pressure on a known party upon which the attacker depends, deterrence might still succeed. The third

group is the most difficult, and is perhaps the primary group in the current state of cyber affairs. These attackers are either so confident in their anonymity, careless as to the costs of their actions, or devoted to their cause (or any combination of those things) that it is unlikely that any deterrence strategy will be effective. Groups such as these might be sponsored by, or even a part of, aggressive governments, but hide underneath a dense shroud of deception and misdirection to prevent affiliation, or at least offer plausible deniability. Without knowing whom adversary decision-makers are, it is difficult to understand the ideology, objectives, perceptions, methods, and risk-taking propensity of an adversary. It also makes in virtually impossible to communicate directly with an adversary, in the style of the direct communications between leaders that helped avoid nuclear war between the U.S. and theSoviet Union.

2.5.2 Deterrent threats

What methods of deterrence might be available in cyberspace? It should first be emphasized that deterrence does not necessarily mean retaliation in the same domain (Beeker, 2009). Just as a border incursion by ground forces might be countered by an air strike, a cyber infiltration could potentially open up retaliatory options in other domains. However, "fighting fire with fire" raises fewer questions of the proportionality of the counter-attack (Libicki, 2009). Where can the line be drawn for cyber attacks warranting conventional retaliation? Reconnaissance, penetration, infiltration, damage to information, damage to cyber assets, damage to physical assets, damage to critical infrastructures, or loss of life? The potential array of cyber attacks is wide. However, stating that the U.S. will only retaliate to cyber attacks in-kind weakens it greatly by taking some of its strongest assets off the table (Libicki, 2009).

Some (e.g., Beeker, 2009; Chilton, 2008) propose a new deterrence triad: strike capability, defense fortifications, and responsive infrastructure (network operations). This new triad must emphasize the integration of these three components to maximize effect. A similar sentiment is echoed in Taipale 2010, which lays out a framework for cyber deterrence that includes penalty, futility, and dependency.

Penalty is the classic deterrent threat. However, one unique aspect of cyber retaliation is

the difficulty of predicting the ramifications. This has three potential problems. First, there is the potential for incidentally attacking innocent bystanders who stand between the defender and aggressor. Without a complete understanding of the path the attack will take (which is guesswork, at best due to the highly interconnected nature of the Internet), a retaliator risks unintended consequences. Second, if a counter-attack is made but results in no actual damage, it not only leaves the defender looking helpless, but also invites counter-retaliation (Libicki, 2009). Third, the U.S. could find its hands tied by making overly specific retaliatory threats or threats to retaliate against exactly those actions that would be used to retaliate (Beeker, 2009).

Another complication of cyber-retaliation is to demonstrate its credibility. In nuclear war, it was seemingly routine for the U.S. and the Soviet Union to conduct publicized tests of nuclear weapons. This left little doubt in the adversary's mind that the retaliation capability existed. However, in the cyber realm the success of an attack depends on many factors not likely known *a priori*. For example, exploits in software depend highly upon the version of that software used by a target. Even within a single military base there could be ten different versions of Microsoft Windows, all with different security patches applied! The challenge is to demonstrate the ability to mount a cyber counter-attack, without specific knowledge of the target, and/or without giving away too much information about the attack or knowledge of the target to allow the adversary to defend against the attack (Beeker, 2009; Taipale, 2010). It is a bit like having to tailor a bunker-busting bomb to detonate at a specific depth below ground level to destroy a control room without any knowledge of the architecture of the bunker!

For this reason, the most important pillars of cyber deterrence are probably futility through a strong defense. This doesn't require a large investment in offensive capabilities, international consensus, or questions of contestability. Defensive fortifications serve as a noise filter for attackers, allowing states to focus only on serious threats (Libicki, 2009), and prevents the need for retaliation. By demonstrating over time that a state's systems are secure and resilient, it stands to reason that an enemy's perception of the benefits of attacking will decrease (Habiger, 2010).

Dependency requires a state find ways to encourage potential adversaries to rely upon it to achieve their objectives (Taipale, 2010). A good example is the Global Positioning System (GPS). This system is operated by U.S. contractors and agencies, but its destruction would have negative impacts on adversaries. Primarily, any adversary who relies upon GPS for navigation or targeting would be impaired. However, it also would have the effect of making attacks by the U.S. less precise, thus increasing the chances of collateral damage and making the effects of a U.S. counterattack even worse (Beeker, 2009).

The U.S. is certainly in a position to leverage dependencies in many areas of cyberspace today. The majority of the root name servers, responsible for directing human-readable domain names to computer-readable addresses, are housed in the U.S.. Many of the world's largest sources of news, research, entertainment, and information are accessible to the world only through the networks of telecommunications operators based in the U.S.. These and other dependencies provide the U.S. a very large "carrot" with which to keep adversaries at bay.

To deny benefits, a defender must enhance its ability to detect and even preempt cyber attacks. This is a field where a great deal of research remains and the nature of cyberspace presents no small challenge. Second, it must be made obvious that the defender has the ability to fight through a cyber attack. It should be made obvious that the defender is not entirely dependent upon cyberspace, and is capable of operating without it. This is quite a feat in the days of software radios, GPS navigation, remotely-controlled drones, and high-grade encryption. By hardening potential targets, introducing redundancies and resiliencies where necessary, tightly controlling access to critical systems, and obfuscating the available information about cyber infrastructures, the defender can lower the temptation of attacking its cyber infrastructure.

2.5.3 Open questions

It should be obvious at this point that the matter of deterring cyber attacks is far from settled. A further examination of problems with cyber deterrence will be conducted later, but for now it is enlightening to consider some key questions posed in the literature.

First, is deterrence already working (Libicki, 2009)? This might seem like a silly question, given the large amount of press coverage of cyber intrusions and the amount of government effort going into developing deterrence and offensive capabilities. However, it can be said that to this point there is no known example of a cyber attack that has resulted in damage comparable to a conventional attack. Most recorded episodes are tantamount to espionage, not attack. This is despite, as mentioned above, the large number of actors who are more than capable of conducting such attacks. So it is fair to ask whether the U.S. need to up the ante. Perhaps the U.S. is already formidable enough, both on the cyber battlefield and off, that a large number of potential cyber attacks are already deterred. Assuming, however, this is not the case, Libicki 2009 raises the following nine questions for conducting cyber deterrence.

First, is there a reliable way to determine the identity of the attacker (also Moore, 2008)? Can the methods used to identify the attacker be shared in an international forum to convince third parties of the attribution? If attribution is not sound, the defender risks running afoul of international standards on acceptable behavior. There is a wide range of technical methods available to mask the true origin of an attack, requiring non-technical attribution methods to play a role. Tracking cyber attackers to their origin requires an unprecedented level of cooperation between the political, law enforcement, and private sectors of many countries. Even allies might be unwilling or unable to provide information to help.

Second, does the defender possess sufficient capability to credibly threaten an adversary? For all the reasons mentioned above, it is extremely difficult to develop a tested offensive capability and declare it to adversaries. International acceptance is also a potentially high hurdle, and only treaties between individual nations currently address this topic (Moore, 2008). Additionally, a counterattack must be palatable for the defender's stakeholders or the decision to retaliate could have negative political repercussions for decision-makers (Moore, 2008).

Third, is the ability to retaliate repeatable? Since cyber attacks rely on system vulnerabilities, any network administrator should, upon detecting an attack, find and close the hole that allowed it to succeed. There is also the risk that such vulnerabilities will be discovered on their own, before an exploit is attempted.

Fourth, if unable to deter an attacker, can one at least disarm them? Many systems used to attack might be extremely difficult to disarm or destroy, even by the aggressor (e.g., botnets). Other systems might not be accessible from the Internet (e.g., supercomputers used for code breaking) or be dispersed among multiple sets of hardware and geographic locations (e.g., content distribution systems such as Google, Amazon, or Akamai).

Fifth, is the fight contained to just the aggressor and defender? There's a possibility of involving allies or other parties who would like to see harm come to the defender or attacker dog pile onto an attack. In such a case, even if the primary aggressor is stopped by retaliation, other attacks might continue. Of course, this cuts both ways and could serve in itself as a deterrent. Understanding the cascading effects of a cyber attack would likely be a challenge even for the designers of many critical systems, not to mention an aggressor with limited access and intelligence (Moore, 2008).

Sixth, does retaliation send the right message to stakeholders? If the federal government is responsible for deterring attacks, does that mean that private companies are exempt from liability in the case of cyber war, a la force majeure? Does the fact that governments reserve the right to counter cyber attack with cyber attack give private citizens and organizations an expectation of freedom to do the same?

Seventh, where is the line for retaliation? If the defender states a zero-tolerance policy, it will consume itself tracking and punishing perpetrators. Cyber warfare is in many ways the twenty-first century espionage and is a continuous activity by friends and foes alike. A deterrent threat is only effective if sufficiently severe to make an aggressor think twice. Many cyber attacks are comparatively small in the scale of belligerence. Criteria such as loss of life or economic cost gel well with conventional warfare, but are unlikely to directly result from cyber attacks and are difficult to quantify, respectively. These "lines in the sand" should be well-defined, but shared only in vague terms for fear of allowing a precisely calibrated attack (Moore, 2008).

Eighth, in cyber deterrence, unlike nuclear deterrence, there is a concern of escalation. In nuclear war, a first strike was pretty much as bad as it could get. There really was no way to

top a nuclear attack. However, in cyber war there is the possibility that an adversary might respond to cyber retaliation with conventional weapons, economic or political maneuvers, or even nuclear war. If the labeled aggressors do not believe that retaliation was merited, face internal pressure to make a point, or think they will benefit from moving the conflict out of the cyber domain, there is a significant risk of escalation.

Finally, the defender must consider aggressors with little or nothing worthy of cyber retaliation. Given that a devastating attack could, in theory, be launched from a single laptop with a dial-up Internet connection, it follows that retaliation in kind might be no deterrent at all. The only response could come from conventional kinetic weapons, but at a risk of escalation or international condemnation.

2.6 Criticality of Attribution

Deterrence only works if the defender makes a credible retaliatory threat. If the attacker believes his identity will not be discovered by the defender, there can be no retaliation; thus, all deterrence fails. This is the principal reason that attribution is so important, and why many say that deterrence in cyberspace is impossible. To date, no sufficient methods of attribution have surfaced to make any deterrent threat credible. The next chapter will discuss existing attribution methods.

Of course in a perfect world, attribution might be easier. Take an example of a traditional crime, a bank robbery. There are many ways investigators might be able to establish the identify of the robber. Perhaps he left a handwritten note, had a unique voice, did not wear a mask, drove a getaway car with license plates, carried a unique weapon, or wore unusual clothing. It would not make much sense for the police to instead try to determine which roads the robber took to arrive at the bank, and follow them back to his house! Clearly, such an approach would be fraught with difficulties — unreliable eye witnesses, broken traffic cameras, open areas with no witnesses, multiple forms of transportation, uncooperative transportation workers, among others.

It is similarly difficult to try to identify a cyber attack by tracking down the path between

attacker and target. Poor record-keeping, misconfigured or lack of detection systems, intentional attempts to mask identity, uncooperative network administrators — the list goes on. Attackers are aware of this and deterrence fails because everyone knows that tracking down an individual in cyberspace in many cases is simply impossible — and difficult in most cases. For the reasons mentioned above that defense is difficult, attribution is equally difficult. Perhaps it is just being approached in the wrong way.

Regardless of how it is accomplished, attribution is critical. Retaliation is risky at best and counter-productive at worst. Without a clear, confident guess to the identity of the attacker, this risk is increased. Retaliating against the wrong party is not only embarrassing and ineffective, it might run afoul of international law. As previously discussed, any actions after an attack has happened are likely to be difficult and messy for the victim, even if the law is on their side. It is then most favorable to keep an attack from happening in the first place. Attribution is the key to doing so.

CHAPTER 3. ATTRIBUTION

A number of methods have been developed in an attempt to attribute the source of a cyber attack. These generally fit into one of several levels at which attribution can take place — the network level, the traffic level, on individual hosts, and non-technical methods. Each of these levels can be divided into techniques that can be implemented only if control of all devices in the path is held and those which are able to suggest an identity with a limited presence. The following is based upon an analysis of other surveys of attribution methods (Daniels, 2002; Hunker et al., 2008; Wheeler and Larsen, 2003).

Due to the highly interconnected and dynamic nature of the Internet, the majority of techniques can only give a high-confidence attribution in the former case. Given the highly anarchic nature of the Internet, any one organization is unlikely to exert control over all, or even the majority of it. Therefore, one must be able to obtain information for networks that belong to others to aid in attribution. As discussed previously, many states might be unwilling or unable to share such information for a variety of reasons. This implies that significant resources must be placed into establishing and maintaining presences in unfriendly or hostile networks if one is to successfully attribute cyber attacks, in general.

3.1 Network-based Attribution

3.1.1 Traffic logging

At the network level, it is necessary to utilize the devices responsible for relaying traffic between nodes on the Internet to perform attribution. These methods all require some level of physical access to and control of such devices. The most apparent method is to require network devices to store logs of all the traffic they handle, including source and destination. An attack

could then simply be traced link-by-link until the origin is found. Such a tactic would quickly fill even the largest information storage devices and make identifying individual traffic streams a computationally hard problem. Take, for example, a 10Mbps half-duplex (one-way) link, roughly equivalent to a decent home Internet connection and thus a gross underestimate for the high-traffic, 24x7 operating level 1 routers of the Internet. On such a link at full-capacity, 108GB of data would be generated daily, or a little over 39TB per year. This is equivalent to 27 digital video discs (DVDs) per day (or almost 10,000 per year). ISPs operate links which can reach 100,000 Mbps, so clearly storing a complete log of all traffic is not feasible.

3.1.2 Summary logging

Storing just the headers of packets is unlikely to solve the problem. A header contains the source and destination of the packet (ostensibly), as well as parameters which guide the handling of the packet by intermediate network devices. The information contained in the header allows traffic prioritization, loop prevention, congestion avoidance, re-sequencing of packets which arrive out of order, integrity verification, and connection state tracking. The minimum IPv4 header is 20 bytes (IPv6 headers are at least twice this size) and a typical transmission unit size is 1500 bytes. Given this ratio, one could expect at most a 98.67% savings by omitting the payload from archival. However, even in this case the storage requirements for an ISP monitoring a gigabit link might approach 50TB per year (100TB if full-duplex). At the time of writing, the largest single drives that can be purchased are in the 2-3TB range. It is clearly not cost-effective for an ISP to record all traffic.

3.1.3 Probabilistic logging

It has been suggested this problem might be solved by storing only parts of each transmission. Perhaps just the first and last packet, or every n-th packet. While this would certainly reduce storage requirements, it increases computational requirements. If every packet is recorded, at least the network device did not need to decide whether or not to record each packet. On the other hand, to maintain consistent sampling, the device must keep track of

every ongoing session, how many packets it has seen, and decide whether or not each packet should be recorded. This will also require additional temporary storage in memory to keep track of the packet count for all ongoing connections.

3.1.4 Compact logging

Similarly, one might suggest keeping only a hash of each packet in storage. A hash is a cryptographic function that maps a potentially infinite length input into a fixed length output. It is often used for integrity checking, because these algorithms (such as the Message Digest [MD] or Secure Hashing Algorithm [SHA]) are highly sensitive to the input. An ideal hashing algorithm will change every bit of its output given only a single bit change in the input.

There are limitations to these methods, but a more thorough treatment is beyond the scope of this analysis. Suffice it to say that one might regard a hashing algorithm as a reliable method to represent arbitrarily large data elements in a finite space. As such, one might be able to generate a hash of each packet crossing a network device. An MD5 hash is 128 bits (16 bytes) and a SHA-1 hash is 160 bits (20 bytes). If one were to record this hash and some information about the link on which the packet arrived and departed, he might be able to keep a fairly compact database of all traffic crossing a device.

This has several disadvantages. Like storing only packet headers, even with large savings in per-packet storage requirements, there is still a potentially large storage requirement unless the retention period is small. Like sampling, there is a computational requirement to generate the hash values as packets pass through the network device. This computation is significantly more difficult than the simple decision based on packet ordering, and will cause an even greater performance impact.

Finally, another feature of hashing algorithms is that they are one-way functions. That is, it is easy to use these algorithms to generate a hash value from an input, but to do the reverse is extremely difficult or impossible. Because of the use of modular arithmetic in most hashing algorithms, one cannot deterministically identify the input even with an infinite amount of resources devoted to the problem. It is as though someone says that she has added some

multiple of seven hours to a previous time and arrived at 1PM. The original time might have been 6PM, 11AM, 4AM, and so on. No amount of computation will solve this problem as there are multiple possible answers. The only way to "break" these functions is to iterate through every possible input and find values that produce the hash value in question and also look like valid inputs. This method of brute-forcing the input to a hash function is extremely computationally-intensive, and it is still non-deterministic. Therefore, one could generate the hash value of an attacker's packet and submit it to ISPs to look up in a database. The ISPs would be unable to find related traffic as they would be unable to inspect the content of the packet to find clues. Even assuming the victim gave them the full packet contents, they would only be able to make a single leap before coming up with more hashed packets.

3.1.5 Embedded logging

Another suggestion is to embed tracking information directly into packets or the network traffic streams. This method is employed in electronic mail messages. In the simple mail transport protocol (SMTP) header, there is something like a chain of custody for the message. Each device involved in routing the message to the recipient inserts a line in this log with its name, IP address, and the time the message is received. In this way, one can track the path of an email message to its original sender. There are several problems using this approach for broader attribution purposes. First, one must trust that no one has tampered with the header. It would be trivial for an intermediate party to alter the header to inject or remove hops from the path. It also assumes the first device in this recorded path is where the sender is actually "sitting". That is to say, it assumes the person behind the keyboard and the first device in the path are equivalent. As a hacker might compromise remote systems (even doing so on large scales, such as bot-nets) to send messages, this equivalence does not hold.

Finally, this method adds data to the transmission. In the case of email, where speeds are not generally much of a concern, this doesn't cause much of a problem. However, many network applications have very small tolerances for the delay that might be caused by piggybacking data. Voice over IP (VoIP) is one example. VoIP systems are very carefully engineered to

provide the amount of throughput needed to sustain the contracted number of telephone calls, while maintaining high quality. If one introduced the extra data and delay caused by adding path information to the packet at each hop in the path, the quality of voice calls (or at least the throughput efficiency) would likely decrease. This would be true whether the tracking data were injected into the packet or the traffic stream, although the latter case would not require the delay of rewriting the packet at each hop.

3.1.6 Centralized summary logging

Suppose routers then just generated periodic summaries of traffic they have seen and sent them to a central logging server for archival. This is the worst of each scenario. This requires storage on a central logging server in the way that storing the traffic (or samplings of it) does. It requires additional computational and storage resources on the network devices to keep track of connections and generate reports. It generates additional load for network links. It also introduces a whole new problem — where should these messages be sent? Would each ISP be responsible for maintaining a database? In this case, nothing has been gained from the current situation. One must still rely on ISPs to collect information properly, and to cooperate with investigations. Maybe routers send their information to a kind of escrow server where any party is able to perform investigations. It must then be decided who is trusted to operate such a system, how access can be controlled while still allowing the necessary access, how to mesh the laws of various jurisdictions to preserve the legality of the storage and retrieval of these data, and the not-so-small problem of finding a location capable of accepting incoming traffic reports from every network device in the world!

3.1.7 Network reconfiguration

A tactic that could potentially prove useful, given enough knowledge of the network topology, is that of reconfiguring, or otherwise affecting the network and watching for changes in the attack. As the events in China during 2010 showed (Hijacking, 2010), this is possible. Obviously, for electronic Blitzkriegs, there is not much this method can do, as there is insuffi-

cient time to gather the necessary data. However, for longer term attacks, such as the denial of service flooding attacks that took out networks in Estonia and Georgia, it might prove effective.

If a large portion of the path to the attacker is under the defender's control, specific network devices or links could be temporarily disabled to determine if they are used in the attack. With sufficient experiments of this nature and some heavy use of graph theory, a picture of the path to the attacker might emerge. If the network is not under the defender's control, techniques such as controlled flooding could be used to saturate upstream routers to *effectively* take them offline or significantly increase their latency. However, this could be viewed as a retaliation against a neutral party and invite retaliation of its own or international condemnation.

Such methods assume much about the level of control and observational capabilities of the defender. Perhaps, only a handful of organizations worldwide would be capable of exerting this level of influence over the structure of the Internet. And it is questionable whether they have the instrumentation in place to gather the necessary data to make use of this tactic.

3.1.8 Traceback

The most dramatic method is that of a "traceback". The idea here is the victim of an attack might be able to realize the attack is occurring, and unroll the path to the attacker link-by-link until the true network source is revealed. Of course, this "unrolling" might required that the victim do some hacking of his own. To gain all the information necessary to take another step toward the attacker, a high-level of access to each node is required. Log data, route data, open connections, addressing information, which exploit(s) the attacker used to get into the given node, and whether the node is in fact the true origin — to name a few. Given sufficient information to track the attack aback to its source, any number of attribution or retaliation options might become feasible.

This is perhaps the most dramatic scenario and one that is not new to Hollywood. The idea of a hacker-on-hacker dogfight, where they race to take out each other's connection before falling victim, certainly fits well with classical paradigms of how warfare works. But, this all

assumes events happen in timescales that are comprehensible, let alone actionable, by humans. While there are likely sequences of events the attacker must complete to fully penetrate a system, many of these are likely highly scripted or rehearsed, so they can be executed in quick succession.

The actual break-in most likely consists of exploit code that will run at machine-speed; that is, billions of instructions per second. It is difficult to imagine a human even handling a few tens of instructions per second, even if the instructions are clear and straightforward. How likely is it the victim-turned-attacker will be able to quickly reverse the attack chain with no *a priori* knowledge of the affected systems? At each stage he must follow the normal intrusion steps — footprinting, scanning, enumerating, and exploiting. As any penetration tester (or hacker) will attest, this process is rife with trial and error and is likely to take weeks or months on well-secured systems. The idea that the victim will be able to complete this process on multiple systems in quick succession before the attacker has vanished is a stretch, at best. The only hope for this method is the development of extensive automated attack tools, such as the Metasploit framework, which might be able to accomplish these tasks at high speeds. Even such tools will never be perfect.

Without a complete understanding of all devices traversed, it is also possible that unintended consequences (a la Morris) could wreak havoc on critical systems of allies or other bystanders, and create a much worse situation than originally existed. The Internet was designed to be highly survivable and maintains the ability to reconfigure itself to ensure the arrival of traffic. Such a system is called best-effort in that, within the parameters of its design, it will do everything possible to ensure it meets its design goals.

In the case of the Internet, dynamic route tables and route exchange protocols allow ISPs to work around unreliable or nonoperational links and devices, and balance the load across active links to minimize latency (delay) and jitter (variation in delay between packets), while maximizing throughput. A packet sent between two nodes on the Internet, even in the same town, might transit any number of ISPs and geographic locations. It might cross submarine cables, satellite links, underground lines, leased lines, private network exchanges, or any number

of other autonomous systems (AS). Each of these is potentially a node on the reverse path to the attacker. If the victim accidentally takes out a large ISPs router while tracing the connection, she might not only affect the attacker, but also anyone else using that device. This could mean critical infrastructure, hospitals, private citizens, allies, or any number of other bystanders. Because of the dynamic nature of the Internet, it is nearly impossible to say for certain (or control) the path one's traffic will take to its destination. Even U.S. DOD-controlled networks, like the GIG, likely make use of private relays for the sake of efficiency.

3.2 Traffic-based Attribution

The next class of attribution methods are those based on the modification or analysis of attack traffic. Certain protocols on the Internet are more spoof-able (easy to fake the origin) than others. Therefore, where possible, more secure protocols should be used. One example is that of the Transmission Control Protocol (TCP) versus the User Datagram Protocol (UDP), two of the most common protocols for transferring information over the Internet. The former requires a three-way handshake between sender and receiver, which at least guarantees the origin is who it says it is (to say nothing of stepping-stone attacks). The latter allows a user to send traffic to any host on the Internet with no verification of its origin or receipt. Protocols incorporating cryptographic guarantees of authenticity (such as IPSEC, the security enabled brother of the Internet Protocol) also make it extremely difficult to hide the origin, but have additional overhead to establish and process a secure channel.

3.2.1 Attributable sources

A luxury most network administrators do not have is to whitelist traffic coming into their networks. Instead, most administrators must rely on blacklists. The difference being, the former case assumes no one is allowed entry, but those explicitly defined. The latter case is the opposite. For most Internet sites, the goal is to encourage new users to visit the site. These visitors contribute to the bottom line of the web site, either through conversions to electronic sales, or indirectly through advertising impressions or "click-throughs" to sponsors.

In such cases, it it undesirable to require explicit permission granted to each and every individual wishing to use the site. This is not to say that e-commerce sites do not require logins; obviously, they do. It is to say that if one wanted to access Amazon.com, he must first contact Jeff Bezos and request that his computer's IP address be given access to the site. Only then could one visit the site and create an account.

Rather, one must open a site to everyone and take retroactive measures to curtail improper or unauthorized use (i.e., blacklist individuals). This is quite difficult, as the same individual might hold any number of virtual identities consisting of IP addresses, user names, email addresses, credit cards, or any other token selected. IP addresses are of special concern since an attacker might have hundreds or thousands of addresses at her disposal. One cannot simply block large swaths of address space for fear of locking out legitimate users. This gives the attacker the upper hand as she can effectively change addresses ad infinitum when detected. This will be even more the case in IPv6 where the size of the address space grows by 2^{96}! So, in most cases, one can assume a blacklist-based approach is essentially fully open to anyone who has the means to access it.

Conversely, networks such as military installations, critical infrastructure, and industrial control systems (e.g., SCADA) might have a very clearly defined set of users. These users might reside in a small portion of the IP address space (that is to say: out of 4 billion IPv4 addresses, they might all share a small subset). In this case, the whitelist approach might be suitable. One can ensure that only systems fully secured and accessed by authorized users are allowed to make use of sensitive resources. The U.S. DOD does this through the use of the Common Access Card (CAC) each employee carries. The identifying information on this card gives personnel access to sensitive information and systems, and uniquely ties their cyber actions to their person. Then,network administrators can configure systems to only allow access by those systems which support the CAC mechanism, and make use of access control lists to tightly restrict the flow of information into and out of the system.

One can envision agreements between network providers to make such restrictions work outside of these tightly-restricted environs, but it is a stretch. Any mention of a national

identity card or practically any other common access card for U.S. citizens has encountered stark opposition, due to privacy concerns. It is unlikely that any CAC equivalent for the general populace will exist in the near future. Even if it did, one would have to limit visitors to domestic addresses (if possible) until international agreement about the forms and exchange of identification information is reached. A virtual passport is simply not on the horizon.

There are some areas where this line of thinking might bear fruit, however. For example, e-mail servers can support the Sender Policy Framework (SPF), which verifies that the sender of a message is actually authorized to send mail for a given domain. If an unauthorized user wants to send a message from secure.com, but secure.com has set up SPF, the receiving server will see the message did not come from the authorized secure.com mail server and have the option to reject it. This is an example of a fully decentralized and easily configurable mechanism to reduce identify forgery in cyberspace. However, it must be coupled with a list of email service providers known to have attribution-friendly policies to be of the greatest use.

Network administrators can also reduce traffic forgery by configuring network devices and servers to intelligently watch incoming traffic. If traffic arrives on an interface that has no path to the network address the traffic claims, it is likely that the traffic was spoofed (sender address forged) as part of an attack, or at least a reconnaissance attempt. Due to the route exchange protocols on the Internet, and a number of special subnets defined in IPv4 and IPv6 (used mainly for private networks, non-routed networks, and research networks), this is fairly trivial to achieve. However, like SPF, this monitoring only helps to ensure that a sender does not forge his identity. It does not guarantee the sender is coming from an attributable network.

3.2.2 Watermarking

A more subtle way of tracking traffic has been suggested — that of traffic watermarking. Unlike watermarks on financial documents or letters, these watermarks are inserted into the network traffic stream. This discussion will place aside the use of steganographic watermarks (i.e., embedding information within the payload in cryptographically secure and hard to detect ways), since their use is probably not feasible in the high-speed environment of real-time Inter-

net traffic. The idea is to subtly alter the timing between subsequent packets (the inter-arrival time, IAT) in a detectable way. However, "detectable" is based on perspective. The alteration must be significant enough to rise above normal variations in timing, due to congestion, but minor enough to not draw the attacker's attention to the fact that he is being monitored. This is easier in bulk-transfer applications, such as web browsing or file transfers, but much more difficult in time-sensitive applications, such as interactive login sessions, VoIP, or multimedia streaming. The more time-sensitive the communication, the more likely an attacker will notice something is out of the ordinary. This is to say nothing of systems which might specifically be looking for such watermarks.

So the watermark must be very subtle. But, at some point it will no longer survive the jitter added by normal network data transmission, requiring detection to rely on statistical measures to detect partial watermarks. This surely will become messy very quickly.

Additionally, all of this assumes that the observer is able to position herself in such a way to both modify the traffic sent at the destination, and monitor the traffic along intermediate and destination devices. If one already has this amount of control over the network, the additional value of watermarking is questionable. There are still applications for encrypted traffic, where one cannot readily observe the payload, but even encrypted packets have a header in plain text, so they can be routed.

If one wishes to hide this information, she must use a system such as Tor, which will accept fully encrypted packets destined for its otherwise benign network address and relay them through a secure network to their destination. In such cases, the intermediate nodes will "re-packetize" the traffic and destroy the watermark anyway! Along the same vein, if the attacker uses any intermediate stepping stones in the attack, there is a possibility this will also corrupt the watermark during retransmission.

A technique exists that has been found to break the anonymity of Tor-like systems (Mathewson and Dingledine, 2005; Murdoch and Zielinski, 2007). If the defender can observe both the ingress and egress points for a given stream, it is possible to associate patterns of traffic by headers, throughput, or timing to make guesses as to the source and destination of a stream

of traffic. It is very difficult for a user of these systems to circumvent this technique, as one typically exerts little control over the encrypted "circuit" his traffic uses - even if this is an option. But, it is also difficult for an attacker to control enough nodes to have a good chance of observing both the entrance and exit of traffic through the network. Perhaps, this might all be beside the point, as the latency introduced by these systems is so high it makes them unusable for most interactive connections.

Additionally, watermarking can occur by modifying the packet. Known modifications can be made to traffic along the path between the attacker and the victim. A defender then uses network devices in various locations along potential paths to watch for these modifications. For example, a unique string could be inserted into the return packets of any connection triggering an alert on an intrusion detection device. If this string is detected elsewhere on the network, it can be said with some certainty that a malicious user might be at or near that location. However, this requires coordination in advance of resources and a great deal of luck with placement of sensors. The question again arises - if the traffic near the attacker can be monitored for the watermark, why can the defender not just monitor for the attack?

3.2.3 Self-identification

It is also possible an attacker is careless about the information she conveys to the victim. Basically, this is a degenerate case of attribution. Perhaps the attacker fails to mask the source IP address of the attack or sends a phishing email from his personal email account. In such cases attribution is very easy, but there are other subtle forms of self-identification, or at least providing information that can aid attribution.

Any phishing scam's intent is to get the user to take some action to reveal otherwise privileged information. This is necessarily a two-way communication with the attacker. The user might be tricked into visiting a fake bank website, running a script that calls back to a server under the attacker's control, asked to email or telephone her password to the attacker. In every case, there is a trail to the attacker, and the question becomes how well the attacker hid that trail. Unfortunately, in many cases the answer is very well.

It is trivial to set up email accounts and phone numbers using free services, such as Google Mail and Google Voice, that can be simply impossible to tie to an individual or organization. Hackers frequently make use of compromised machines or machines in jurisdictions which will mask their true identities as bases for their attacks. They might string together a number of compromised systems to hedge their bets, or even trick bystanders into performing the attack for them!

In less obvious ways, information contained in the network connection exchange might shed light on the attacker's identity or affiliation. For example, whenever visiting a website, the user leaves behind a kind of fingerprint containing information about his system known as the "user-agent" string. For example:

```
Mozilla/5.0 (Windows; U; Windows NT 5.1; en-U.S.; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1
Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en) AppleWebKit/125.2 (KHTML, like Gecko) Safari/125.8
Links (2.1pre15; FreeBSD 5.3-RELEASE i386; 196x84)
Googlebot/2.1 (+http://www.googlebot.com/bot.html)
```

In each case, some information is revealed about the end user. With the exception of the last entry, these examples would unlikely provide sufficient information about the user to lead to an attribution. However, suppose one came upon a user-agent string that contained "Red Flag Linux" (the Chinese Linux distribution), or non-English words, or even custom strings resulting from customized browsers or operating systems. In these cases, this information might help.

This example is somewhat elementary, but the concept, in general, still applies. Network administrators make use of operating system fingerprinting with scanning tools, such as Nmap, to determine software running on remote systems. This is based on the fact that certain operating systems respond to network traffic in particular ways. This might be an unusual sequence of flag settings, timings, packet sizes, or other protocol-level features. Older operating systems might change sequence numbers in network packets in semi-predictable ways.

So in subtle or not-so-subtle ways, attackers might reveal more about themselves than they realize. These techniques are highly operating system, application, and perhaps attacker-

dependent, however. One would need to be aware of peculiarities that might be exhibited in all of these different situations and with which individuals or groups they are affiliated. Because of the large degree of homogeneity of hardware, software, and networking protocols, it is unlikely that information gleaned in this way would be sufficient to pin down an individual. Still, it is another tool in the box.

3.3 Host-based Attribution

A natural progression, based on the many difficulties of network-based attribution, is host-based attribution. Unlike network-based techniques, host-based techniques operate in a much less volatile and unpredictable playing field. In many cases the same types of information might be obtained, but the method is different.

3.3.1 Caveats

The main advantage of host-based attribution is it is easy to access target devices. It might also be easier to covertly load special purpose software onto, say, a console in an Iranian Internet cafe than onto a network device belonging to an ISP. The latter case is likely to be fraught with much peril — legal, political, and technical. As mentioned previously, a poorly planned modification to a network device could have far-reaching effects on innocent bystanders. It is also much more difficult to obtain the legal authorization and cooperation of an ISP. Conversely, relatively little harm is likely to come from making changes to a single system (unless that system serves a critical purpose). Defense software can be applied only to those devices at risk. Offensive implants can be inserted only into devices that present a threat. This has the potential to significantly increase the signal-to-noise ratio of information gathered for analysis.

On the other hand, to maintain such an extensive presence is to make several assumptions. Primarily, one must have some idea of where an attack might originate or terminate to ensure the correct systems are instrumented ahead of time or inspected after-the-fact. Doing so represents a potentially significant investment of time, money, service downtime, and perhaps

even political capital and legal risk-taking. Because of these costs, one would wish to specifically target as few hosts as possible with implants or forensics. One could argue that one must have a fairly well-defined idea of where the attack came from to target host-based attribution methods. This is to say, as with watermarking, these methods provide little marginal value.

3.3.2 Local host analysis

Setting aside this caveat, the discussion returns to methods which can be used to obtain information from hosts to aid in attribution. The easiest are forensic methods applied to one's own systems. This could be log file analysis, file system forensics, intrusion detection alerts, or analyzing exploit code that has been left behind. All of these methods bypass the legal and political constraints that might arise outside of one's own network. However, there is a limit to the value obtained from them.

Log file analysis and intrusion network alerts, like many of the network-based forensics, only reveal a link in the chain (unless the attacker was highly careless). An in this case, it is only the very first link in the chain. Still, information from these might be used as part of a broader profile of the attack, a topic which will be discussed in the next chapter.

Similarly, analysis of any remnants the attacker left behind might help to build a profile of the attacker. But again, these are not likely to be of much use on their own. Unless the attacker used very unique tools or methods, which might identify him or his affiliation, one has only information to use for comparison with other attacks.

One might choose to construct honeypots — systems whose only purpose is to invite attack. These systems can be specially instrumented to gather additional information about the attacker that a normal system might not collect. Perhaps additional details about the network connection characteristics could be recorded, records of all system calls could be sent to a secure archival system, the file system could use a "tripwire" system to watch for any changes made by an attacker. All of these things can help shed additional light on the activity of the attacker, but are still mostly limited to building a profile.

3.3.3 Remote host analysis

Often, systems are already instrumented to allow their administrators to monitor their health and status. If a defender has some idea of where an attack is originating, but needs to gather additional information, she might be able to query a host directly for more information. For example, many systems make use of the Simple Network Management Protocol (SNMP) to make information about running processes, available resources, logged in users, or other system stats available to a central monitoring system. If these are not properly secured (or the security can be bypassed), this might help to jump to the next link in the attack chain, or verify the origination point of an attack.

Comparable to the traceback mentioned above is a "hackback", where the defender directly exploits a system being used to attack. Of course the same limitations exist as with network-tracebacks. Assuming, however, that this method is feasible, one might be able to implant software onto an attacker's system (or an intermediate node in a stepping stone attack) that will deliver additional information about the attack and attacker.

Unfortunately, maintaining a presence on remote hosts is probably the most labor intensive method of attribution. One needs either a highly-specific target or a very large net. A very large net comes with the reality that systems are routinely patched, moved, reinstalled, upgraded, discarded, bought, sold, and so on. One cannot simply deploy a large number of implants and be done with it. It is an ongoing task to monitor the health and viability of implants already "in-the-field" and restore functionality when necessary. If software vendors are doing their jobs, this will become more difficult with each new release of an application. And, of course, all of this must be done in such a way as to evade detection.

Viruses and root-kits have become quite good at this task, but this is likely only because of their abundance (call it natural selection). Malware protection software is constantly evolving to detect such malicious modifications to systems. The irony of the situation is that all of the effort put into making one's own systems more secure might very well be used by the adversary to make it more difficult to penetrate their systems. For example, one might be diligent about submitting new threats to an anti-virus vendor to ensure that his protection is

the most robust it can be. There is a good chance, however, that the adversary might be using the same anti-virus software! By helping the anti-virus vendor, one potentially also aids the enemy.

3.4 Non-technical Attribution

All of the above methodsrequire some level of access to the devices used for an attack, which can prove difficult when confronting a well-informed adversary or when making requests to uncooperative third parties. It is worth mentioning that there is a long precedent of non-technical intelligence gathering going back millenia. By working in concert with law enforcement and intelligence agencies, such traditional trade craft might be employed to aid attribution.

If there is evidence an individual or group is planning or executing an attack, it might pay to observe them or gather further information about their activities. This could mean any number of traditional espionage techniques, as well as twenty-first century techniques such as intercepting e-mail [mail], installing keystroke recorders [wiretapping], or monitoring via electromagnetic radiation [eavesdropping]. It has been confirmed that even recording the sound of a person typing on a keyboard can reveal what is entered (Zhuang et al., 2009) and the contents of a cathode ray tube monitor can be read at a distance with specialized receiving equipment (Kuhn, 2003).

It is outside the scope of this work to enumerate the many ways that attribution in the cyber domain might be aided by tactics outside it. Many other options exist to determine the source of an attack, which do not rely on any of the cyber-techniques mentioned previously. However, such tactics are likely very difficult to employ even for governmental agencies, and most of them are simply out of the realm of possibility for other defenders. It is also potentially more difficult to integrate information collected in these ways with the fast-paced and highly-dynamic stream of information coming from cyber attribution methods.

3.5 Need for New Attribution Strategies

From the previous discussion it should be clear that none of the existing attribution methods are well-suited for all cases and many of them are unlikely to be useful in more than a few cases. There are simply too many ways for an attacker to befuddle attempts at learning his identity. The assumptions of current attribution methods fall into the categories of omniscience, omnipresence, or *a priori* positioning. Each of these is an unreasonable assumption and, thus, a new method needs developed, which does not make them. A summary is shown in Table 3.1.

Many methods assume that one can obtain information from all, or at least a significant portion of, the nodes between attacker and victim. This is the assumption of omniscience, that a defender (or attributor) can obtain much of the information necessary to analyze the path of the attack and determine its origin. Network-based attribution assumes that one can obtain information about the traffic flowing through nodes on the network. Traffic-based methods assume one can observe traffic at various points along the network. Host-based assumes one knows where to target surgical attribution efforts and that the information pulled out will be sufficient to aid attribution. In the highly decentralized world of cyber space, one cannot assume knowledge of any traffic beyond her own network borders. This is to say that any general-purpose attribution technique must be able to function with knowledge only of traffic, devices, and users of the victimized network, and nothing beyond.

Second, many of the techniques require software agents in place at strategic points along a network path. Whether this is in network devices through which traffic will pass, entry and exit nodes of anonymization networks, systems in proximity to the origin of the attack, or elsewhere. In the worst case, this requires constant "care and feeding" of a large number of software agents throughout both friendly and hostile networks (and everything in between). It is pure luck, if agents are in the right place at the right time. Perhaps luck is too harsh. But, in the exception of highly targeted agent installations, one must rely on large numbers of agents to increase the chances of gathering sufficient information for attribution. In the best case, where it is both technically and legally feasible to place an agent at such locations, the speed of doing so means that by the time the agent is in place, the attack might have already

Table 3.1 Attribution Assumptions, Old and New

Old Assumption	New Assumption
Omniscience	Knowledge only of events within one's own network
Omnipresence	Deployment of agents only within one's own network
A Priori Positioning	Must rely on budget-constrained agents deployed in likely attack locations

subsided. More discussion on this will follow. In general, one cannot assume the placement of software agents in any network that is not directly controlled by the defender.

Finally, the best attribution methods and positioning in the world are of no use, if they are not deployed in time to observe the attack. Perhaps forensic examination of network device logs might be an exception, but in most cases the level of detail held in these logs (if they are even enabled) is not sufficient for attribution purposes. The same can be said for host logs. Therefore, if one requires a software agent or network device observe the attack taking place, he must also assume this tool must be in place at the latest when the attack begins. This assumption of *a priori* positioning means, if the assumption of omnipresence does not hold, it must be known ahead of time where to place a limited number of agents.

What is needed then is an attribution method which relies solely on information that can be observed directly by the victim. This still makes a critical assumption— the victim is prepared ahead of time with whatever logging or forensic capabilities required for attribution. This is a much more reasonable assumption than the three previous methods. The limits to this assumption are merely the education, financing, and discipline of the defender. There are no legal, political, or technical barriers to implementing these methods on one's own network.

However, to say that one must be able to identify the perpetrator of an attack, based only on what can be seen at the target site, seems like quite a stretch! While there is obviously no way to obtain a deterministic individual identity, based on such information, it might be possible to at least say that two attacks were perpetrated by the same attacker. As it turns out, such research is already quite commonplace in literary analysis and intellectual property disputes.

CHAPTER 4. TOWARDS A *CYBERPRINT*

To avoid the assumptions in Table 3.1 is to throw out nearly all attribution methods suggested to date. However, there is a distinction in the attribution method that will be proposed here. While many existing methods of attribution are concerned with tracking down the person behind the keyboard (or at least the organization driving that person), *Cyberprints* take a broader view. This new method begins with the assumption that there is some existing knowledge, defined outside the scope of this work, about the origin of past attacks. How this information is obtained could be through any of the attribution methods above (though as demonstrated, this is unlikely to yield much success) or through political, espionage, or open source intelligence channels.

Therefore, it is somewhat unfair to directly compare *Cyberprinting* to all of the methods above, but it might have advantages which they lack. It is often the case that the true origin of a past attack or series of attacks is known with some certainty through non-cyber means. Therfore, it might be desirable to compare a recent or ongoing attack to what is known about past attacks and try to determine if it matches the characteristics of an attack of "known" origin. The question then is how one might perform this comparison.

Cyber attacks vary widely among a large number of dimensions. However, there remains the possibility that certain attackers or groups of them portray some detectable profile. While this profile might be hard to define, it should be detectable through statistical methods if it exists. The digital and highly-quantitative nature of network communications lends itself well to such analyses. Then the question is what method of analysis might detect such a profile and what features of an attack are reliable enough to form a profile?

4.1 Stylometry

A similar problem is had in literature. Often documents of unknown or disputed origin are in need of analysis to determine their author. Like cyber attacks, there simply might not be any way to follow a chain of events to the original author. Also like cyber attacks, it might be the case that a work is suspected of being written by an author who has other works of authenticated origin. A sub-field in literature studies has developed around these parameters, that of Stylometry.

Stylometry is a technique, based in literary stylistics, used to ascertain or verify the authorship of written documents. It is based on the assumption that an author has a particular set of writing habits that he is unable to alter without conscious effort (Corney et al., 2001). This set of features has become known as a *writeprint* (Abbasi and Chen, 2008; Li et al., 2006), analogous to a fingerprint in biometrics. In the same way a biometric feature is inherent, difficult to alter, and expresses itself without the intent of its carrier; a writeprint is something of a psychological (or behavioral) biometric. This technique has been used in a large number of studies for such tasks as determining if Sir Francis Bacon actually wrote the plays of Shakespeare, determining the author of the Federalist Papers, or determining shared authorship of portions of religious texts.

One of the largest challenges in performing this type of analysis is to select an appropriate set of features to analyze. The trick is to select enough appropriate, diverse features to capture the writeprint of an author, without overburdening the analysis. Choosing too many features will increase the computational requirements for the analysis (the "curse of dimensionality") (Kira and Rendell, 1992; Li et al., 2006) and also introduce noise (Kira and Rendell, 1992). This noise will make it more difficult for any method of analysis to tease out the writeprint from the natural variance in the writing style, or even counteract the identifying features and detract from the analysis (Caruana and Freitag, 1994).

Which features, or style markers, can then be used for analysis of texts? Li et al. (2006) suggest these features can be separated into four categories: 1) lexical features, 2) syntactic features, 3) structural features, and 4) content-specific features.

Lexical features are those based upon the usage of language in the text. This represents measures such as the vocabulary distribution (Yule's K) and variety (Simpson's D) (Baayen et al., 1996; Corney et al., 2001; Hayes, 2008; Holmes, 1994), word length distribution (Holmes, 1994), or words only used once (*hapax legomena*) and words only used twice (*hapax dislegomena*) (Corney et al., 2001; Holmes, 1994). Such measures are highly dependent upon the context of the writing.

Syntactic features are attractive because they represent the way an author constructs her writing, regardless of the context. The function word usage (common adverbs, auxiliary verbs, conjunctions, determiners, numbers, prepositions, and pronouns) (Corney et al., 2001; Hayes, 2008; Holmes, 1994; Li et al., 2006), letter frequencies, n-gram (series of n letters or words) frequencies (Corney et al., 2001), and the usage of punctuation, such as commas, exclamation points, question marks, or semicolons (Li et al., 2006) could be used.

Structural features, similar to syntactic features, do not have much dependence upon context. These represent the way an author constructs the language of a document, such as distribution of syllables per word, word length distribution, word collocations (words frequently used together), sentence length, preferred word positions, prepositional phrase structure, distribution of parts of speech (Corney et al., 2001; Holmes, 1994), sentence length, parts of speech, syllables per word, or frequencies of i-syllabled words (Holmes, 1994).

Content-specific features are those which vary greatly, depending upon context. For example, one would expect lots of references to money (or perhaps "greenbacks", "dough", "moola", or some other name) in communications discussing a bank robbery, but not in a communication discussing a chemistry assignment. Words which have a high frequency (Baayen et al., 1996) in a particular communication might reveal its purpose, if not author.

In all of these categories, mistakes made by the author can be revealing. Grammatical mistakes, such as sentence fragments, run-on sentences, spelling mistakes (single consonant instead of double, double consonant instead of single, confused letters, wrong vowel, repeated letter, only one of doubled letter, letter inversion, inserted letter), abbreviations, all capital letters, and repeated non alphanumeric characters, all can aid in analysis (Koppel and Schler,

2003). However, any editing of the communication (proofreading, spell-checking, etc.) can mask these features. Therefore, Rudman's Law says the closest text to the original author's writing should be used for analysis (Juola and Baayen, 2005).

There is an entire set of features outside of the verbiage itself. At least as far back as 1910, there were studies into the physical properties of documents and what they might reveal about the author (Osborn, 1910). Handwriting analysis is perhaps the most commonly used; in fact, the veracity of even contemporary legal documents depends upon the ability to detect counterfeit signatures. However, other features, such as the type of paper or ink, font choices (if typed), margins, or the method of publication, might reflect something about the author. All of these things are less likely to be consistent between a variety of documents from the same author. While a writer's writing habits are subconcious and robust to change, these higher-level features are likely to be context-dependent.

4.2 Software Forensic Analysis

A natural extension of Stylometry is the analysis of software. After all, software is in many ways just another form of literature. Like prose, it must follow a set of rules regarding its construction. Yet, it also allows for a degree of freedom in the particular way in which each programmer implements the same logic. Like poets, many programmers take great pride in expressing the logic required to accomplish a task in the most elegant way possible. Therefore, even simple tasks can be expressed in a number of different ways. For these reasons, many of the same analysis methods might be applied.

The theft of intellectual property, whether it be a term paper or trade secret, has become a major issue as the ease of transferring data quickly and with perfect fidelity has increased. This has led to a great deal of effort in translating work in Stylometry to the domain of software analysis. Additionally, the continued spread of malware, such as viruses and trojans, provides a large incentive to security product vendors to identify the origin of a piece of code and make a determination about its trustworthiness.

Some literary features have a direct application or analog in software analysis: choice of

language, word frequency, line length, typos, or punctuation, for example. Many obviously do not apply, since software source code and machine code are not intended to be read at length by humans. However, software adds a host of its own identifying features.

A degenerate case is the analysis of comments in source code or debugging symbols in machine code (human-readable content left in the final executable to aid in troubleshooting misbehaving software). These might be analyzed with the exact same methods used in literary analysis. However, on their own they are unlikely to provide sufficient information for an attribution, due to their relative scarcity and brevity.

One might also make use of the choice of data structure types in a piece of software. In a complex language, such as C++, a set of related data might be represented by an array, linked list, struct, or vector. While there are operational factors to consider when determining which of these to use, for many cases they can be used interchangeably by the programmer (as long as he is consistent!). The choice of one over the other might simply be a matter of preference or the type of training the programmer received. In such cases, it is likely this choice will be consistent across many pieces of software written by the same individual.

In the same way, the choice of algorithms could be revealing. To sort a list of elements, there are a multitude of algorithms available. Some are generally accepted as being more efficient for a given task or list size, but there are still many ways to achieve this task. Again, the choice is indicative of the background of the programmer. This same profile might emerge in the choice of system calls, operators, and functions used by the programmer (Hayes, 2008).

A quick way to summarize the programming style of an individual is often of interest. Primarily, the more complex the software, the more likely it will be prone to bugs. For this reason, there are a number of complexity metrics (Oman and Cook, 1989) that analyze code to draw attention to areas that should be reviewed in more detail. For example, the Cyclomatic Complexity (developed by Thomas J. McCabe, Sr. in 1976) measures how much of the code is written in ways that cause it to repeat the same instructions, versus maintaining a purely linear execution path. One might also measure the number of simultaneously executing instructions (i.e., the degree to which the program is parallelized), as interactions between

multiple software agents are more prone to race conditions, resource contention, or other hard-to-diagnose defects. All of these are designed to draw attention to potential problem spots in the code, but also represent a potential feature.

One large difference between literature and software is that professional software engineers are likely to make use of coding standards and templates. Coding standards and templates are usually policies of an organization, with which all of their programmers must comply. They dictate the style and mechanisms that should be used in the code, to maintain consistency within a team of programmers. This can serve to mask many of the profile features mentioned above. Additionally, the rules of syntax in source code tend to be much more strict than those of literature and, thus, simple programs are less likely to display unique characteristics than literary works. The "Consistent Programmer Hypothesis" (Hayes, 2008) posits this cuts both ways. Programmers must write code that functions within the rules of the language, which will eliminate some of the profile diversity. This also means that their own style is likely to be highly consistent.

4.2.1 Electronic communications

Somewhere in between traditional communications and programs are electronic communications. These often can be analyzed in much the same way as traditional communications, but have some important differences. One potential problem when applying stylometric techniques to these is that of size. Typically, literary analysis is performed on thousands or tens-of-thousands of words. Like source code, electronic communications, such as forum postings, chat room logs, e-mails, or instant message transcripts, might only consist of tens or hundreds of words (Corney et al., 2001). This small sample size makes statistical classification much more difficult.

On the other hand, in addition to literary features, electronic communications might also have some of the features of source code. They might contain things like server or network path information in a header, IP addresses, or formatting syntax. They might also contain unique literary features, such as greeting and farewell texts, signature blocks (Corney et al.,

2001), attachments, screen names, or emoticons. These might be used consistently (or even automatically) and provide additional stability to statistical analysis.

Electronic communications are also unlikely to be edited by the author or anyone else. This provides a very direct representation of the author's style (Koppel and Schler, 2003). However, all these additional features and potential added stability might simply be overcome by the lack of richness in the data set. This issue of stability versus sensitivity is one that will constitute a major metric of any analysis method and *Cyberprints* are certainly no different.

4.3 Stability and Sensitivity

A *Cyberprint* should comprise a metric that reflects the methods, systems, tools, or infrastructure of an attacker. Attackers are unlikely to develop entirely new tools, build new systems, or set up new bases of operations for each and every attack. Therefore, while their profile might evolve over time, it stands to reason that commonalities in their technique hold the potential for an unintentional and unalterable profile. However, questions remain about which features fit this description. Before this topic is addressed, a more fundamental issue must be addressed. As potential features are analyzed, one of the key questions will be whether the feature is consistent for a given attacker across attacks, but unique enough that it is likely to vary between attackers. A *Cyberprint* must be stable in that it should have a fair chance of recognizing the same attacker in different scenarios, but sensitive enough to discern between different attackers.

4.3.1 Consistency and variance of features

This is the primary challenge of this undertaking. Investigations must be made into various potential features to determine where they fall in this balance. It is difficult to known *a priori* where many features will fall in the spectrum of consistency. This is largely due to a large variety of potential sources of variation. For the purposes of this work, the scope of features will be defined as anything which would be collected by a network traffic capturing device at the site of the target.

A full enumeration of potential features in this scope will be provided in the following section. It is, however, worth listing some sources of variance at this level before going forward. Since these features must all be observable by a network traffic capturing device, it follows they will all be based upon data contained within these network packets. The scope is further constrained to the most common protocols on the Internet, TCP/IP and UDP/IP. A typical deconstruction of network traffic follows along the lines in Table 4.1. In this, the OSI model, data are classified by the level of encapsulation they have undergone.

Encapsulation is the successive process of "wrapping" an item of data into another item of data so they can be handled properly. For example, a packet from a web server to a web browser contains data representing the body of the web page. If it is encrypted, this must first be completed, and the results placed inside a special packet so the receiver knows to decrypt the packet before trying to read it. At this step, the contents of the web page are not important, and so these data can be ignored by the software performing the encryption. It is said the upper level data has been encapsulated by the lower-level agent.

As the transmission traverses down the levels of the OSI model, it will be encapsulated a number of times, each time information required at that level is added to the packet, and the newly encapsulated information becomes irrelevant (save for more strict screening, such as deep-inspection firewalls or IDSs). At the lowest layer, the data have been completely encapsulated and become merely a series of physical representations for zeroes and ones. At the receiving side of the communication, this entire process is reversed. The data are successively unwrapped and the encapsulation removed until it reaches the web browser, and the contents of the web page are available for display.

At the lowest layer, the physical layer, variation is in the form of unique attributes of the transmitting devices. Such features are detectable by idiosyncrasies in the physical construction of devices and link mediums. Some work at Iowa State University (Daniels et al., 2005; Gerdes et al., 2006; Jackson, 2006) has shown that fingerprinting of devices at this layer is possible, implying a significant variance at this level.

At the data link level, the unit of analysis is the frame. A frame consists of a single packet

Table 4.1 Open Systems Interconnection (OSI) Model

Data Unit	**Layer**	**Function**
Data	Application	Communication between client and server, or any two other application endpoints.
	Presentation	Any function required to transform data into application-readable form (e.g., decryption, endian-conversion).
	Session	Inter-host communication.
Segments	Transport	End-to-end connections.
Packet/Datagram	Network	Path determination and logical addressing.
Frame	Data Link	Physical addressing.
Bit	Physical	Transmission of data over medium (e.g., optical, electrical, radio).

of transmitted data on the wire (and thus, the terms "frame" and "packet" will be used interchangeably). While bits, represented by voltage differentials, light pulses, or electromagnetic modulations might be thought of as the atomic unit of transmission, packets are the molecules resulting from the combinations of these bits. Like molecules, it normally takes many packets to make anything useful, so most transmissions consist of a series of packets. The most common data link-layer protocol is Ethernet (IEEE 802.3). Other, largely extinct protocols include Token Ring (IEEE 802.5), LocalTalk, or serial connections.

In addition to Ethernet, some common data link protocols currently in use include IEEE 802.11 (henceforth, referred to simply as "wireless"), Asynchronous Transfer Mode (ATM), Multiprotocol Label Switching (MPLS), Point-to-Point Protocol (RFC 1661 - PPP), or the Fiber Distributed Data Interface (ANSI X3T9.5 - FDDI). Of these, the only protocol of particular interest for *Cyberprints*, other than Ethernet, is wireless. The others are used primarily for wide-area network (WAN) connections between ISP nodes, or for localized storage network connections. In both Ethernet and 802.11 there are a number of fields that must appear to ensure correct delivery of the frame. Any of these might constitute features.

At the next level, network protocols, such as IPv4, IPv6, the Internet Control Message Protocol (ICMP), or the Border Gateway Protocol (BGP) are used. These are responsible for creating logical circuits between two nodes on a network or inter-network (hence, Internet). In the former case, at least in simple networks, this might be little different path-wise than transmissions at the data link layer. However, the addresses used at the transport level are all routable. That is, all network devices must be able to recognize these addresses and make a determination as to where (if anywhere) they should be forwarded to progress closer to their destination. Certain addresses are recognized as "non-routable" by the standards, used for things such as local networks that need no allocation approval or automatically configured networks. However, the non-routing of such packets is a matter of policy (often implemented in the networking software), not protocol.

As with data link packets, the data required to properly handle a network level packet contain a number of features which might be useful in attribution. Some of these might only be useful in generating statistics about a stream, some might be useful in and of themselves. Variety at this layer comes from the different ways in which operating systems developers choose to implement network protocol standards. While that might sound counter-intuitive, there is a large amount of gray area in the standards which can vary from implementation-to-implementation without affecting the success of communications. This will remain true, at least from this layer up, and might also be the case to a lesser extent at lower layers.

The session layer, attribution-wise, will be very similar in variability to the network layer. Session protocols, such as Transmission Control Protocol (TCP), User Datagram Protocol (UDP), or AppleTalk, are used. TCP, by and large, dominates traffic on the Internet. It is a protocol designed to establish a reliable session between two hosts, and is capable of tolerating network congestion, delays, and outages. UDP, conversely, is a very simple session protocol that is simply "best effort". UDP will send a packet to the destination, and assume it got there. If something happens along the way, there will be no indication that a retransmission is required. AppleTalk is largely extinct, but it served a similar purpose.

TCP, a complex protocol, is potentially fruitful for attribution. TCP must manage things

such as re-sequencing packets received out of order, controlling the rate of flow of traffic to comply with the realities of the transmission channel, re-sending packets not acknowledged as received, and termination of idle sessions. In each of these areas is a degree of variability that might vary from attacker-to-attacker, especially when unique or custom systems are used.

Finally, the presentation and application layers are both likely to vary a great deal, depending upon which applications are used. The choice of attack method or target profile might dictate the use of certain software applications used by the attacker. Such applications might have detectable signatures created by the way in which they are constructed and executed, or information which they include in transmissions. This is perhaps the richest area of variation, but also likely the most difficult to pin down.

4.3.2 Number of features

It would be easy to believe that simply throwing all possible features into the pot, stirring, and pulling out an attribution might be the easiest and most fruitful way to accomplish the task at hand. However, it is likely that doing so would, in fact, undermine the ability of any algorithm to make successful identifications. Of primary concern is the fact that with increasing numbers of features comes exponentially increasing computational effort. It is not unacceptable to require significant computational resources, but any additional effort required should be justified by its contribution to the accuracy of the output. This leads to a requirement of parsimony in the selection of features. It should be the case that the features selected for use in *Cyberprints* are the maximum in a set of asymptotically diminishing contributions. Those features with little additional value should be discarded.

Figure 4.1 illustrates this point. In this hypothetical distribution, the explanatory variables have a continuous and monotonic distribution of contributions to the dependent variable. Real distributions are unlikely to be so clearly defined. In this example, explanatory variables are sorted in decreasing order of contribution along the horizontal axis, with the magnitude of their contribution along the vertical axis. To the left of the shaded region are variables which have a high contribution to the dependent variable. To the right are those which have little

or no contribution. Inside the shaded region is an area of decreasing explanatory power and somewhere within this area a delineation should be made between "useful" and "not useful" explanatory variables. All variables to the right of this delineation should be discarded.

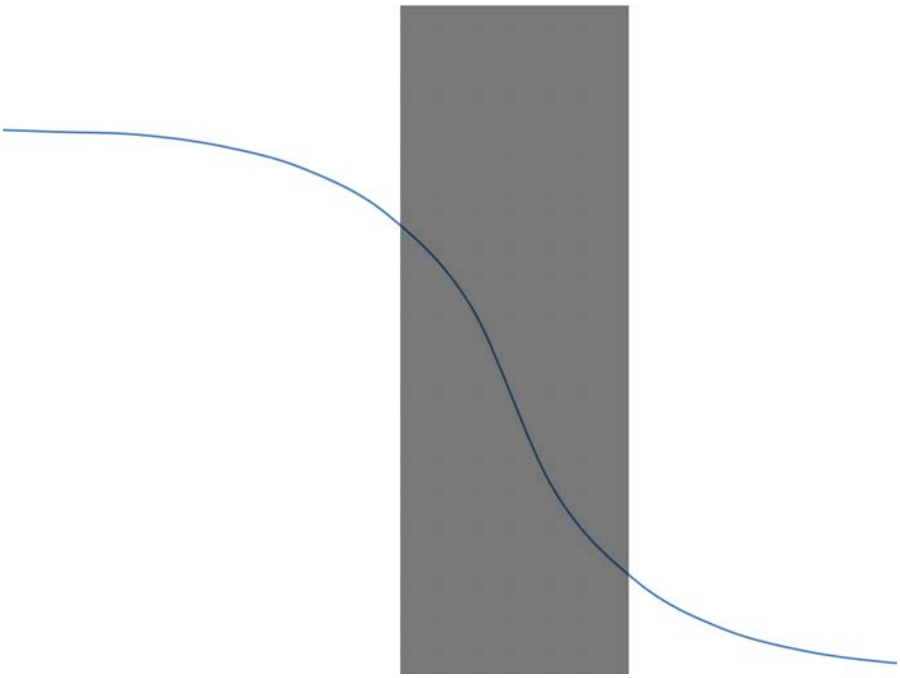

Figure 4.1 Theoretical Explanatory Variable Contributions

All features are given equal weight in the attribution algorithm. In the worst case, the selection of the wrong features could actually damage the attribution process by making it more difficult to separate actual identifying patterns from noise. For example, there could be a set of data in which one feature is a highly reliable indicator of the identify of an attacker, but a large number of others contain little or no attribution value. The relative contribution of this reliable indicator will be diminished, perhaps below the threshold of statistical significance. Thus, it is important to select features that are demonstrably reliable indicators and discard those unlikely or empirically shown to have little or no value.

4.3.3 Outlier sensitivity

Network traffic-based features are likely prone to occasional extreme values, due to all the potential sources of variation. Perhaps there was a momentary network glitch, which caused additional delay in traffic delivery, or the attacker's machine was momentarily heavily loaded. Maybe the attacker normally uses computer A for her attacks, but on one or two occasions needed to use computer B. For this reason, it is important that the selection of algorithm take into account the existence of such outliers.

It is difficult to enumerate all the potential sources of non-conformity. This might serve as a challenge for building a profile to survive such changes. It will be the goal to err on the side of accurate attributions. In this work, it will be preferred, like in the case above, for the attacker to be given redundant identities. If the error was made in the other direction, it would run the risk of incorrectly colliding identifications. Realistically, the same individual attacking from two different locations may very well be two different "attackers" as the motives behind the attacks might be different when at home or at work.

4.4 Feature Selection

4.4.1 Ideal feature characteristics

The description of the potential sources of variance given above is very optimistic. While it is true each of the variables mentioned might vary, based upon the origin of an attack, not all are detectable, reliable, and discernible. These three characteristics are what will be necessary to generate a profile of an attacker, based upon his network traffic.

The obvious first requirement is the feature must be detectable. This can be further broken down into two characteristics. First, the feature must be present in enough of the attacker's traffic that it is guaranteed sufficient to generate an attribution. Ideally, it will be present in all transmissions from the attacker, but this is not strictly necessary. Features that occur only in special cases (unless the special case in itself is a feature) are unlikely to be of much value. Second, the feature must be observable from the defender's vantage point. Further discussion

of this will follow.

The feature must also be reliable within the set of attacks from an attacker. Like diction or a fingerprint, it must express itself independent of the specific action of the source. Ideally, the feature will express itself in the same manner in every attack from the same attacker. However, it is possible that due to the combined contributions of a number of features, an attribution might still be possible even if a subset of the features are not always consistent.

Finally, the feature must vary to some degree *between* attackers. That is to say, the feature should reliably discern between two attackers. Like reliability, the entire set of identifying features need not vary between every attack as long as enough of them vary to generate a unique profile. However, in the ideal case, every attacker will exhibit every feature in a unique way. Due to the limited number of features which meet the detectability criterion, the requirements of reliability and discern-ability might often be in conflict.

4.4.2 Sources of features

Returning to Table 4.1, potential sources of features can be extracted from each of the communication layers. At the lowest level, the physical layer, are simply "bits on the wire" (or pulses in/on the carrier). Here, there is no notion of routing, applications, sessions, or anything but the physical representation of the signal. One could just as easily understand this level with an oscilloscope as a network traffic analyzer. Since a network connection is likely to consist of multiple physical links, it is unlikely that any identifying information at this level will be transmitted beyond the other end of the wire. Such local identification is useful for authenticating that a device is actually who it asserts to be at the link level. However, for the purposes of *Cyberprints*, where attacks are likely to come from many links away, such information is of no value. The physical layer fails the requirement of detectability.

Similarly, at the data-link level, most of the information is pertinent only to devices on the local network. The contents of an Ethernet frame header, for example, consist of a constant preamble (for synchronization), a constant start of frame delimiter, a source and destination Media Access Control (MAC) address, a Virtual Local Area Network (VLAN) tag, and an

EtherType field. The payload is then appended with a frame check sequence (cyclic redundancy check - CRC) and a 12-octet gap. The only features which might be valuable for attribution in this list are the source and destination MAC address, and the CRC. The others will be constant for every packet on the local network segment. However, these will be removed and replaced as soon as the frame leaves the local network. Unless the attacker is on the local network, data-link-level methods have no value. This layer also fails the requirement of detectability, as well as that of discernability.

Next is the network layer. Finally, at this layer some potentially useful features start to emerge. Because this is the first level at which features will survive outside the boundaries of the local area network (detectability), it automatically warrants consideration for identifying features. The most common end-to-end protocols at this level are the Internet Protocol (primarily version 4, but increasingly also version 6) and the Internet Control Message Protocol (ICMP). There are a number of other protocols at this level, but many of them are little used or are used only in special cases, such as information exchange between ISP routers. The requirements of reliability and discernability will require further analysis at this level.

Layered on top of IP are the two most widely used protocols on the Internet, the Transmission Control Protocol (TCP/IP) and the User Datagram Protocol (UDP/IP). Because TCP creates its own four-layer model of network traffic, distinct from the seven-layer OSI model, TCP and UDP do not clearly fall into either the network layer or session layer of the OSI model. TCP and UDP are both used to transmit data using IP addressing and seemingly fall into the network layer. However, TCP also contains a powerful state-based architecture able to establish, maintain, and terminate network sessions, which are more abstract than a single transmission. For this reason (and because TCP and UDP often serve similar purposes) both TCP and UDP might also fall in the session layer of the OSI model. Regardless of where they fall, they are bursting with potential features for use in attribution. Because of its complexity, TCP holds much promise. TCP clearly meets the definition of detectability (although there are ways for attackers to cloak these features). Analysis of features at this level has been shown to hold value for determining the operating system on the other end of a communication (Pouget

and Dacier, 2004; Spangler, 2003). It is reasonable to be optimistic about the chances for reliable and discernible features in these protocols.

The presentation and application layers are likely to be highly dependent upon the exact target of attack. However, they might also be indicative of the specific attack tool(s) used. In many ways, these layers can be considered together, as many applications handle both the presentation and user-interface tasks. For example, a web browser is responsible for negotiating encryption and compression in the transmission with the web server, which is clearly a presentation layer issue (i.e., data prepared for presentation). However,it is also responsible for rendering the data output from the decryption and decompression for use by the user. Because compression and encryption both essentially randomize the payload, they are unlikely to prove useful in attribution. However, if a particular kind of compression or encryption is used, this might represent a feature.

Assuming the application payload is detectable (i.e., observed after decryption and decompression, if applicable), it contains features with the most promise of discrimination between attackers. Attackers are likely to use different attack tools and methods, and this might be reflected in the data they transmit. However, it also is likely to have a great deal of variation even when coming from a single attacker. Therefore, reliability might be a concern. Only experimentation will tell how useful these data are.

Spread across multiple layers are features dependent upon the user or organization perpetrating the attack. Such things are sometimes called "Layer 8" in the OSI model — the human element of network communications. Such things as time of day when attacks are perpetrated, days when the attacks are not observed (e.g., holidays), timing in relation to current events, intentional or accidental self-identification by the attacker, or any number of other "soft" features might be useful for attribution. However, many of these are extremely difficult to quantify and process in an automated way. Therefore, only features which can be directly determined directly from the observed traffic, in a quantifiable way, will be considered for use in a *Cyberprint*.

Potential sources of features have been identified, as shown in Figure 4.2. In the next

section, features from each source will be enumerated.

- Network Layer: Internet Protocol, Internet Control Message Protocol
- Transmission Control Protocol, User Datagram Protocol
- Presentation Layer: Compression, encryption schemes
- Application Layer: Data related to tools or methods used by the attacker
- "Layer 8": Time of day, day of year, self-identification, other quantifiable "soft" features directly observable from network traffic

Figure 4.2 Sources of *Cyberprint* Features

4.4.3 Potential features

These sources of features now serve as the launching point for a more thorough enumeration of potentially useful information in network packets and streams. This distinction is important, and will play an important part in determining how features are analyzed. Due to the packetized nature of Internet communications, a single transmission might consist of a multitude of packets. For now, when features are discussed in the context of network streams, it is to mean aggregate statistics about the features of all packets contained within a single transmission. When the distinction between single packet features and network stream statistics might otherwise be ambiguous, it will be explicitly stated. Otherwise, one should assume that all features are features of network streams.

At the network layer, available features will be confined to those found in the Internet Protocol (IP) header or Internet Control Message Protocol (ICMP) header. These features are listed in Tables 4.2, 4.3, and 4.4. Many of these features are volatile or subject to manipulation in that they can easily be masked or varied by using a stepping stone in the attack. However, some or all might survive, depending upon the attack and attacker.

In the most simple case, the only information needed from these streams is the source IP address. However, if this approach was useful in general, there would be little discussion of

more advanced attribution techniques. So, it should be assumed this field will not directly identify the attacker. However, it might not be useless for identification. Perhaps an attacker uses the same set of intermediate nodes, or always launches the attack from a given country or Internet service provider. Even though the IP address is not uniquely tied to the attacker, it might still be useful in developing a profile.

Many other features in IP headers will be a function of the particular software used by the attacker. Different operating systems (and versions of the same operating system) might resolve ambiguity in network standards in different ways. This might show itself in the exact sequence in which options are set during a communication, for example, or perhaps the aggressiveness of fragmentation. Support for explicit congestion notification and congestion avoidance, as well as traffic prioritization through the DiffServ Codepoint field, might also be unique.

If an attacker is on an unreliable system or connection, is using poorly written attack tools, or is attempting to manipulate traffic en route, this might also be detectable. IP packets make use of a checksum value computed before transmission, so the recipient can verify the data were communicated correctly. In the examples above, there are reasons why this value might often be incorrect.

In IPv6, there is still a great deal of haziness in implementation. For example, the use of scoped extension headers — meant for consumption by the endpoint, routers, or each hop — might indicate a particular implementation. Currently, there is a discrepancy between Windows and Unix-based implementations. Windows clients randomize the address suffix while Unix-based software (by default) uses the hardware address of the sender. Additionally, the use of rarely used or experimental features (such as the Flow label) will likely be useful as part of a profile in IPv6 for some time as the protocol matures and is taken up by more users. However, its use relative to IPv4 is still quite limited. Thus, it is less likely to be part of many attacks.

The way in which hosts handle ICMP errors or the specific codes they return in response to probes are informative in the same way. However, unlike IP headers, ICMP messages are highly likely to be modified en route or blocked completely. They also have the disadvantage

Table 4.2 IPv4 Features

Label	Content
IP Version	4
Header Length	Length of entire header
DiffServ Codepoint (DSCP)	Traffic prioritization
Explicit Congestion Notification (ECN)	Congestion avoidance
Length	Total packet length
Time to Live (TTL)	Number of additional hops before packet is returned to sender
Identification (ID)	Pseudo-unique identifier
Fragmentation Flags	Concerning the reassembly of fragments
Segment Offset	The distance of this packet from the first packet in the stream
Next Protocol	The protocol (e.g., TCP) encapsulated by this header
Checksum	An integrity check
Source/Destination	IP addresses of stream endpoints
Options	Additional (optional) header data

of usually being most useful in active probes of a network and are infrequently used in attacks (at least recently).

Similarly, the values in the TCP or UDP headers might be useful. Tables 4.5 and 4.6 enumerate these parameters. Like IP and ICMP, there is a possibility that part or all of the profile will not survive retransmission along a multi-step path. However, the complexity of TCP, in particular, makes it likely that idiosyncrasies in protocol implementations will exist.

Source port numbers, for example, are supposed to be random (at least between 1025 and 65535). However, attack tools might manually craft network packets that use the same port or range of ports for this purpose. They might also always try to attack the same range of destination ports — for example hitting 80 (the common HTTP port), 443 (the HTTP over SSL port), 8080, 8081, 8443, 8888, and 10,000 (some common ports used by other web software).

Initial sequence number generation has received much attention in the last ten years, as many operating systems traditionally simply increment this number for each connection. This

Table 4.3 IPv6 Features

Label	Content
Version	6
Traffic Class	Combination of DSCP and ECN (above)
Flow Label	Uses being defined, intended for real-time communications
Length	Size of payload
Next Header	Type of the next header in the chain
Hop Limit	Analogous to TTL (above)
Source/Destination	IP addresses of stream endpoints
Extension Headers	
Extension Header Length	Total length of extension header
Options	Type of the next header in the chain
Routing Type	Indicates special routing treatment
Fragment Offset	The distance of this packet from the first packet in the stream
More Fragments	Indicates whether or not more fragments are to follow
ID	Pseudo-unique identifier used for reassembly
Authentication/Encryption Encapsulating Payload	Used for IPSec, the security extensions to IP

Table 4.4 ICMP Features

Label	Content
Type	Indicates purpose of transmission
Code	More detail regarding the purpose
Checksum	An integrity check
Variable Content Field	Varies based upon type and code

made it easy to mount a "man-in-the-middle" (interception and modification of traffic) attack. It also made it easy to peer behind firewalls to determine the number of devices hiding behind the translation. Recent operating systems randomize this value for each connection or at least make it more difficult to predict. However, the pattern of ISNs in an attack might reveal something about an attacker's environment.

Window sizing is a process highly dependent upon the connection endpoints. The TCP protocol attempts to make the most optimal use of a connection medium by sending packets as quickly as possible from end-to-end. In the case of links at (or over) capacity, links which are unreliable and drop packets, or recipient applications which cannot keep up with the traffic (as a few examples), the sender and receiver might need to scale down the speed at which traffic is sent. Conversely, a sender might initially start sending traffic at a rate below the capacity of the medium and recipient, requiring a gradual increase in rate until the capacity is reached. At any point during this process, competition for capacity might enter the picture, requiring an adjustment of the window size.

Like options in IP, flags in TCP might be used in novel ways. While there are strictly adhered to sequences for initiating and terminating connections (SYN, SYN-ACK, ACK; FIN, ACK, FIN, ACK) and some other situations, the use of some of the other flags such as URG, PSH, or ECE are highly dependent upon the systems involved. This kind of identification has been shown useful to identify the operating system of a remote host (Spangler, 2003).

Timestamps allow for the calculation of average round trip time (latency), as well as jitter (variation in latency) for a session. These values might be useful, if an attacker always uses the same type of connection(s) or relays traffic through several intermediate nodes. Each connection and node adds some processing delay while the traffic is received, analyzed, perhaps modified, and re-transmitted.

There is also the possibility that some of the "layer eight" features of the attacker will be revealed through the timestamp. Do attacks always happen during business hours in a particular time zone? Does the attack stop during Ramadan, Christmas, Hanukkah, or some other holiday that might suggest a particular region or country? Did the attack correspond

with the timing of a conventional (non-cyber) attack? Of course these last two would require both external information about the attacker and current events, as well as a broader view of the attack history, but the potential remains.

There is also likely to be a correlation between geographic distance and latency, but it is not as strong as one might expect, due to the many different kinds of connections used. For example, sending traffic between two buildings next door to each other using a satellite connection would entail tens or hundreds of milliseconds of latency, but sending traffic from New York to Beijing might take only a few milliseconds using underground and undersea optical cables. However, one can expect more connections and longer connections (the former more than the latter) should have an overall positive correlation with latency. The level of development within a country might also have a correlation — more developed countries are likely to have higher quality and lower latency links.

The presentation and application layers are likely to vary, based upon the application endpoints. However, there is no guarantee there will be any interceptable indication of what schemes are used. Data transformations at this layer are things such as (de)compression, (de)encryption, or format conversions.

If known, the choice of algorithm for these tasks might be informative, but this information is not likely to be present. In many cases the application endpoints will be designed or configured with a certain algorithm in mind, removing the need to signal the algorithm being used in the transmission. However, certain protocols, such as the Hypertext Transfer Protocol (HTTP), have optional presentation layer components that might be observable.

If an application makes use of encryption or compression at this layer, observation of the traffic in-motion is unlikely to be of much use. A good encryption scheme will generate a randomized output for a given input (dependent upon the encryption scheme, of course). Thus, even the same data sent twice might not appear the same both times. Similarly, compression is highly input dependent, and even a small change in input could conceivably result in a very different compressed payload. Therefore, from a profiling perspective, there is little value in encrypted or compressed payloads, and analysis should encompass only the network layer.

Table 4.5 TCP Features

Label	Content
Source/Destination	Port numbers on endpoints. Many destination ports are standardized.
Sequence Number	Number of packet in sequence, beginning with Initial Sequence Number (ISN).
Acknowledgment Number	Next sequence number expected by sender.
Offset	The distance of this packet from the first packet in the stream.
Window Size	Number of bytes the sender is willing to receive in a single packet.
Checksum	An integrity check.
Urgent Pointer	Offset of last urgent packet in stream (if URG flag is set).
Flags	
NS	Nonce sum, protects against malicious concealment of ECN-marked packets.
CWR	Congestion window reduced, confirmation of ECE (below) receipt and window size adjustment.
ECE	Explicit congestion notification echo, notification of ECN support or experienced congestion.
URG	Urgent data.
ACK	Acknowledgment.
PSH	Push buffered data to application.
RST	Reset session.
SYN	Synchronize sequence numbers (sometimes: Synthesize session).
FIN	Terminate (Finish) session.
Options	
No-Operation	Aligns fields on word boundaries.
Maximum Segment Size	Largest transmittable packet.
Window Scale	Magnifies the value of the window size field.
Selective Acknowledgment Permitted	Notify of support for confirmation of partially complete transmissions.
Selective Acknowledgment	Confirmation of partially complete transmission.
Timestamp and Echo of Previous Timestamp	Compute round-trip time and protect against wrap-around of sequence number.
Alternate Checksum Request	Request a different checksum algorithm.
Alternate Checksum Data	Value of alternate checksum.

Table 4.6 UDP Features

Label	Content
Source/Destination	Port numbers on endpoints. Many destination ports are standardized.
Length	Size of entire datagram.
Checksum	An integrity check.

However, if the payload is clear-text and uncompressed, it is perhaps as close to literary analysis as any part of *Cyberprinting*. There are no general features of payloads that can be used for analysis, since the payload is by definition a free-form field. However, existing techniques, such as n-gram analysis, or overall statistics, such as length or distribution of binary values, might prove useful.

CHAPTER 5. METHOD

5.1 Datasets

To perform this experiment, methods of generating and dissecting a variety of packets were needed. To generate the traffic, a set of virtual machines was created, each running one of Debian Etch, Red Flag Linux, FreeBSD 5, FreeBSD 9, Windows XP, or Windows 7. Each of these machines had three attack tools installed: THC Hydra, Nmap version 4, and Nmap version 5. Some difficulty was incurred to run identical versions of each program in each operating system. This largely determined the software applications used. THC Hydra was used to perform an HTTP and IMAP brute force attack against the Apache and Dovecot services running on the target. Nmap (both versions) was used to perform a port scan of the target.

The network topology for this test was virtual, as well. Each machine/application combination was used to attack the target virtual machine, which was running Debian Etch. The attack traffic (and only the attack traffic) was captured on the target for later analysis. Each machine/application combination was used to attack the target directly (i.e., no intermediate network devices) through a FreeBSD 9 virtual machine acting as a router (i.e., no translation) and the same FreeBSD 9 virtual machine acting as a network address translating router (i.e., all traffic appears to originate from the same source address on the external side of this router).

For the purposes of analysis, the mergecap utility was used to combine these data into files that contained all traffic for each operating system and application. The tcprewrite program was then used to undo the network address translation (for the data sets in which this occurred) of the source address, so that a consistent identifier was present for analysis. During analysis this value was used only as a tag and was randomized before being used in any statistical

methods.

The feature set developed included all header parameters of the IPv4 and TCP headers collected using the open source JNetPcap library. A full listing of these features can be seen in the results section. Each packet capture file was passed through a TCP reassembly step, and packets were assigned into streams by their source and destination.

The features from each stream were combined into a database, which also included the host, application, and level of obfuscation (direct, routed, or translated) for the stream. This database was used as an input to a multivariate analysis of variance (MANOVA) that computed the significance of each feature for differentiation of the operating system, application, level of obfuscation, and the interaction of operating system and application.

5.2 Analysis Method

Now that a list of potential features has been created and a test dataset generated, how are the data processed to reveal a profile? Like the features, any analysis method chosen for this task must be sufficiently sensitive to detect differences between attackers, but sufficiently reliable to detect the same attacker in multiple situations. It should also be possible to tune the sensitivity of the method and ideally to give some bounds on the level of certainty of an attribution.

Given the limited amount of data available to create a profile, this seems like a tall order. The central thesis of this work is that by using statistical methods similar or inspired by those used in literary analysis, some level of discernment between attackers can be made, based upon only the traffic visible to target of the attack. The statistical profile created by the analysis method is the *Cyberprint* of the attack and, ideally, the attacker.

5.2.1 Analysis of variance

Before delving too deeply into the signatures present in the dataset, first it is worth evaluating whether sufficient information exists to make a discernment. For this reason, the first step to determine the feasibility of *Cyberprints* is to run an analysis of variance. Specifically,

an analysis will be performed to determine the effect of the operating system, attack tool, obfuscation method, and the interaction of operating system and attack tool upon each identified feature. Because this involves multiple independent variables and the interaction of two of them, a Multivariate Analysis of Variance (MANOVA) will be performed.

Based upon this output, it will be determined which of these factors have a non-zero, but not significant impact, upon each of the features. This will allow for a demonstration of the effects of noise in later examinations. Similarly, those features that do not vary, and thus have zero impact, can be separated from the overall features.

5.2.2 Feature extraction (Principal Component Analysis)

After the data are collected, they need to be normalized suitable for statistical analysis. Additionally, it will improve the performance of analysis methods if the dimensionality can be reduced. To achieve this, Principal Component Analysis (PCA, sometimes called Factor Analysis) will be used. An extensive treatment of the theory and mechanisms of PCA is given in the text by Ian Joliffe (Joliffe, 2002), but a brief summary will be provided here for the unfamiliar reader.

A good dataset to use for this explanation is one of biological measures of a set of individuals. For example: height, weight, blood pressure, blood glucose, heart rate, and ethnicity. When measured by a physician, these are treated as separate features. One does not calculate blood pressure as some function of height and weight. However, it is well-known that these measures are not independent by any means. The ratio of weight to height, perhaps combined with some element of ethnicity, is known to be a fairly reliable predictor of hypertension and other cardiovascular diseases. The exact link, or causal factor, is subject to much debate, but most would agree it is there.

Using PCA, one could take these same six features of a sample set and identify some underlying component X, a function of these features. The exact causal mechanism will of course not be identified, so one is free to label this factor as seen fit. However, it can be said when this new component varies, the directly measured features co-vary with it. This is the

essence of PCA — covariance.

The primary use of PCA is to reduce the dimensionality of highly-dimensional datasets. One can imagine in the example above, there might be any other number of biological measures. If there were twenty measures taken and it was desired to find a pattern to help determine the causes of cardiovascular disease, one would need to consider all twenty measures, which amounts to a feature space in twenty dimensions. Humans are not good at seeing patterns in data in more than three dimensions, and even computational methods become quickly bogged down as dimensionality increases. One might also assume that some of the features have little impact on the dependent variable in question.

PCA takes the covariance matrix of all of the features across all observations and performs an eigendecomposition. The output of this transformation is a matrix of eigenvalues and eigenvectors. Each eigenvector represents the coordinates of each of the original points in a newly constructed component space. The eigenvalue corresponding to that vector indicates the relative contribution of that component to the overall variance of the dataset. After ordering these vectors by descending magnitude of the eigenvalues, one can select some number of dominant vectors as the *principal components*. The cutoff is dependent upon the level of variance desired to be accounted. In highly noisy datasets with many features, the number of components required to account for a satisfactory amount of variance might be nearly the same as the number of features, and little benefit is had. On the other hand, this new dimensionality might be a significant reduction if a carefully selected set of features which have a high level of dependence.

A simpler example might help illustrate this point more clearly. If one were to measure the horsepower and sales volume of every model of car on a dealer's lot, one would expect some correlation. In fact, one might expect a negative correlation, not because people do not want to buy fast cars, but because fast cars are more expensive. Imagine a plot of horsepower versus sales volume in two dimensions. It is not likely that this would be a strict $y = N - x$ function, but if one were to draw the line of linear regression through the dataset, it would probably approximate it. Now, imagine rotating the dataset so this regression line is horizontally level.

One might now say that to move to the right on this line would be predictive of a higher car cost and to move left the inverse. In other words, as horsepower increases and sales volume decreases, a new component — hypothesized to be cost — increases.

If a third feature, gas mileage, were introduced, it would likely co-vary in a similar fashion. But, there might also be another component now identified, which one could perhaps identify as weight. Sticking with the two-dimensional example above, a third dimension would now be added for this new component, and PCA would place the axis of this new dimension orthogonally through the two-dimensional plot in a way that maximizes the amount of the covariance not accounted for by the first component accounted for by the second component. This process is repeated until all components are accounted, and one can then choose the number of components to retain for explanatory purposes.

5.2.3 Cluster Analysis

Recall that the output resulting from PCA is a set of vectors which represents the coordinates of the original points in a a new component space. If one looks at the example of biological measures above, it can be imagined that two individuals with a similar height and weight might have more in common than two individuals who differ greatly in these measures. Therefore, one can say these two features provide some information about the similarity between the two individuals. Like the independent variables in feature space, one can look to proximity as a measure of similarity between two observations in component space. Visualizing and measuring this distance, like the space in which it resides, is beyond the abilities of humans for anything beyond three dimensions. Thus, a general method is needed to determine proximity in n-space.

The simplest method to achieve this is by extending Euclidean distance to n-dimensions. Just as distance can be computed in 2-space as $d^2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$, distance can be computed in n-space by adding the additional coordinates. This works well for two points, but what if instead of measuring the Euclidean distance, it is desired to determine the most similar points from a large set of points in n-space? The relationship is called a "cluster" and

a single dataset might have any number of clusters of similar points within it.

The Euclidean distance between every pair of points can be calculated and one can set a threshold for how close two points must be to be defined as "closely related". There are three problems with this. Primarily, it is computationally inefficient to compare every pair of points ($O(n^2)$). Second, determining this threshold is a subjective measure and one must experiment with different values to "tune" the algorithm to the application. Third, this method does not perform well on clusters not evenly distributed and equi-distant from a centerpoint, or datasets in which the clusters are different sizes. A cluster that is skewed or stretched along an axis will most likely be identified as two or more different clusters by such a method. Similarly, two adjacent and similarly shaped, but separate, clusters might be identified as one.

To resolve this difficulty, a number of clustering algorithms have been developed over the last several decades. The final step in the analysis of *Cyberprint's* potential is evaluating the performance of a selection of these algorithms on the PCA output. To this end, each clustering algorithm described below will be used to cluster the data, and the correctness of fit will be evaluated. To aid in this analysis, the ELKI (Achtert et al., 2008) Java application will be used. This program contains a variety of clustering methods and evaluation functions ready to use with an input file.

For comparison purposes, a simple "by label" clustering will be performed. This method simply assigns each point to a cluster, based upon the label given it. For this and the following algorithms, clustering will be compared to operating system labels, attack tool (application) labels, and the combined operating system/application label.

The K-Means algorithm (Hartigan and Wong, 1976) is one of the simpler algorithms evaluated. The algorithm starts with K points selected at random from the dataset (a modified K-Means++ algorithm performs this selection to maximize dispersion). Every point in the dataset is assigned to one of these points, according to which is closest (Euclidean measure). This effectively partitions the space into Voronoi cells in n-dimensions. Then, for as many iterations as desired, the algorithm will find a point as close to the center of each cell as possible and repeat the assignment. After these iterations are finished, the clusters are simply the

assignments of points to the nearest center point. This algorithm is well-suited to datasets where the clusters are of similar size and shape, and the number of clusters is known *a priori*. Neither of these assumptions is likely to hold, in general, for *Cyberprints*.

In the Shared Nearest Neighbor algorithm (Ertoz et al., 2002), a set of nearest-neighbors is computed for every node. A threshold is set regarding the number of nodes allowable in this set. After all of these sets are computed, a comparison between every set of nodes is made and any two nodes which have each other in their nearest-neighbor set are identified as being in the same cluster. This has the advantage of including outlier points that might be ignored by a simple nearest-neighbor approach (such as K-Means). The potential for branching clusters that do not originate from a single central point allows for the detection of irregularly shaped or unevenly sized clusters. It also has an advantage over methods in that one does not need to specify a target number of clusters. In SNN the work of determining what constitutes a cluster and how many are present is part and parcel of the clustering technique. While it unfortunately does not address the complexity requirements, this is still a more useful approach for *Cyberprints*.

Other algorithms that would have been evaluated, had the first two performed well, included Expectation-Maximization (Dempster et al., 1977), DBSCAN (Ester et al., 1996), OPTICS (Ankerst et al., 1999), and SUBCLU (Kailing et al., 2004). However, as will be explained in the next chapter, analysis along this avenue was terminated before reaching these more complex methods. Instead, a different path was taken.

5.2.4 Kolmogorov-Smirnov Test

An alternative method of evaluating the similarity of two network streams is to compare their features directly. There are two disadvantages of this method. The first is that the dimensionality of the dataset to be analyzed will be much higher, leading to increased computational and storage requirements. Second, it leaves a great deal of room for noise, since there is no mechanism of ranking features by their contribution (see the discussion of parsimony above). Additionally, there is no reason to expect that any clustering methods would work better in

feature-space than component-space. In fact, one might expect them to perform more poorly for the reasons above.

To analyze the test datasets for *Cyberprints*, there is a way around these problems. The number of features, and thus dimensions, can be greatly decreased by leaving out the features with little to contribute. Again, there is no automatic way to accomplish this. However, it is possible to use knowledge of the network protocols involved and the test dataset to focus the analysis.

By selecting a subset of the features to use for this analysis, this method might become tractable. For each of these features, a frequency distribution of values for each type of operating system and attack tool in the dataset can be generated. For the purposes of evaluation, ten random subsets will be chosen from each class. These subsets can be compared against others in their own class, as well as other classes, to determine whether these distributions might present an identifying profile.

To do so, however, a metric is needed to decide how well two frequency distributions match. For this, the Kolmogorov-Smirnov test can be used. This statistic provides the lowest upper bound (supremum) of the difference between two cumulative frequency functions. In other words, one converts the frequency distributions into cumulative frequency distributions. For each frequency, one then computes the difference between the two functions. The biggest divergence is D, the Kolmogorov-Smirnov statistic.

For the purposes of rejecting the null hypothesis, this value must be greater than a critical value. However, for the purposes of evaluating its potential in *Cyberprints*, these values will be averaged across the selected features. A matrix can then be constructed and the relative similarity of within-class and between-class average D values can be examined. One would expect the differences within a class to be consistently less than those between classes.

CHAPTER 6. RESULTS

6.1 Multivariate Analysis of Variance (MANOVA)

The MANOVA test compares variance within and between groups to determine the effect of each explanatory variable upon each dependendent variable, as well as the effect of interactions between explanatory variables. This statistic allows for the elimination of several features from further analysis. First, however, several features arre removed that were meaningless in the test dataset.

The initial set of features, based upon the selection of TCP/IP traffic for testing and the capabilities of JNetPcap to extract them, are shown in Table 6.1. After running a MANOVA, several features were found to have no variation. These included the IPv4 Average Header Length; IPv4 Average Offset; IPv4 Number of More Fragment Flags; Minimum, Average, and Maximum TCP Sequence and Acknolwedgment numbers; and IPv4 Checksum Error Rate. The source and destination address were removed, since in this test case the destination is constant and the source is randomized before analysis. The destination port was removed, since it is constant between sets.

As shown in Figure 6.1, MANOVA also shows the strength of each feature to discriminate the operating system, application, and operating system x application interaction. Every feature not already eliminated was useful for discriminating the application (at a 95% confidence interval). The minimum ID value, number of IPv4 ECE and ECN flags, average Maximum Segment Size, average Timestamp interval, average Urgent offset, number of CWR flags, number of TCP ECE flags, number of Urgent flags, and variations in any of the features were not useful for discriminating the operating system. The same set, with the exceptions of the minimum ID value, number of IPv4 ECN flags, and variation in sequence number were not

Table 6.1 Initial Set of Features for Consideration and their Relevance

Feature	Operating System	Application	OS x App
IPv4: Source Address			
IPv4: Destination Address			
IPv4: Average Header Length			
IPv4: Average Packet Length	X	X	X
IPv4: Average Offset			
IPv4: Minimum ID		X	X
IPv4: Average ID	X	X	X
IPv4: Maximum ID	X	X	X
IPv4: Variation in ID	X	X	X
IPv4: Minimum TTL	X	X	X
IPv4: Maximum TTL	X	X	X
IPv4: Average DSCP		X	X
IPv4: Number of ECE Flags		X	
IPv4: Number of ECN Flags		X	X
IPv4: Number of Don't Fragment Flags	X	X	X
IPv4: Number of More Fragment Flags			
IPv4: Checksum Error Rate			
TCP: Source Port	X	X	X
TCP: Destination Port			
TCP: Average Maximum Segment Size		X	
TCP: Average Urgent Offset		X	
TCP: Average Window Size	X	X	X
TCP: Average Timestamp Interval		X	
TCP: Variation in Timestamp		X	
TCP: Minimum Acknowledgment Number			
TCP: Average Acknowledgment Number			
TCP: Maximum Acknowledgment Number			
TCP: Variation in Acknowledgment Number		X	
TCP: Minimum Sequence Number			
TCP: Average Sequence Number			
TCP: Maximum Sequence Number			
TCP: Variation in Sequence Number		X	X
TCP: Number of ACK Flags	X	X	X
TCP: Number of CWR Flags		X	
TCP: Number of ECE Flags		X	
TCP: Number of FIN Flags	X	X	X
TCP: Number of PSH Flags	X	X	X
TCP: Number of RST Flags	X	X	X
TCP: Number of SYN Flags	X	X	X
TCP: Number of URG Flags		X	

useful for discriminating the operating system x application interaction. This is summarized in Table 6.1

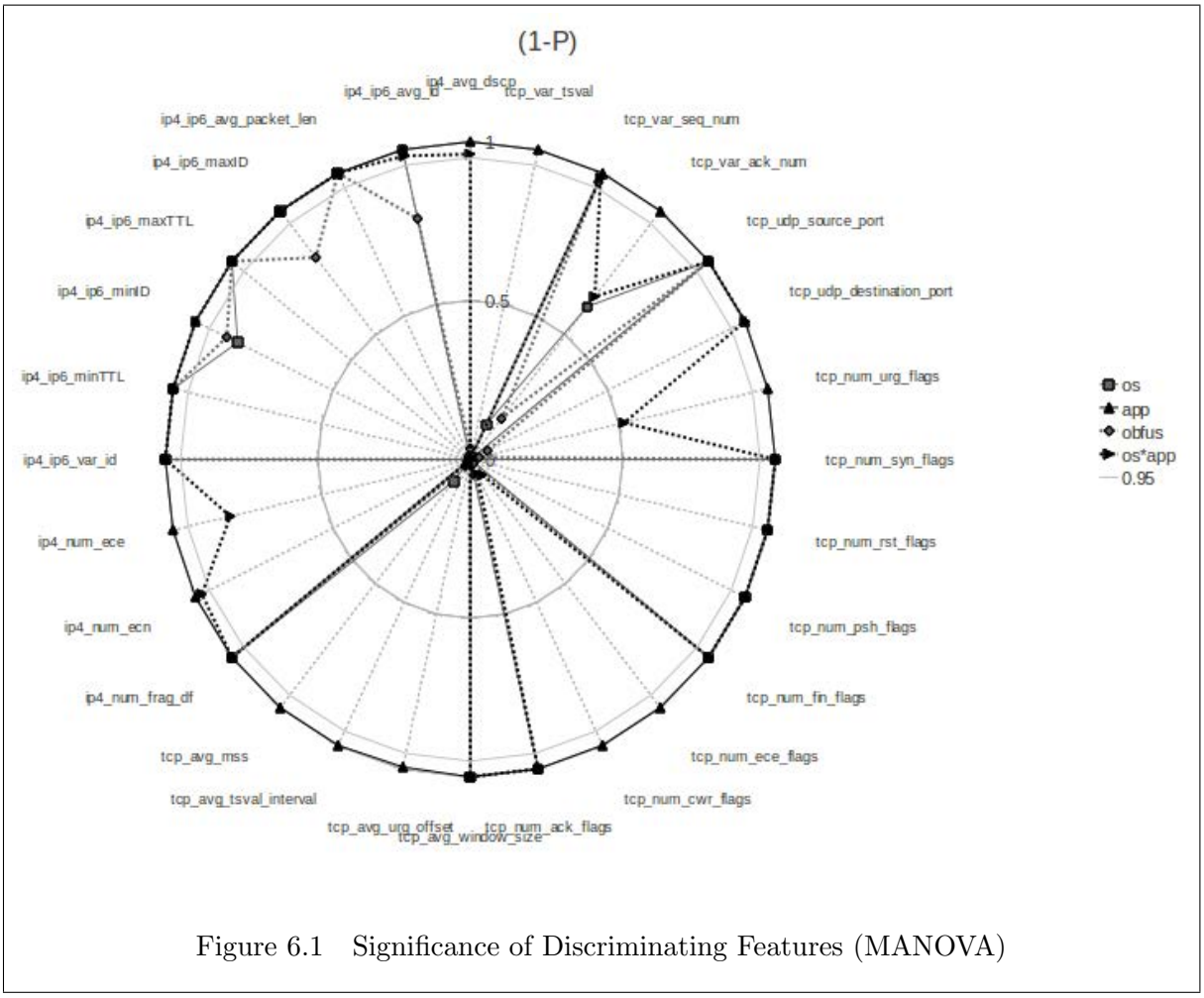

Figure 6.1 Significance of Discriminating Features (MANOVA)

6.2 Principal Component Analysis and Clustering

The second phase of analysis, principal component analysis, proved not to reveal any useful information. After computing the principal components for the set of features identified as relevant above, it was found that nearly all (greater than 99.99%) of the variance was accounted for by a single component. As shown in Figures 6.2 and 6.3 (generated with ELKI), the distribution of observations within this component space was nearly uniform, leaving no hope for clustering. Even though the individual features have significance in discriminating the

operating system and application (as shown in the MANOVA), it seems the combination of them is too noisy for PCA to handle.

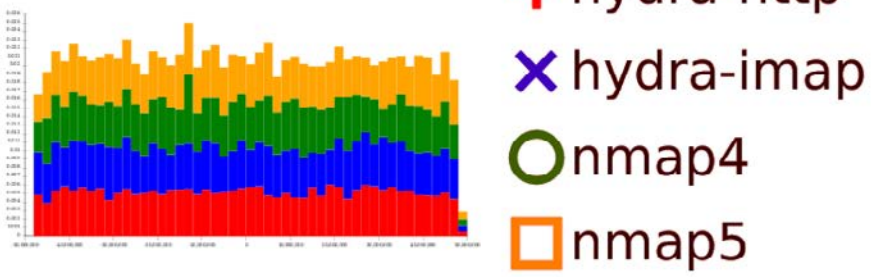

Figure 6.2 Output of Principal Component Analysis, by application

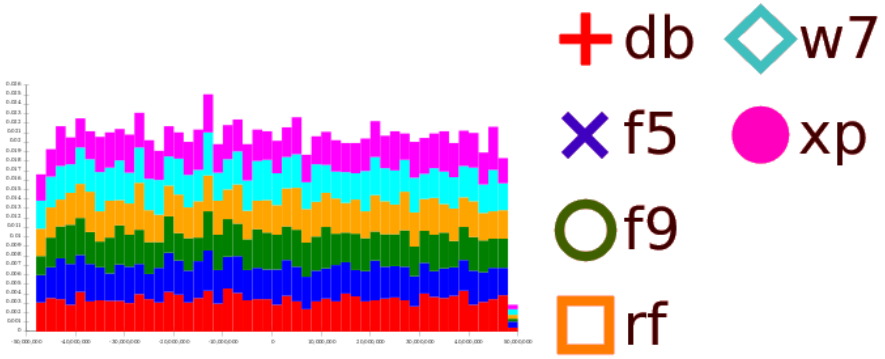

Figure 6.3 Output of Principal Component Analysis, by operating system

To take a step back, there was still the possibility that clustering might work within feature-space, even if it did not work in component space. Again, using the set of relevant features, a simple "by-label" clustering was run on the data. Visualizations of this output are shown in Figures 6.4 and 6.5. The top row of histograms shows the distribution of values within each dimension. Using this additional tool, features which showed a single peak or uniform distributions were removed from the analysis. This leaves the 12 dimensions shown in the previously referenced figures (which also include the source address, left in for comparison). These features were: the number of ACK flags; the minimum, average, and maximum ID; the number of Don't Fragment flags; the number of FIN flags; the number of PSH flags; the source

port; the average packet length; the minimum and maximum TTL; and the window size.

The lower panels in these figures shown cross sections comparing every pair of dimensions. Within these, one can see evidence of clusters. The largest panel shows a plot in 3-dimensions of the first three dimensions. While this leaves out most of the features, it supports the idea that clusters might be identifiable within the data.

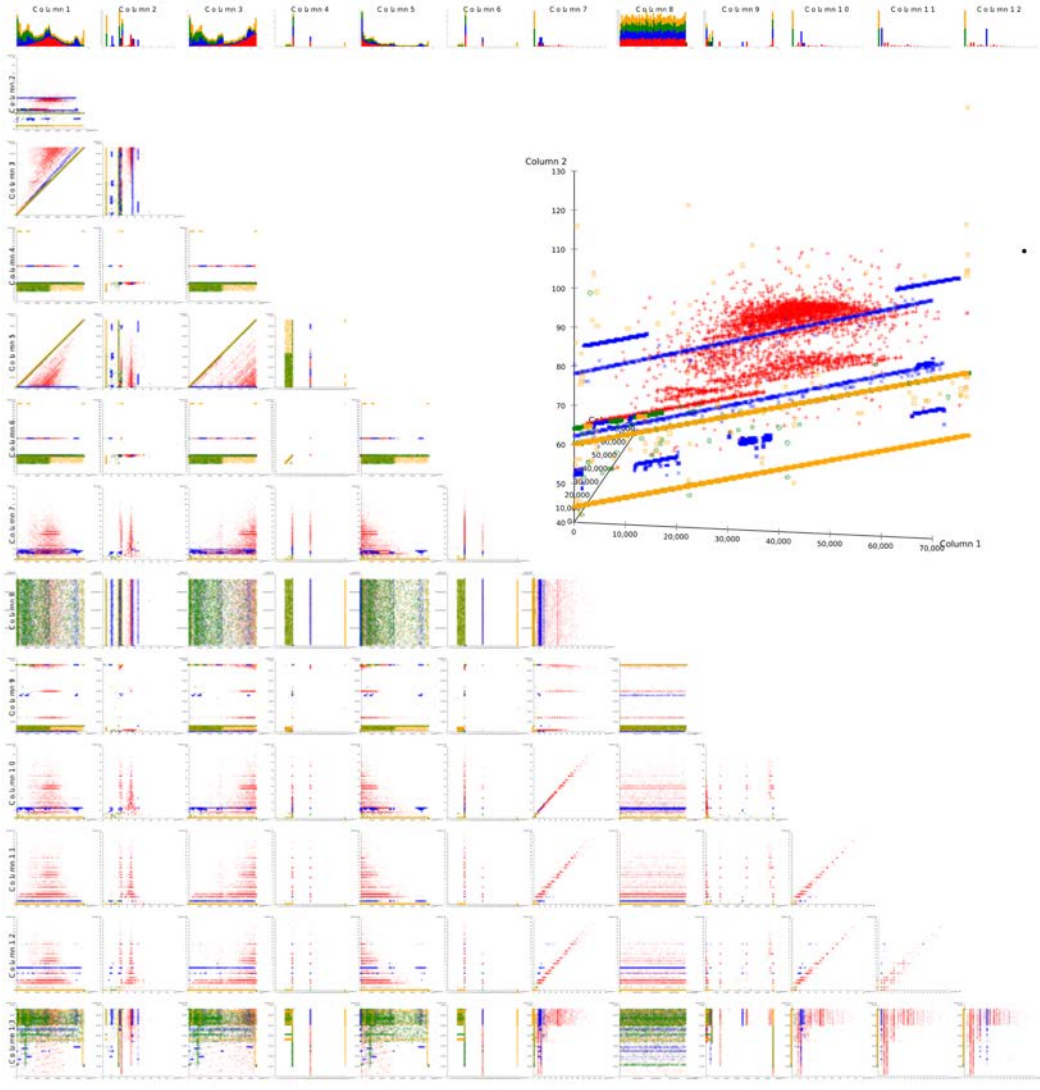

Figure 6.4 Analysis of 13-dimensional feature space of significant features, by application

The K-Means and Shared Nearest Neighbor clustering algorithms were then run on this

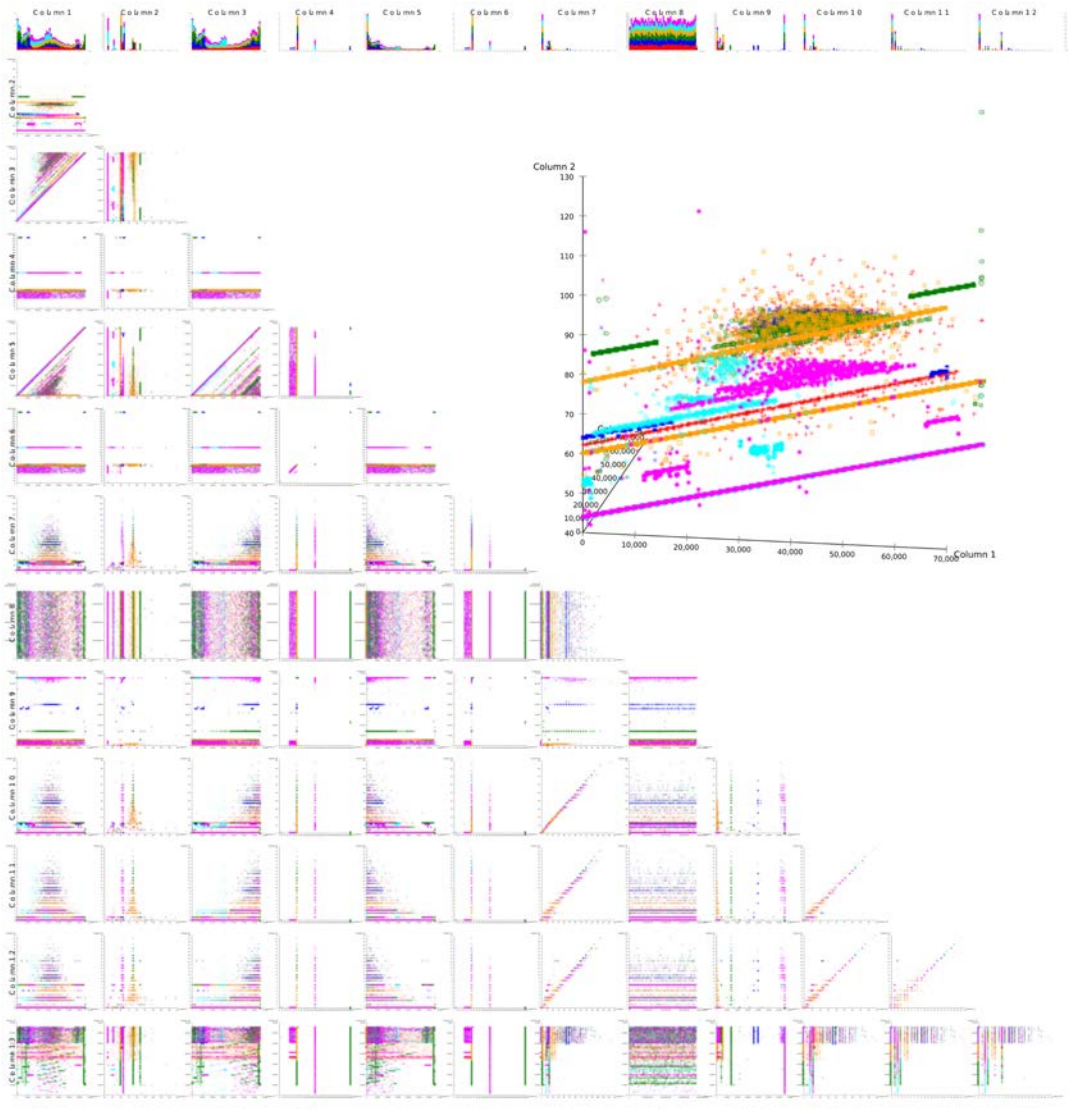

Figure 6.5 Analysis of 13-dimensional feature space of significant features, by operating system

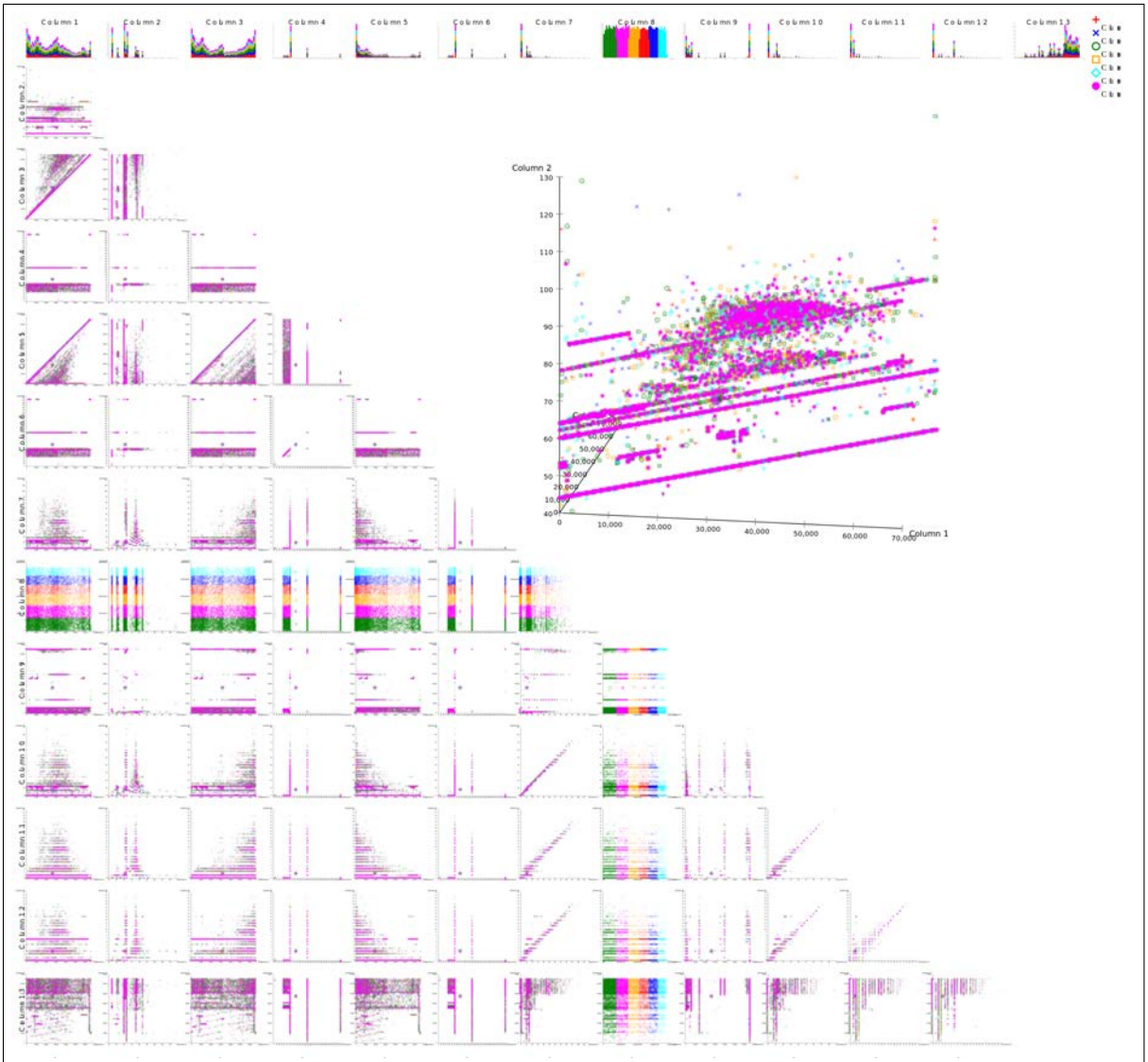

Figure 6.6 K-Means analysis of 13-dimensional feature space of significant features (k=6, 10 iterations)

same set of relevant features. The output is shown in Figure 6.6 for K-Means. As shown, the clusters sometimes overlapp, are irregularly shaped, or are disjoint. This type of dataset is known to be poorly suited for K-Means, and thus an SNN analysis was run. However, SNN was unable to identify any clusters, even with great experimentation with neighborhood size and cluster density. Most likely, this was due to the fact that the ranges of values vary widely between dimensions — a problem that PCA would fix, if it worked otherwise. Other clustering algorithms, even those suited for higher-dimensionality datasets, would likely suffer these same issues. Instead of continuing down the path of clustering algorithms, it was decided to try a new approach.

6.3 Kolmogorov-Smirnov Analysis

Principal Component Analysis, which otherwise would have evened out the large range of values in the explanatory variables, did not prove useful. These ranges of values are too broad for an overall clustering to be useful. Therefore, a method is needed that allows for aggregating a similarity comparison of each individual feature. Using the Kolmogorov-Smirnov statistic, this is possible.

For each operating system and application, ten random groups of 100 streams were selected. From each of these group, the cumulative distribution function (CDF) was computed for each feature. An example of this is shown in Figure 6.7. Representative charts for each class are included in Appendix A.

As evidenced in these figures, there are definite distinctions in the CDF for each class. As expected, different operating systems and applications make different choices regarding the generation of these features. The next step is to determine if the differences between classes for these features are greater than the differences within classes.

To show this graphically, a heatmap-type illustration is used. Figure 6.8 uses this method to show the overall average of the comparisons for each feature (more discussion in a moment). Along each axis are rows and columns for each random sample, sorted by class. The value at the intersection of each row and column is the Kolmogorov-Smirnov D-value derived by

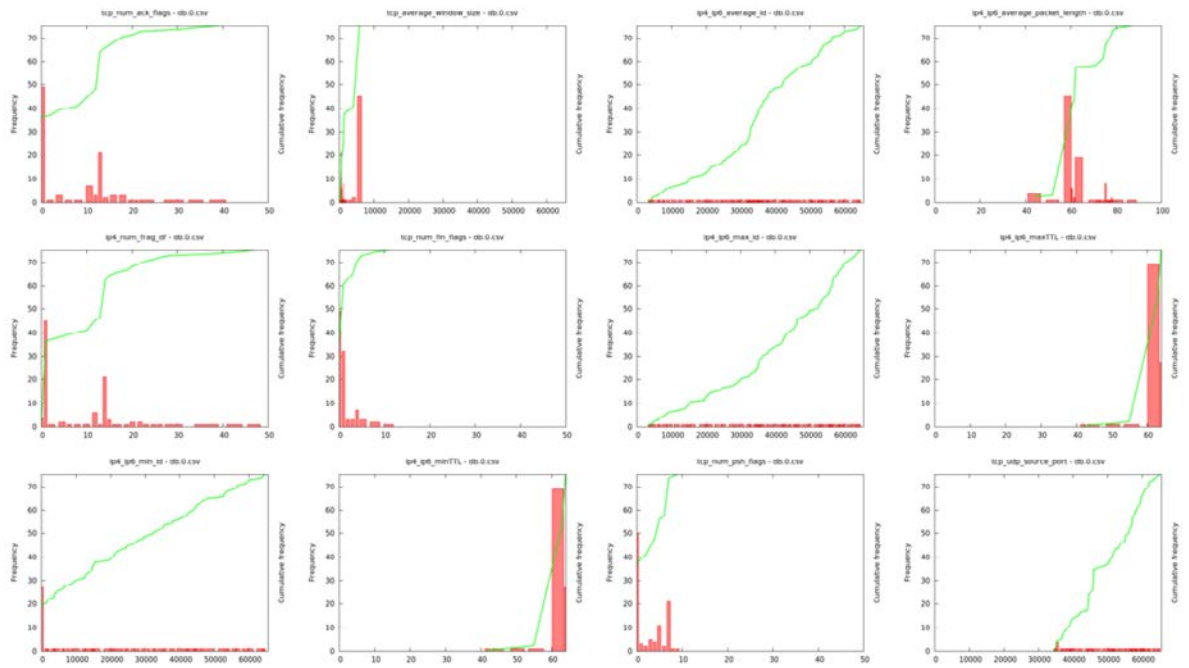

Figure 6.7 Feature Distribution Histogram: Debian Linux

comparing these two samples. These values are hidden in the heatmap, in favor of a color gradient from green (most similar) to red (least similar), which is easier to analyze visually. The heatmaps for Kolomogorov-Smirnov comparisons of individual features are included in Appendix B.

One would expect that for each feature lower D-values would be found along the diagonal. Obviously, a distribution compared to itself will have a value of zero for this statistic. Random samples taken from the same class should also have low (if not zero) D-values. The perfect outcome would be a set of ten, 10x10 green squares aligned on the diagonal $y = 100 - x$, accompanied by 90 red squares off the diagonal. This would represent a perfect discrimination between classes.

As shown in the remaining figures in Appendix B, some of the features are more reliable for this method than others. Having already eliminated features that have little statistical discernment value in the previous steps of MANOVA and histogram analysis, the list can be further refined by removing features that have a poor showing in the Kolmogorov-Smirnov

comparisons. Since this analysis has been visual so far, a threshold needs to be set to remove the subjectivity from this process. For lack of a better threshold, features that display more than 20 off-diagonal 10x10 blocks (including symmetrical mirrors) that are at least 60% green will be eliminated. This is probably an overly restrictive selection and is not to say that these last few eliminated features have no value whatsoever. In fact, the test for statistical significance of the D-value is not used (as the primary concern is the relative values), so these features might have value upon further inspection.

After this criterion is applied, the minimum IPv4 identifier, average IPv4 identifier, maximum IPv4 identifier, minimum TTL, maximum TTL, TCP FIN flags, and TCP PSH flags are eliminated. After removing these from the overall average (leaving packet length, window size, and source port) the heatmap shown in Figure 6.8 is obtained. For comparison, the heatmap before removal of the features in shown in Figure B.1.

Several features of this visualization stand out and are worth discussing. First, it is apparent that similarity is very much concentrated along the diagonal, as expected. Within classes there appears to be a great deal of similarity.

Between similar classes, there is also similarity. There are three green blocks just off the diagonal (six, if the symmetrical mirrors are counted). These blocks show discrimination between versions of FreeBSD, Nmap, or Windows is possibly not reliable using the three included features. The other green block, far removed from the diagonal, shows a similarity between Debian Linux and Red Flag Linux. This is not surprising since, even with different implementations of Linux, the underlying Kernel is similar or identical (sans version disparities). The slight green shift in squares where an operating system is compared to an application is due to the overlap in these datasets.

Concerning similarity between dissimilar datasets, the picture also appears positive. FreeBSD, Windows, and Linux appear to be reliably discernible from each other. So, too, is the use of Hydra in attacking HTTP versus IMAP. There is some small level of similarity between FreeBSD 5 and Debian, Windows 7, and Windows XP. For a Kolmogorov-Smirnov comparison with $n_1 = n_2 = 100$ and $\alpha = 0.05$, the critical value is approximately 0.1923 (calculated from

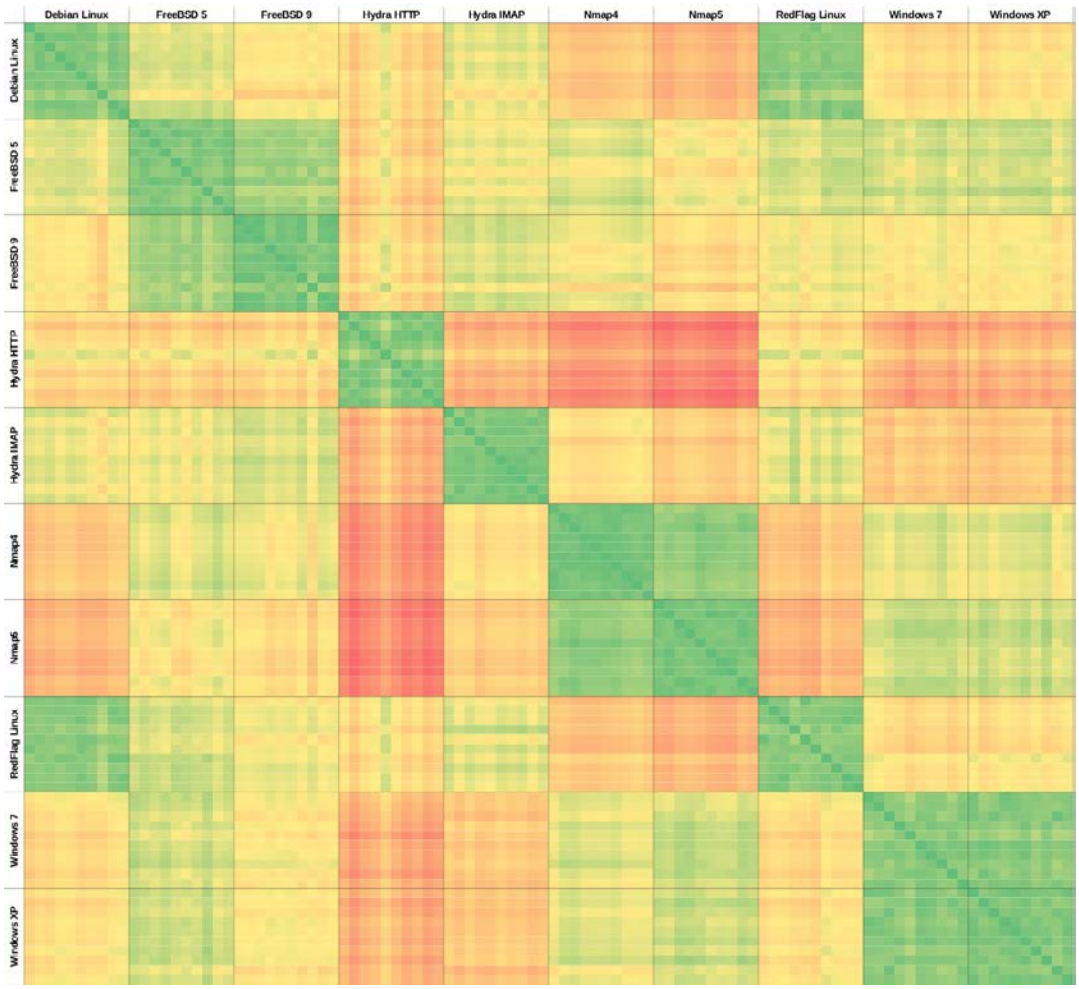

Figure 6.8 Kolmogorov-Smirnov Comparisons: Overall Average (Best Features)

the Kolmogorov-Smirnov critical value formula, $D_c = c(\alpha)\sqrt{\frac{n_1+n2}{n_1*n_2}}$, where $c(\alpha)$ is the coefficient given in a lookup table. No values off the exact diagonal met this threshold, meaning these between-disparate-classes similarities, while relatively anomalous, are not troubling in a statistical sense (even at $\alpha = 0.001$).

CHAPTER 7. CONCLUSIONS

7.1 Contributions

This dissertation has made several contributions to the cyber security body of knowledge. First, the case for cyber deterrence was established. We demonstrated how asymmetric the cyber battlefield is, and showed that there is a history of successful cyber attacks. We demonstrated that the only way to establish cyber superiority is to deter attacks from ever making attempts. A determined attacker, with enough time, will be able to find her way into even the most secure systems. Therefore, one must remove this determination by affecting the decision making calculus. This is a matter of either raising the perceived costs of attacking — by demonstrating retaliatory capability — or lower the benefits — by establishing redundancies and increasing the difficulty of the attack. However, due to the asymmetry of cyber warfare, attackers who believe they will not be caught are unlikely to be dissuaded by difficulty (in fact they might be enticed by it). By this reasoning, we have shown that a critical factor in countering cyber attacks is the ability to attribute them.

After establishing this fact, we surveyed the literature to examine existing attribution methods. We found these lacking, due to the following assumptions: 1) Omniscience, 2) Omnipresence, and 3) *A Priori* Positioning. Removing these assumptions from the attribution equation leaves only methods which can be performed with data directly accessible to the defender, pulled from systems directly under his control, and with detection systems in place at the time of attack. We showed that none of the existing attribution strategies hold up under these restraints. After comparing cyber attribution to traditional methods of criminal investigation, we showed that if one draws parallels between conventional crime and cyber crime, a more realistic attribution technique is profiling — an, as yet, unexplored method of

cyber attribution.

By examining the literature for literary analysis and software forensics, we found the method of Stylometry. This method has been used with success in attributing the authors of written documents using feature analysis. We proposed that this same methodology might be used in cyber attribution, and compared it to the methods used in investigating other types of crime such as arson or attacks with explosives.

Using what we learned from Stylometry, we first tried to make use of Principal Component Analysis to tease apart the variance discovered using a Multivariate Analysis of Variance. It was seen that the most discernment was detectable by attack tool, but the operating system also had a number of distinctive characteristics, as well an interaction between operating system and attack tool. These findings provide support to the idea that sophisticated attackers, who use a constant set of attack tools (as they would if they had undergone formal training) or custom tools used only by them, will leave a signature behind.

PCA showed to be an unusable method, at least with our selected set of features and generated set of data. The features chosen, despite having significant statistical diversity, were too closely correlated for PCA to separate into more than one usable axis. This single axis showed a nearly uniform distribution of points across the component space for both operating system and attack tool labeling. After rethinking the entire process, we decided to try an alternate method.

We then performed Kolmogorov-Smirnov analyses of the features in our dataset directly (i.e., before performing any analysis of variance or covariance). By eliminating variables with known insignificant contributions (from MANOVA and histogram analyses), we were able to generate a subset of features likely to be usable in this type of analysis. We generated heatmaps for the Kolmogorov-Smirnov comparisons of each of 10 randomly selected sets of 100 streams from each group of data (by application and operating system) and for each feature set. These heatmap visualizations were used to show that similar operating systems and applications might be difficult to differentiate, but that between-classes comparisons were relatively divergent from within-class comparisons. We demonstrated that there are several

IPv4 and TCP features that are particularly good predictors of the operating system and application used by the attacker. These were the packet length, window size, and source port. A secondary set of features that was useful, but didn't meet our stricter criteria for selection, included the minimum and maximum time to live, the don't fragment flags, the acknowledgement flags, the FIN flags, and the push flags.

These findings lend hope for a *Cyberprint* — a new method of network forensic analysis. This method makes none of the assumptions of conventional attribution methods, relies upon simple computations, and narrows the field of analyzed features. Just as law enforcement and military personnel use feature-based techniques to identify biological, behavioral, and methodological aspects of an attacker, so might *Cyberprints* illuminate the virtual face.

7.2 Future Work

The first extension of this work would be to refine the Kolmogorov-Smirnov comparison process to the point where it can deliver a single, statistically-bounded answer to whether two streams come from the same source. Being able to say these streams have a Kolmogorov-Smirnov D-value of 5 is not useful in itself for decisions about attribution. Only in relation to the D-values for comparisons with other traffic does this carry weight. Finding a way to develop an overall score for the similarity of two streams would make this technique "field-ready".

Second, more work needs to be completed to determine other features that might be useful for *Cyberprints*. This analysis focused upon IPv4 and TCP features. Other protocols at the network layer, as well as payload-specific features, should also be considered for use. "Layer 8" features, such as geolocation, could also prove useful. Finding ways to incorporate socio-political metadata related to a stream, perhaps pulled from real-time feeds, would allow other considerations, such as current world events, known attackers, and enemies of the defender. Really, the potential for features is as boundless as cyberspace. If a feature can be correlated with a traffic stream and quantified for analysis, it could potentially be used for a *Cyberprint*. Finding features that, in general, have discriminatory power will enhance the capabilities of this technique.

Similarly, the test data used in this dissertation, while representing a variety of operating systems and several applications, is far from exhaustive. Testing this technique on more complex datasets might reveal ways to make it more reliable. Only be exposing it to new scenarios can it be refined to handle them. Thus, this work should be viewed as the genesis of *Cyberprints*, which have much evolving to do.

One cannot think about any method of countering attackers without thinking of the manner in which attackers will respond. For this reason analysis should be performed to determine what methods of countering *Cyberprints* might be developed. Already, it is known that a person wishing to hide her operating system can use a border device to scramble features, such as the acknowledgment number, or proxy particular flags. If word of a signature-based attribution method reach cyber criminals, they will no doubt find ways to make their signatures more difficult to detect.

APPENDIX A. FEATURE DISTRIBUTION HISTOGRAMS

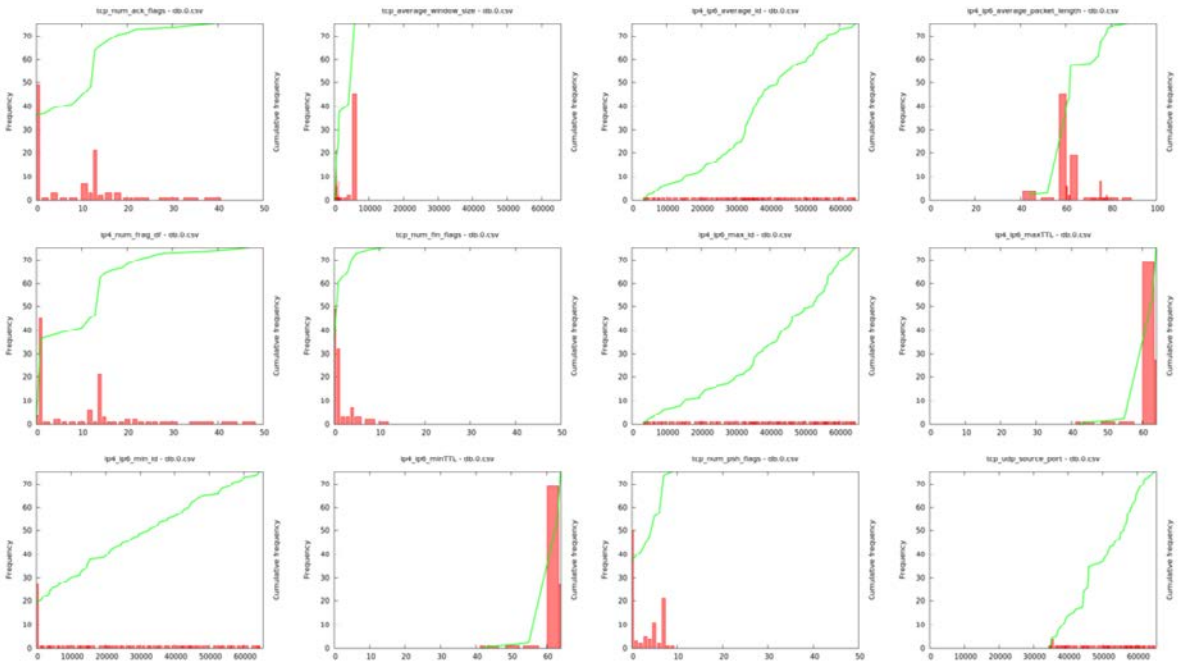

Figure A.1 Feature Distribution Histogram: Debian Linux

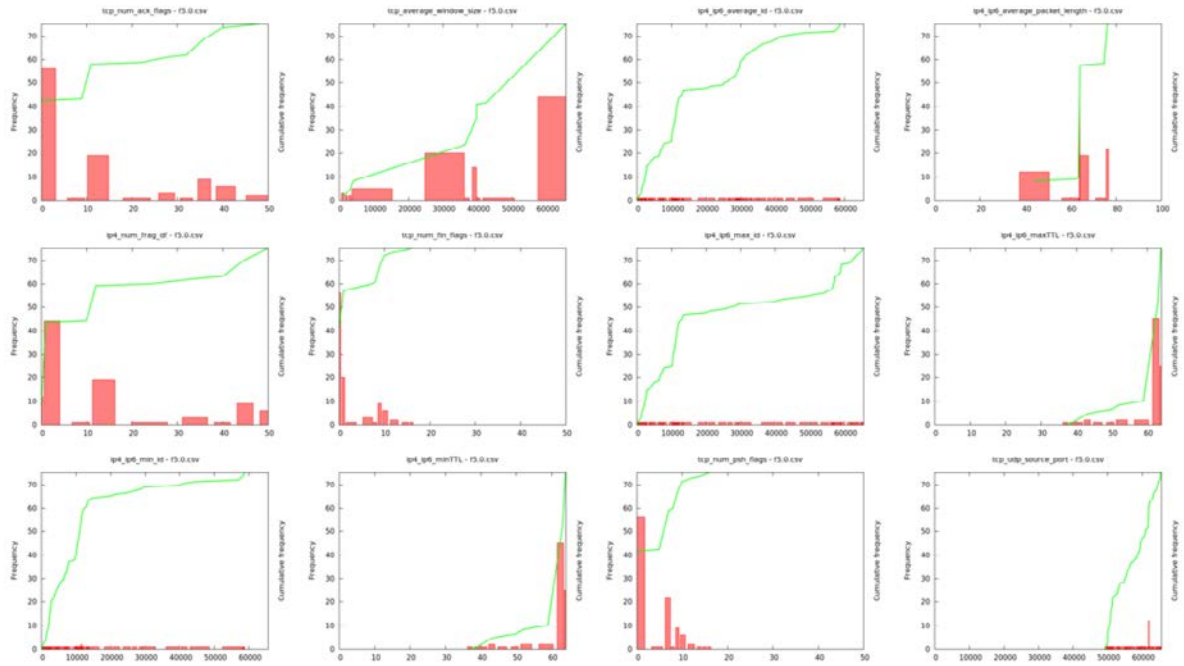

Figure A.2 Feature Distribution Histogram: FreeBSD 5

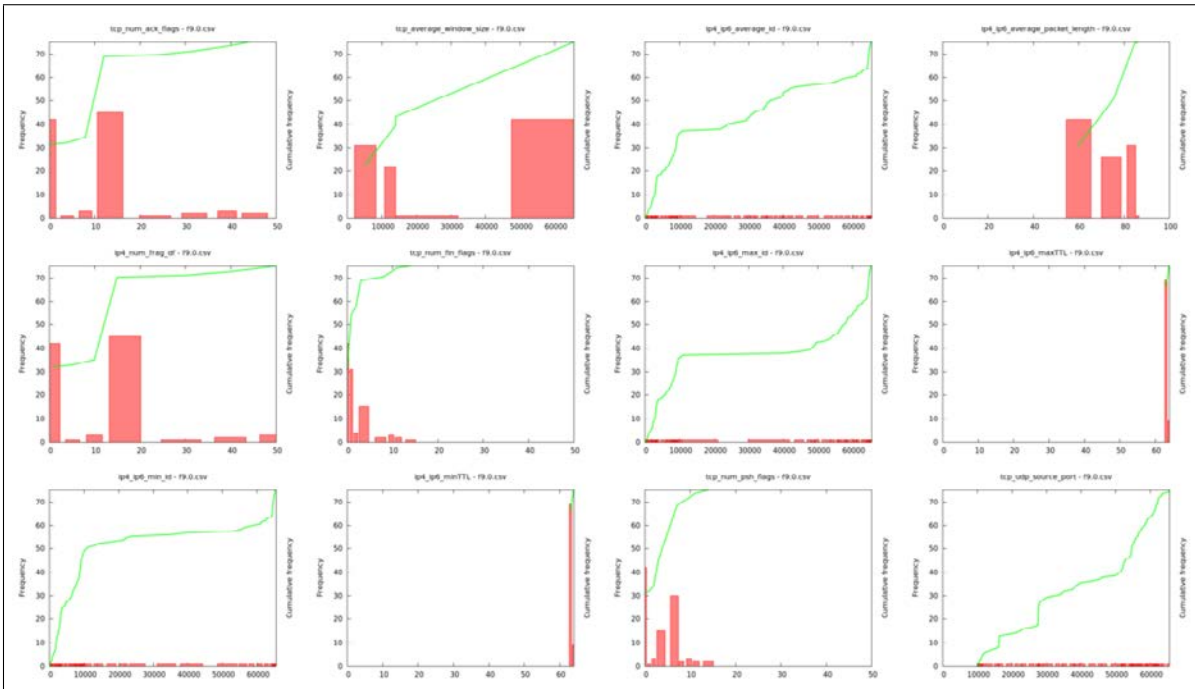

Figure A.3 Feature Distribution Histogram: FreeBSD 9

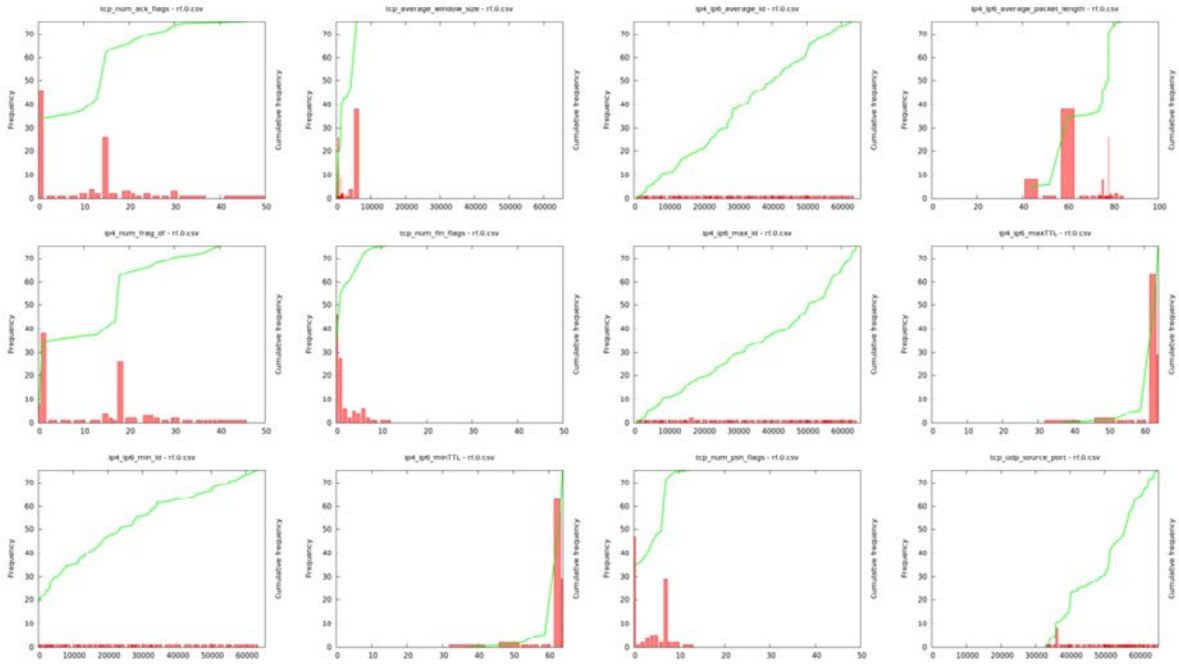

Figure A.4 Feature Distribution Histogram: Red Flag Linux

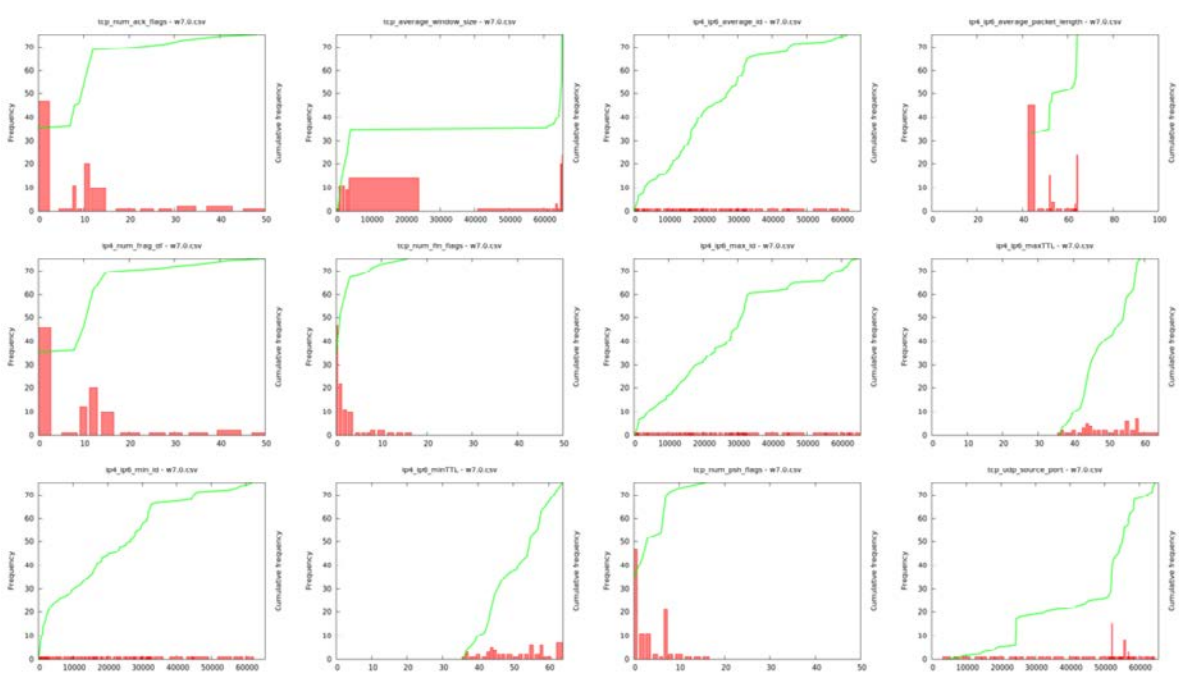

Figure A.5 Feature Distribution Histogram: Windows 7

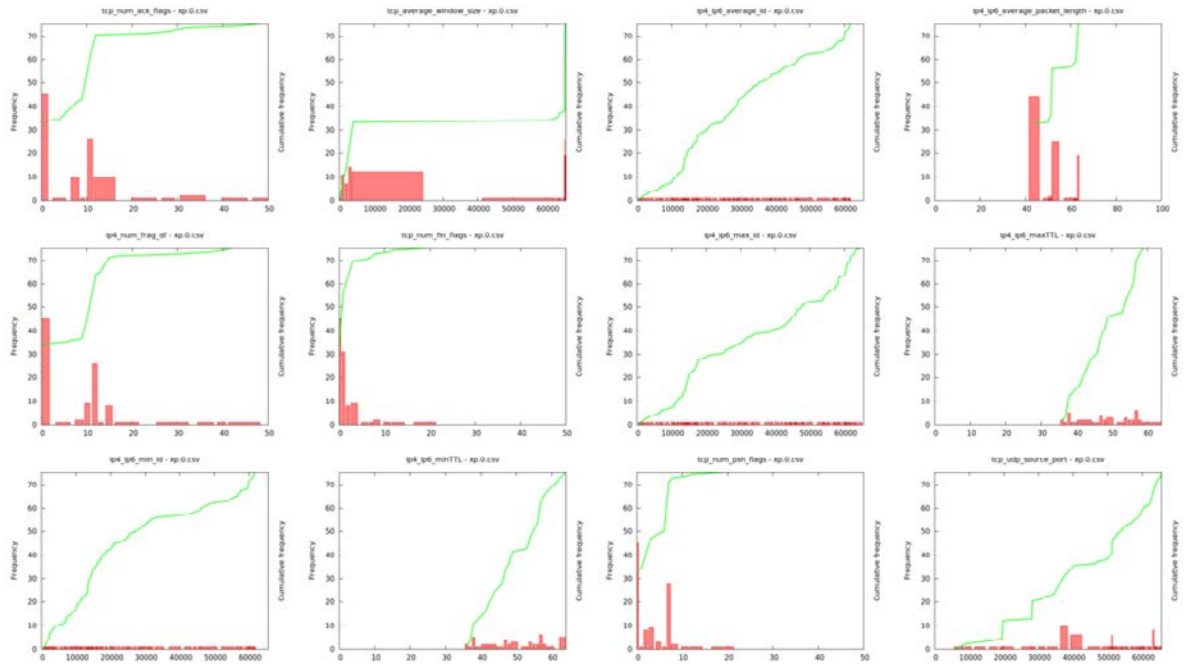

Figure A.6 Feature Distribution Histogram: Windows XP

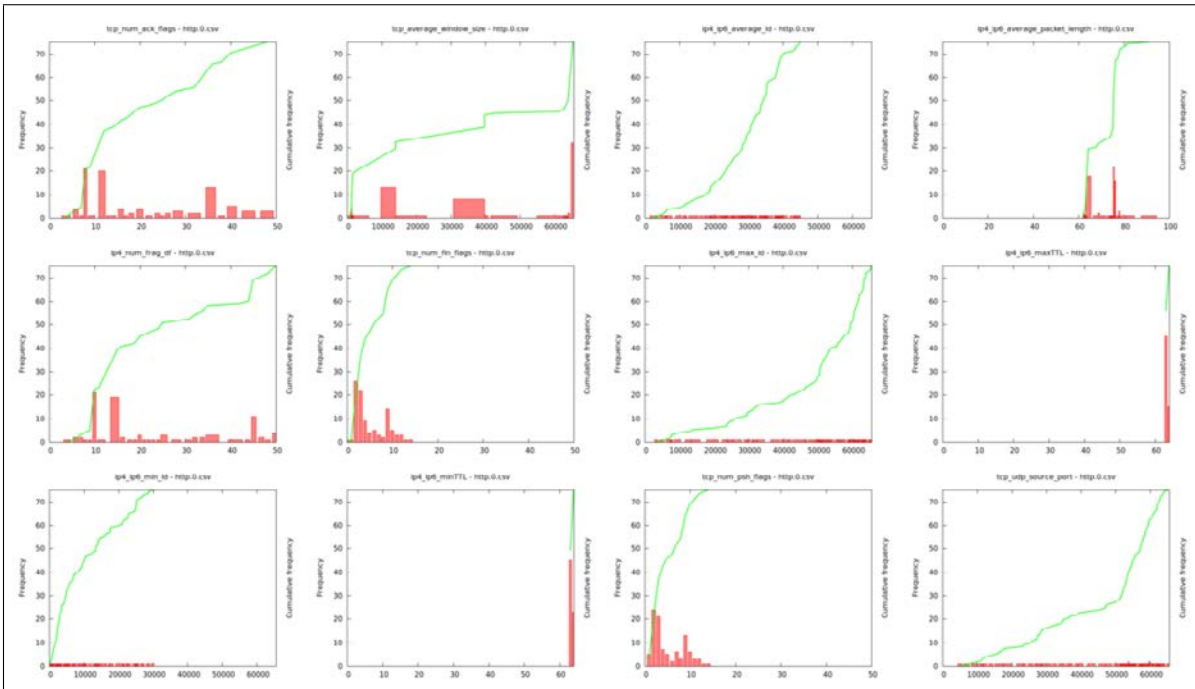

Figure A.7 Feature Distribution Histogram: Hydra HTTP Brute-Force

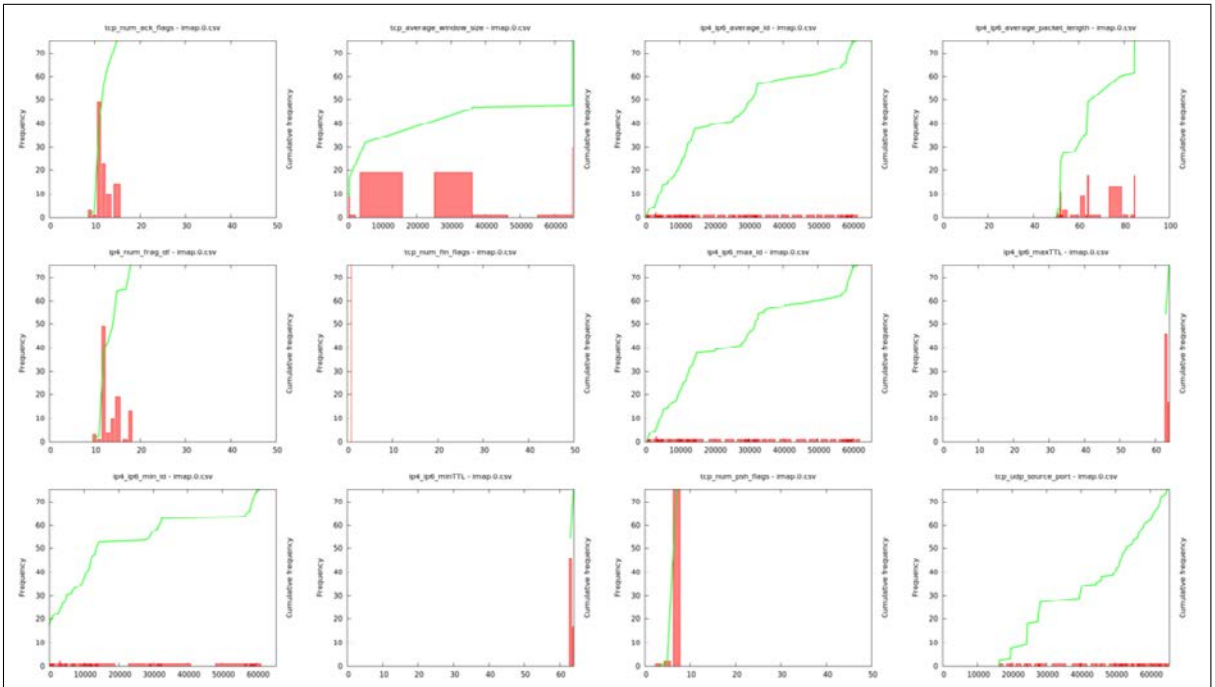

Figure A.8 Feature Distribution Histogram: Hydra IMAP Brute-Force

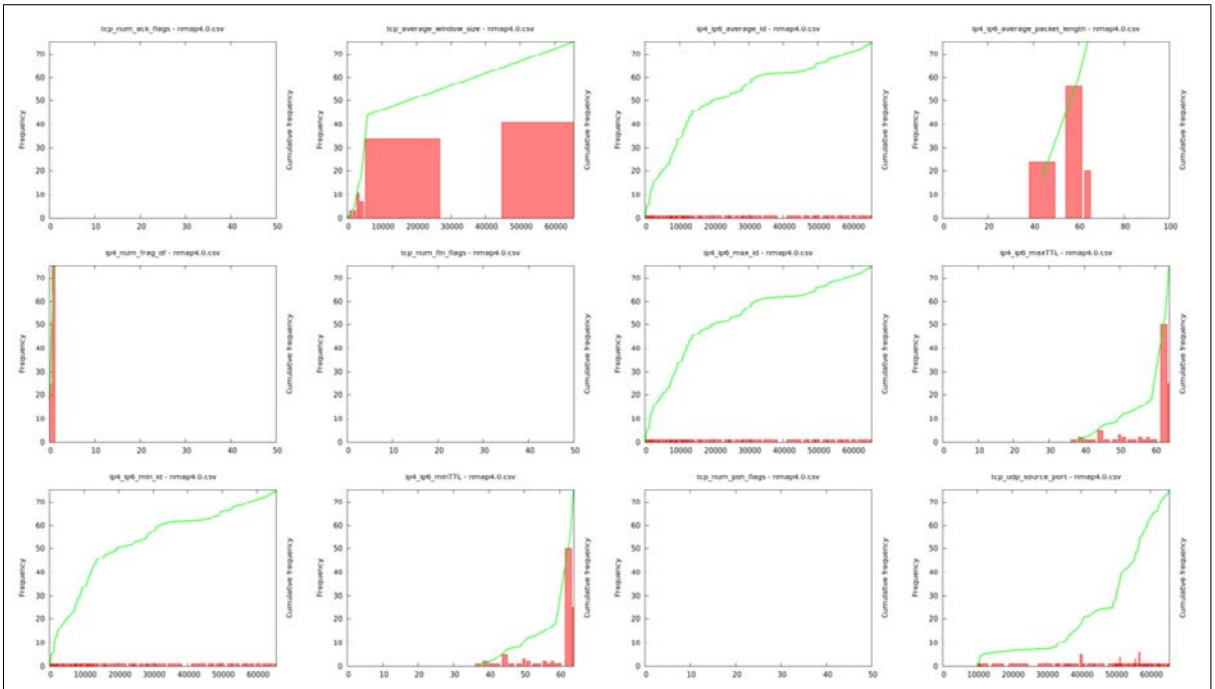

Figure A.9 Feature Distribution Histogram: Nmap 4 Scan

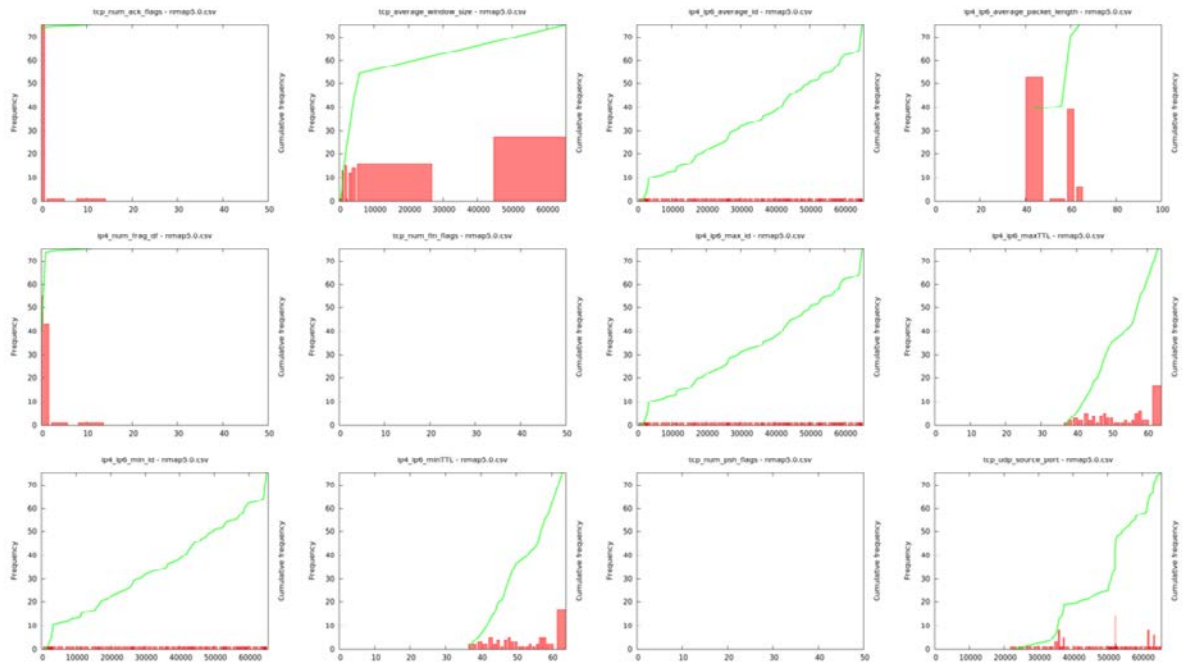

Figure A.10 Feature Distribution Histogram: Nmap 5 Scan

APPENDIX B. KOLMOGOROV-SMIRNOV HEATMAPS

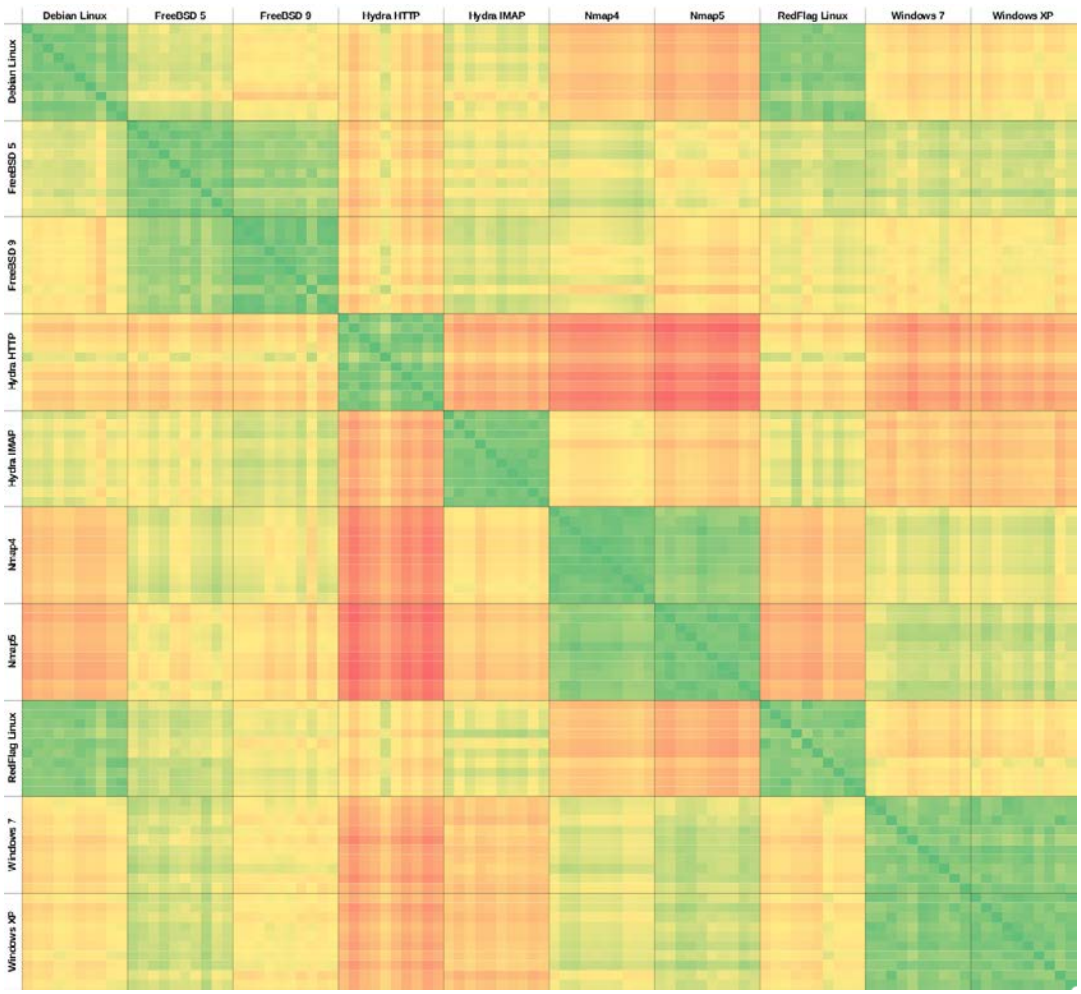

Figure B.1 Kolmogorov-Smirnov Comparisons: Overall Average, before filtering

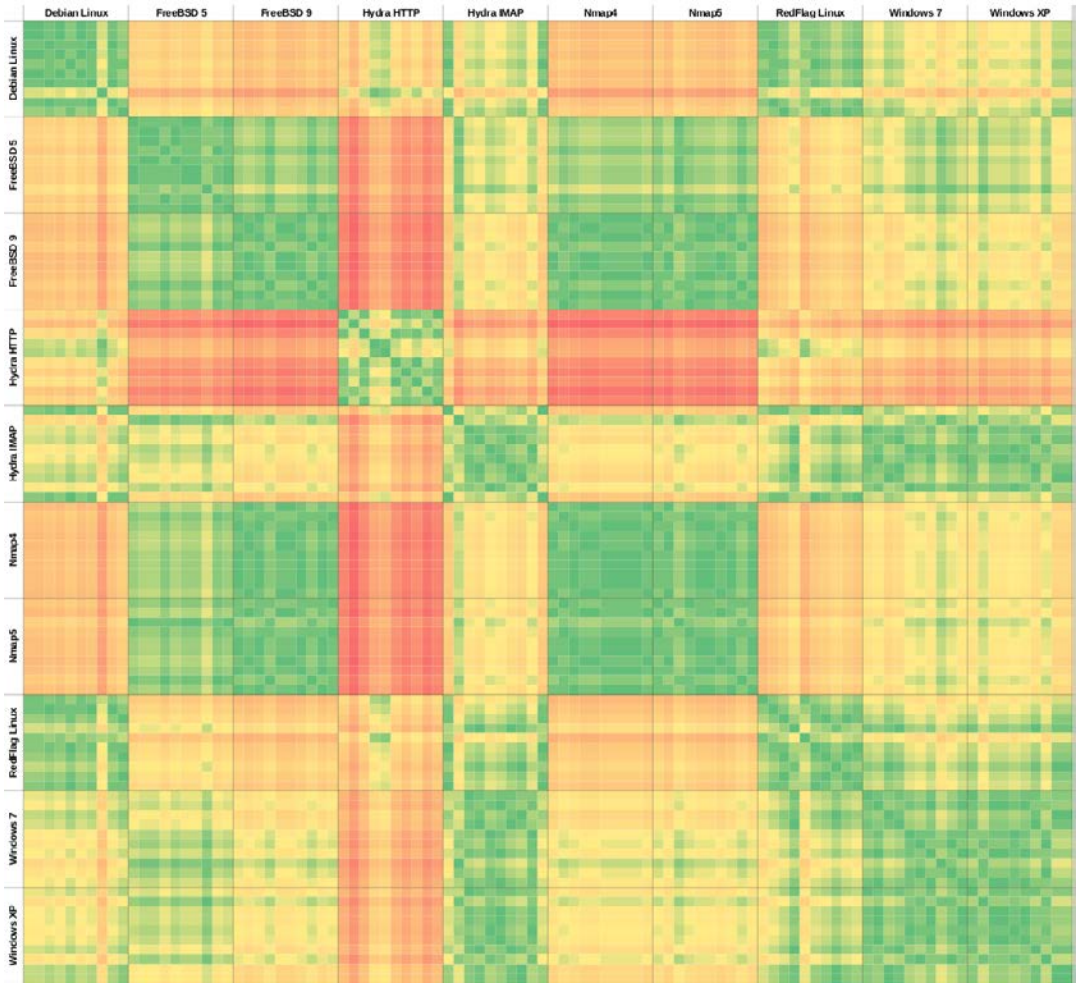

Figure B.2 Kolmogorov-Smirnov Heatmap: Packet Length

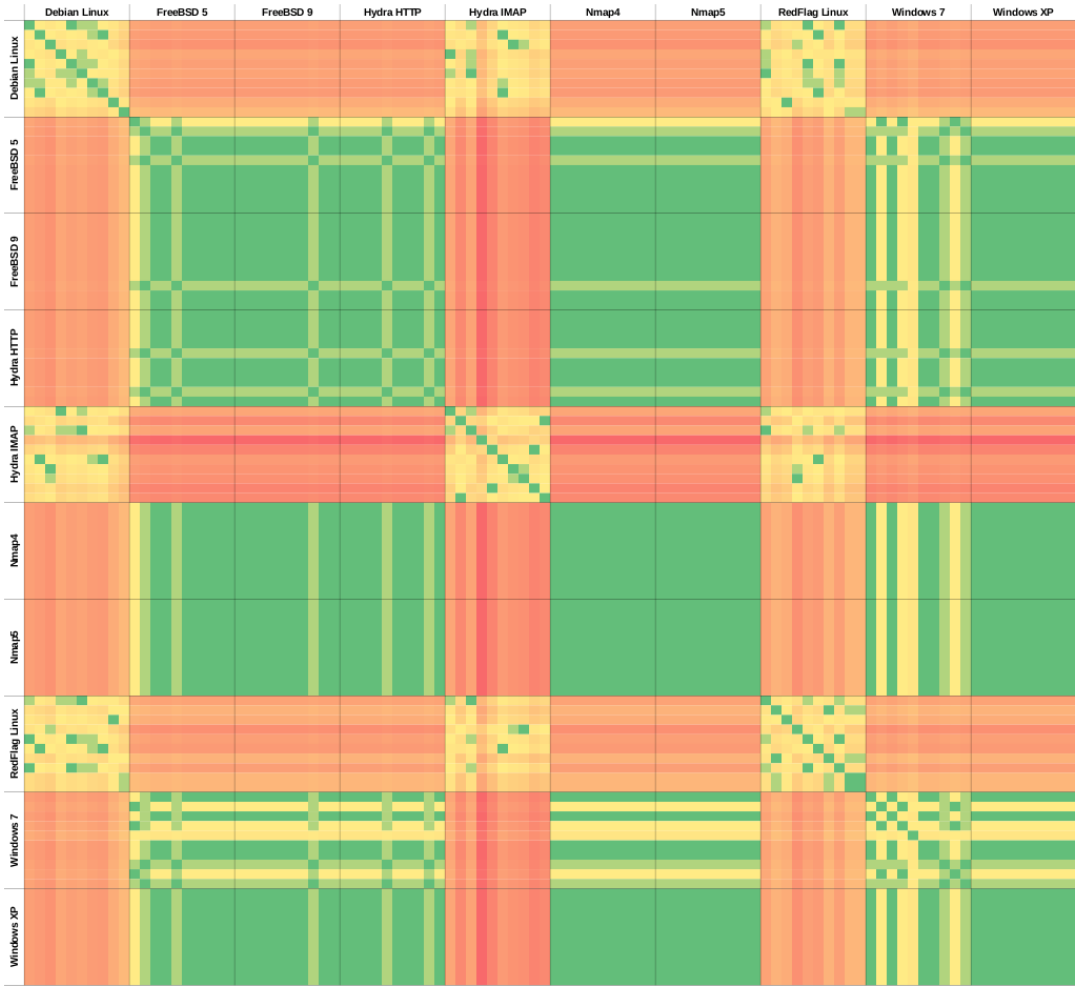

Figure B.3 Kolmogorov-Smirnov Heatmap: Minimum IPv4 Identifier

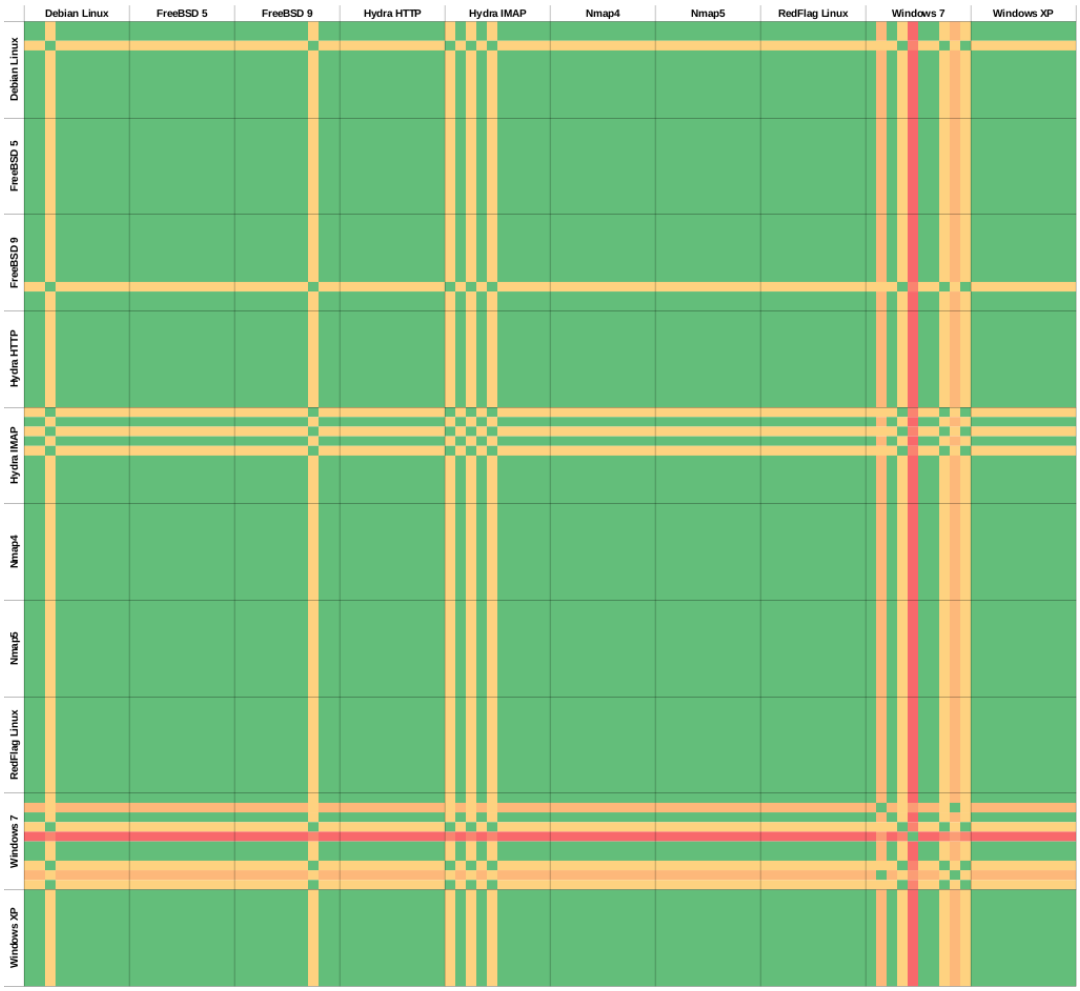

Figure B.4 Kolmogorov-Smirnov Heatmap: Average IPv4 Identifier

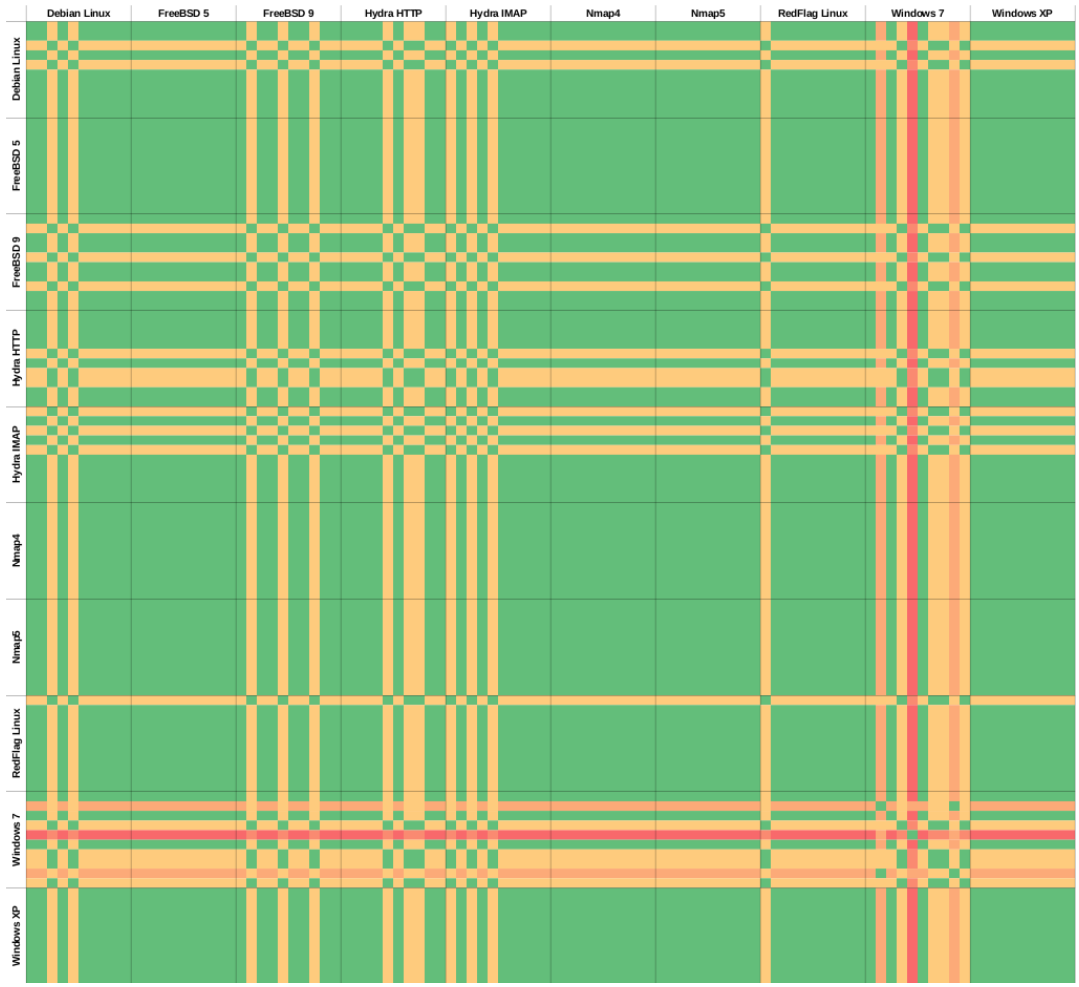

Figure B.5 Kolmogorov-Smirnov Heatmap: Maximum IPv4 Identifier

Figure B.6 Kolmogorov-Smirnov Heatmap: Minimum IPv4 Time-to-Live

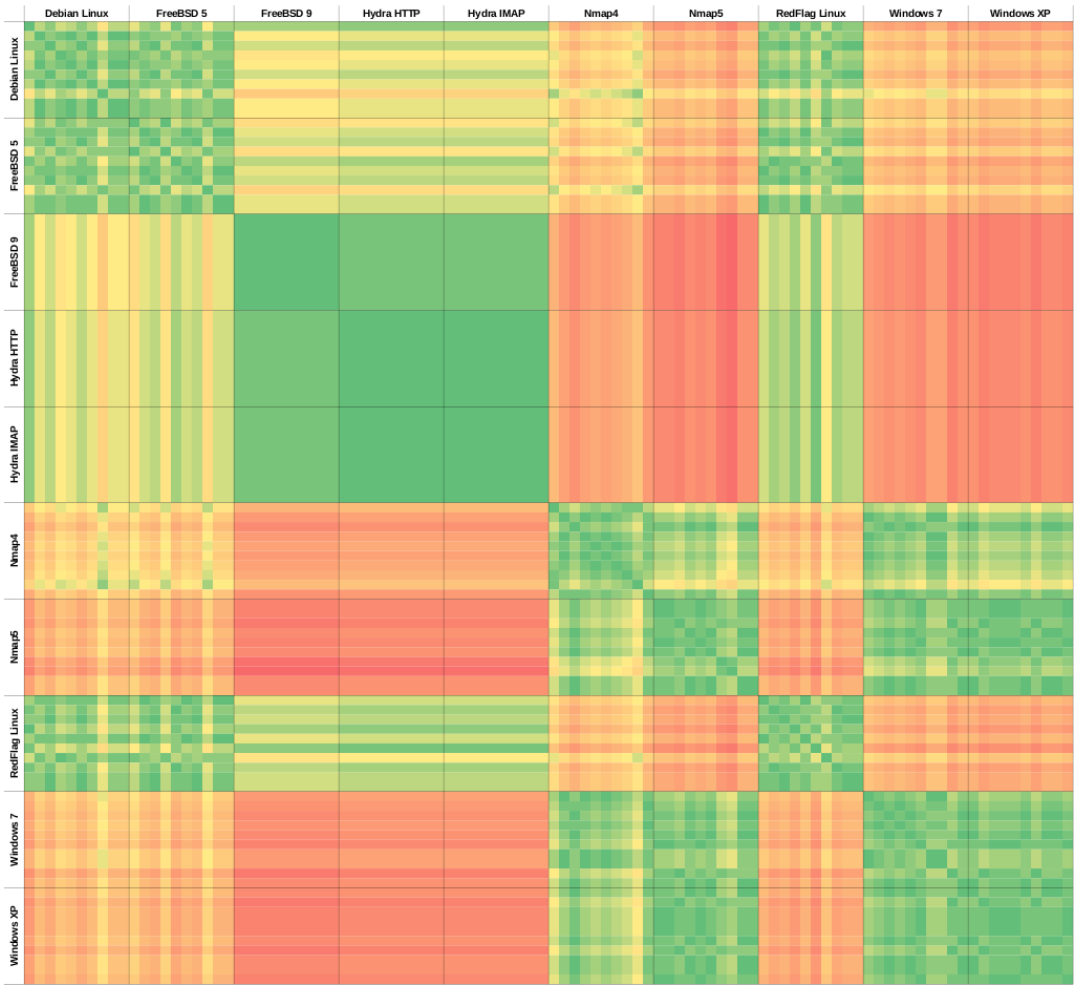

Figure B.7 Kolmogorov-Smirnov Heatmap: Maximum IPv4 Time-to-Live

Figure B.8 Kolmogorov-Smirnov Heatmap: IPv4 Don't Fragment Flags

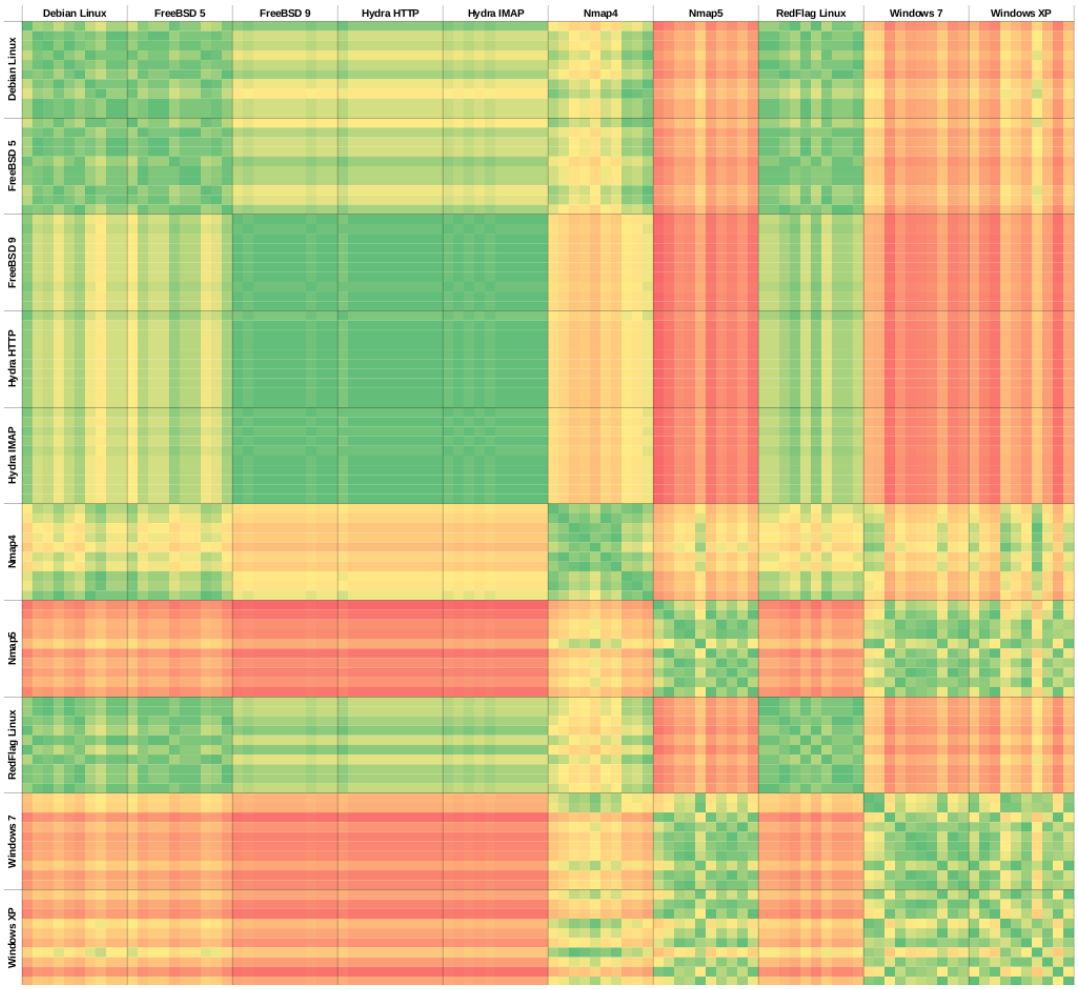

Figure B.9 Kolmogorov-Smirnov Heatmap: TCP Source Port

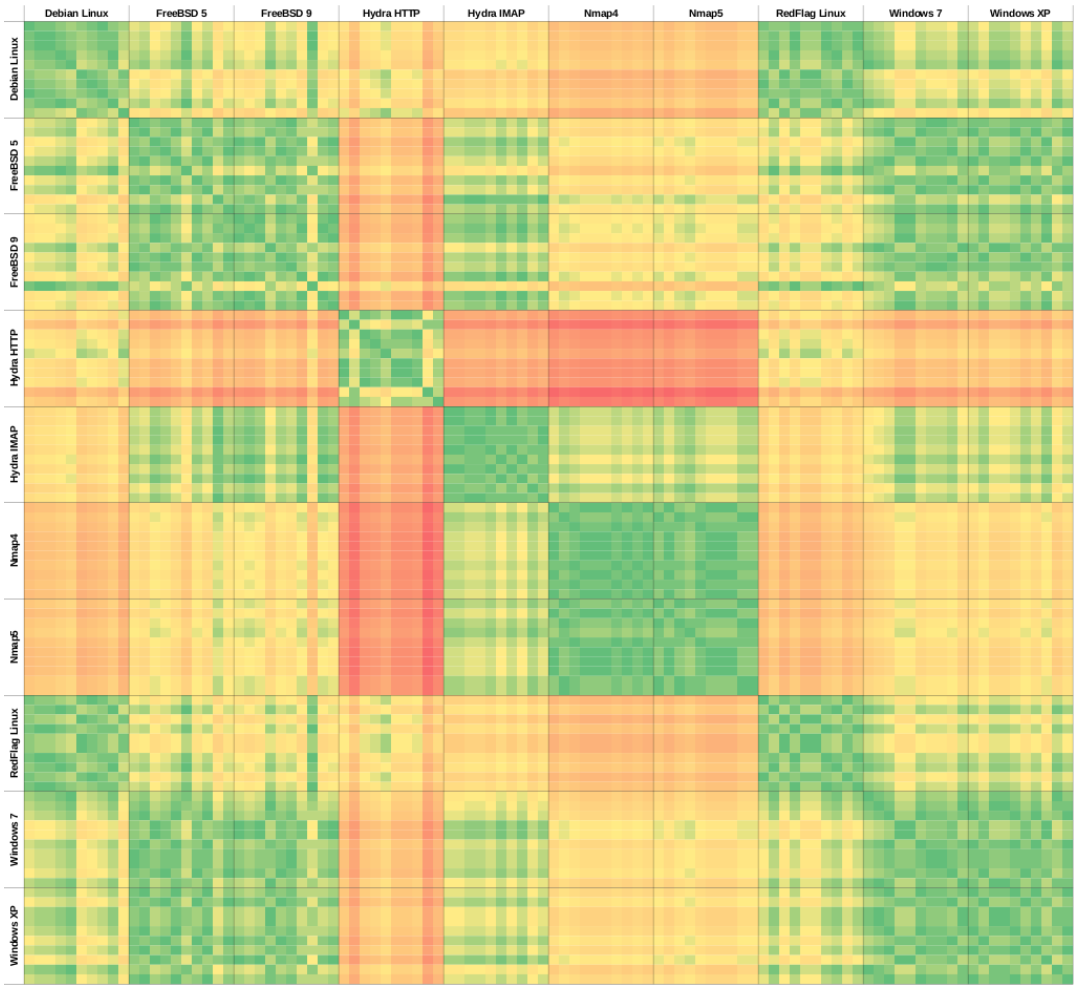

Figure B.10 Kolmogorov-Smirnov Heatmap: TCP Acknowledgment Flags

Figure B.11 Kolmogorov-Smirnov Heatmap: TCP FIN Flags

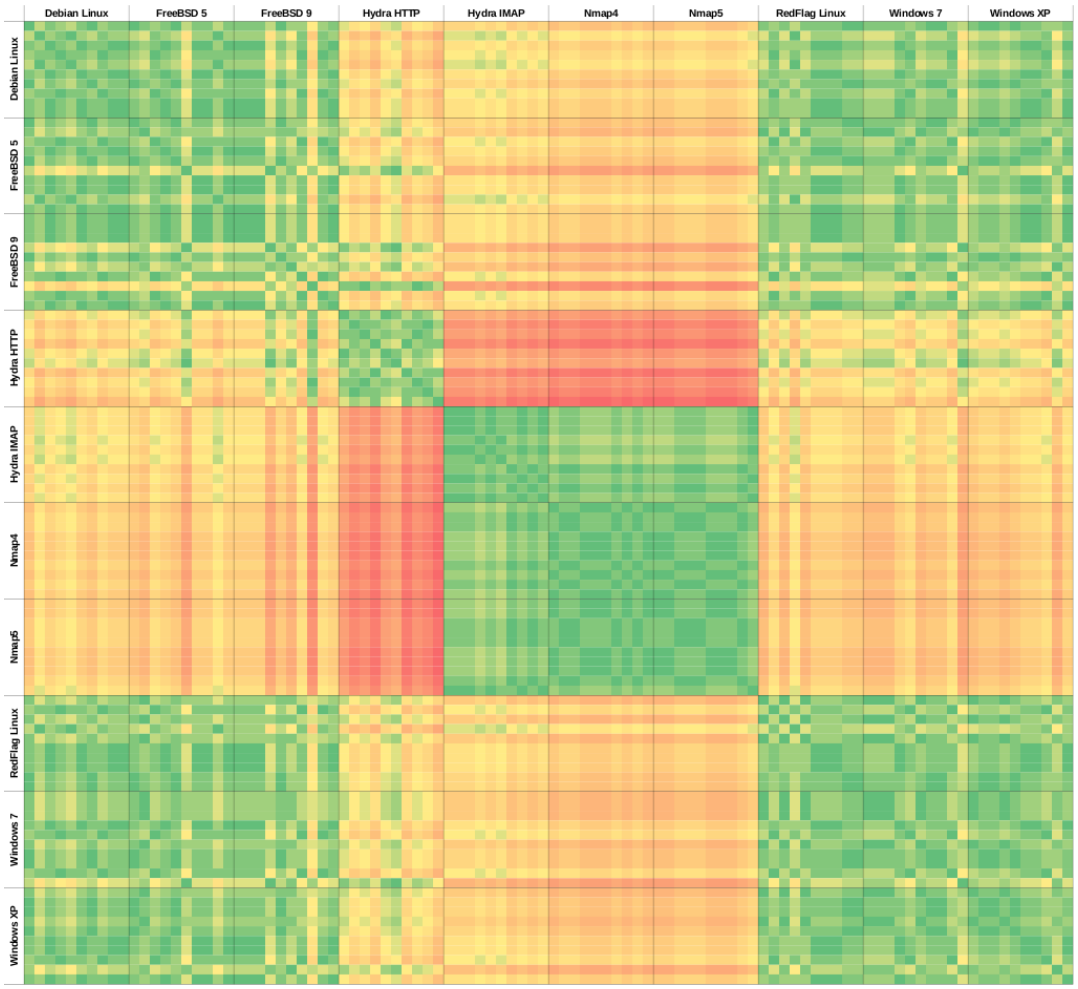

Figure B.12 Kolmogorov-Smirnov Heatmap: TCP Push Flags

Figure B.13 Kolmogorov-Smirnov Heatmap: TCP Window Size

GLOSSARY

Advanced Persistent Threat An adversary that possesses sophisticated levels of expertise and significant resources, which allow creation of opportunities to achieve its objectives by using multiple attack vectors (e.g., cyber, physical, and deception). These objectives typically include establishing and extending footholds within the information technology infrastructure of targeted organizations for purposes of exfiltrating information, undermining or impeding critical aspects of a mission, program, or organization; or positioning itself to carry out these objectives in the future. The advanced persistent threat: (i) pursues its objectives repeatedly over an extended period of time; (ii) adapts to defenders efforts to resist it; and (iii) is determined to maintain the level of interaction needed to execute its objectives. (NIST SP 800-39, 2010)

Anonymity The state of not being associated with an identifiable party.

Anonymization The act of hiding one's identity by some means of obfuscation.

APT See *Advanced Persistent Threat*.

Attribution The process of determining the source of an event, which might require overcoming attempts at *anonymization*.

Bit Binary value 0 or 1. See also *Byte*.

Blacklist A list of entities denied to access a resource. This implies all entities not on the list are granted access. See also *Whitelist*.

Botnet A distributed network of software implants. Often remotely controlled and installed without the knowledge or consent of the host system's owner. May be used in a variety of attacks.

Byte A series of eight bits. Usually measured using Greek/Latin prefixes for orders of magnitude (e.g., MB for Megabytes, or one million bytes). See also *Bit*.

Casus Belli Justification for retaliation or other initiation of war. Latin for *case (casus) for war (bellum)*.

Chief Information Security Officer The executive in an organization tasked with *Information Assurance*-related objectives. This position is sometimes combined with the Chief Security Officer, who might also be responsible for additional security concerns.

CISO See *Chief Information Security Officer*.

Classification The separation of data into tiers, based upon the damage that might result if it were to be exposed to unauthorized parties. Used extensively in military applications, in particular.

Cloud An abstraction from the traditional model of static and single-purpose local physical servers which makes data, services, and even entire infrastructures appear as a service to an end-user. Often used to increase robustness of provided services, while decreasing requirements for power, space, and cooling of *data centers*.

Critical Infrastructure Systems required for the safety and productivity of a population. Examples include power transmission networks, emergency response services, transportation networks, and communication networks.

Cyberprint A new technique used to identify the probable source of a network transmission using feature analysis.

Cyberspace The abstract idea of a domain of interaction in addition to those of land, sea, air, and space. Consists of the digital exchange of information over network *links* between

disparate devices.

Data Center A special-purpose facility, which houses a large number of computer and network systems. Often hardened against attack, natural disaster, and other disturbances.

DDOS See *Distributed Denial of Service Attack*.

Deterrence The use of signaling through threats or demonstrations of force to affect the decision-making process of a (potential) adversary.

Distributed Denial of Service Attack An adversarial action that uses large amounts of traffic destined for a single host or network, generated by a large number of attacking systems, to degrade or disable data transmission by the target. This effectively takes the target offline. See also *Botnet*.

Encryption The obfuscation of the the contents of a communication to protect these contents from being observed by unauthorized parties.

Espionage The act of covertly gathering information about another party, usually without authorization and perhaps illegally.

Executable A software program on a computer system. A piece of data which can be launched (executed) to operate its own set of instructions.

Exploit A technique which can be used to take advantage of a *vulnerability* in a system.

Forensics (Cyber) The analysis of a computer system or recorded network transmission to determine information about the identity or intent of an unknown party.

Header The portion of a network *packet* that contains basic information necessary for it to reach its destination, possibly with assurances of transmission properties, such as priority.

Honeypot, Honeynet A system or network constructed with the purpose of enticing attackers to divert their attention from production systems or to learn more about their methods.

ICANN See *Internet Corporation for Assigned Names and Numbers.*

IDS See *Intrusion Detection/Prevention System.*

Information Assurance The profession of protecting confidentiality, integrity, and availability of information required for a business, government, or other entity to function in the desired state.

Internet Corporation for Assigned Names and Numbers The international agency responsible for delegating control of identifying tokens on the Internet. Specifically, *domains* and *IP Addresses.*

Internet Protocol The protocol which exists at layer 3 of the OSI model (the network layer) responsible for routing traffic between two nodes on a network or internetwork, using specially crafted and assigned numeric addresses.

Internet Protocol Address The specially crafted numeric identifier used by the *Internet Protocol* to determine a node's identity on a network or internetwork.

Internet Service Provider An agency that maintains a substantial amount of throughput and *Internet Protocol* address space, and allocates these resources to customers to allow them to communicate with non-local nodes.

Intrusion Detection/Prevention System A system configured to watch for unauthorized or malicious activity on a network connection or host, and (in the case of an IPS) take steps to stop the attack and remediate its effects.

IP, IPv4, IPv6 See *Internet Protocol.*

IP Address See *Internet Protocol Address.*

IPS See *Intrusion Detection/Prevention System.*

ISP See *Internet Service Provider.*

Jitter Variation in *latency.* A second-order *link* statistic.

Jurisdiction The scope of authority to which an entity is legally entitled.

Latency The delay incurred during *packet* transmission, due to *link* characteristics or processing delays.

Link A channel (usually physical) between two devices that allows them to exchange information. When referring to a virtual channel (i.e., comprised of multiple physical links), usually the term connection or session is used instead.

Malware Software which performs a function against the best interests of the owner of a system. Often installed without the consent or knowledge of the owner, and often used as part of either information theft or distributed denial of service attacks.

Packet A single unit of information, the smallest quanta that can be sent or received by a network device. Consists of a *header* and *payload*.

Payload The actual data transmitted from source to destination without any peripheral data, such as the *header*.

P2P See *Peer-to-Peer Network*.

Peer-to-Peer Network A network which has little or no hierarchy, in which all nodes communicate directly with each other. Some might have "super" nodes responsible for helping locate other normal nodes, but all information is distributed among many nodes.

Proxy (Network) A system responsible for obtaining information from another server on behalf of a client, or that acts as a go-between for a client and server, due to access controls that would otherwise prevent communication between these two parties.

Risk The possibility of an unintended or undesired event occurring.

Root-kit A special type of *Malware* that tightly couples itself with the operating system, making it difficult to remove and potentially very dangerous to the safety of information on the compromised system.

Router A network device which serves as an intermediary node to relay *packets* between other devices on the *Internet*, even when they are far dispersed.

Secure Socket Layer/Transport Layer Security (SSL/TLS) The protocols which provide an *encrypted* connection between a client and server, optionally with verification of the identification of both parties, over which other higher-level network protocols can communicate. The most common usage is in HTTP/S, used to secure many financial and otherwise sensitive web sites.

Spam Unsolicited electronic messages. Usually marketing material, often offensive or illegible in nature.

SSL See *Secure Socket Layer*.

Steganography The hiding of a secondary, often *encrypted*, payload within an overt payload (or 'cover' payload) to prevent knowledge of the existence of the secondary payload by unauthorized parties.

Stepping Stone An intermediate node between a source and destination. Often used to disguise the true origin of a transmission.

Stylometry A literary technique using statistical methods to analyze various features (metrics) of a written document to ascertain shared authorship.

TCP See *Transmission Control Protocol*.

TLS See *Secure Socket Layer/Transport Layer Security*.

Topology The set of network devices and links which comprise a network. Important in making decisions regarding transmission of *packets* through a network.

Traffic The set of all *packets* transmitted on a *link* or set of links comprising a network.

Transmission Control Protocol The connection-oriented communication mechanism used on top of the *Internet Protocol* to establish a reliable communication channel, which will try to ensure optimal transmission.

Threat A specific realization of a *risk*, which might use an *exploit* to take advantage of a *vulnerability* in a system.

UDP See *User Datagram Protocol*.

Unclassified See *Classification (Data)*.

Universal Serial Bus An interface for connecting peripherial devices to a computer system. Of particular relevance to security, such devices can be used to plant *Malware* onto a host system.

USB See *Universal Serial Bus*.

User Datagram Protocol The datagram-oriented communication mechanism used on top of the *Internet Protocol* to send data from point-to-point without first establishing a connection. No guarantees are made for delivery.

Vulnerability A flaw in a system which creates a *risk*.

Whitelist A list of entities allowed to access a resource. This implies all entities not on the list are denied access. See also *Blacklist*.

Zero-day exploit An attack that makes use of a previously unknown *vulnerability* in a system.

BIBLIOGRAPHY

Abbasi, A. and Chen, H. (2008). Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems*, 26(2).

Achen, C. and Snidal, D. (1989). Rational Deterrence Theory and Comparative Case Studies. *World Politics*, 41(2):143–169.

Achtert, E., Kriegel, H.-P., and Zimek, A. (2008). Elki: A software system for evaluation of subspace clustering algorithms. In *Proceedings of the 20th international conference on Scientific and Statistical Database Management*, SSDBM '08, pages 580–585, Berlin, Heidelberg. Springer-Verlag.

Adams, J. (2001). Virtual Defense. *Foreign Affairs*, 80(3):98–112.

Ankerst, M., Breunig, M., and Kriegel, H.-P. (1999). OPTICS: Ordering Points to Identify the Clustering Structure. *ACM SIGMOD international conference on Management of data*, pages 49–60.

Baayen, R., van Halteren, H., and Tweedie, F. (1996). Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing*, 11(3):121–131.

Beeker, M. K. (2009). Strategic Deterrence in Cyberspace: Practical Application. Master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio - Graduate School of Engineering and Management.

Beidleman, L. C. S. (2009). Defining and Deterring Cyber War. Technical report, Army War College, Carlisle Barracks, Pennsylvania.

Blotzer, L. (2000). Deterrence in the 21st Century. *The Collins Center Update*, pages 2–5.

Caruana, R. and Freitag, D. (1994). Greedy Attribute Selection. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 28–36. Morgan Kaufmann.

Center, N. W. C. C. (2010). 2010 Internet Crime Report. Technical report, Internet Crime Complaint Center.

Chakraborty, R., Narasimhan, S., and Bhunia, S. (2009). Hardware Trojan: Threats and emerging solutions. *High Level Design Validation and Test Workshop, 2009. HLDVT 2009. IEEE International*, 4-6:166–171.

Chilton, K. P. (2008). *Statement of General Kevin P. Chilton before the House Armed Services Subcommittee on United States Strategic Command.* United States Strategic Command, Washington, D.C.

Corney, M. W., Anderson, A. M., Mohay, G. M., and de Vel, O. (2001). Identifying the Authors of Suspect Email. Technical report, Queensland University of Technology and Defence Science and Technology Organisation (Australia).

Daniels, T. (2002). *Reference Models for the Concealment and Observation of Origin Identity in Store-and-Forward Networks.* PhD dissertation, Purdue University, West Lafayette, Indiana.

Daniels, T., Mina, M., and Russell, S. (2005). A signal fingerprinting paradigm for general physical layer and sensor network security and assurance. In *First International Conference on Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005.*

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum Likelihood from Incomplete Data

via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

DOJOC (2006). *Deterrence Operations Joint Operating Concept. Version 2.0.* United States Strategic Command, Offut Air Force Base, Nebraska.

DSA-1571-1 (2008). *DSA-1571-1 openssl – predictable random number generator.* Debian.

Ertoz, L., Steinbach, M., and Kumar, V. (2002). A New Shared Nearest Neighbor Clustering Algorithm and its Applications. In *Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining.*

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231.

Evans, N. (2009). *Information technology social engineering: an academic definition and study of social engineering - analyzing the human firewall.* PhD dissertation, Iowa State University.

George, A. and Smoke, R. (1974). *Deterrence in American Foreign Policy.* Columbia University Press.

Gerdes, R., Daniels, T., Mina, M., and Russell, S. (2006). Device Identification via Analog Signal Fingerprinting. In *Proceedings of the Network and Distributed System Security Symposium Conference.*

Habiger, E. (2010). Cyberwarfare and Cyberterrorism: The Need for a New U.S. Strategic Approach. Technical report, Cyber Security Institute.

Hartigan, J. and Wong, M. (1976). Algorithm AS 136: A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108.

Hayes, J. H. (2008). Authorship Attribution: A Principal Component and Linear Discriminant Analysis of the Consistent Programmer Hypothesis. Technical report, University of Kentucky.

Hijacking (2010). China denies hijacking a huge chunk of U.S. net traffic. *BBC*.

Holmes, D. I. (1994). Authorship Attribution. *Computers and the Humanities*, 2(2):87–106.

Hunker, J., Hutchinson, B., and Margulies, J. (2008). Roles and Challenges for Sufficient Cyber-Attack Attribution. Technical report, Institute for Information Infrastructure Protection.

Jackson, E. (2006). Detecting intrusions at layer one: device fingerprinting for network access authorization. Master's thesis, Iowa State University.

Jervis, R. (1984). *Strategy and Nuclear Deterrence*, chapter Deterrence and Perception, pages 57–84. Princeton University Press.

Jervis, R., Lebow, R., and Stein, J. (1989). *Psychology and Deterrence*. Johns Hopkins Univ Press, Baltimore.

Jin, Y., Kupp, N., and Makris, Y. (2009). Experiences in hardware trojan design and implementation. *International Workshop on Hardware-Oriented Security and Trust, 2009. HOST '09. IEEE*, 27-27:50–57.

Joliffe, I. (2002). *Principal Component Analysis*. Springer, Secaucus, New Jersey.

Juola, P. and Baayen, R. H. (2005). A Controlled-corpus Experiment in Authorship Identification by Cross-entropy. *Literary and Linguistic Computing*, 20:59–67.

Kailing, K., Kriegel, H.-P., and Krger, P. (2004). Density-connected subspace clustering for high-dimensional data. In *Proc. SIAM Int. Conf. on Data Mining (SDM'04)*, pages 246–257.

Kira, K. and Rendell, L. A. (1992). A practical approach to feature selection. In *Proceedings of the ninth international workshop on Machine learning*, pages 249–256.

Koppel, M. and Schler, J. (2003). Exploiting Stylistic Idiosyncrasies for Authorship Attribution. In *Proceedings of the International Joint Conferences on Artificial Intelligence 2003 workshop on computational approaches to style analysis and synthesis*, pages 69–72.

Kuhn, M. G. (2003). Compromising Emanations: Eavesdropping Risks of Computer Displays. Technical report, University of Cambridge Computer Laboratory.

Li, J., Zheng, R., and Chen, H. (2006). From Fingerprint to Writeprint. *Communications of the ACM*, 49:76–82.

Libicki, M. (2009). *Cyberdeterrence and Cyberwar*. The RAND Corporation, Santa Monica, California.

Mathewson, N. and Dingledine, R. (2005). Practical traffic analysis: Extending and resisting statistical disclosure. In Martin, D. and Serjantov, A., editors, *Privacy Enhancing Technologies*, volume 3424 of *Lecture Notes in Computer Science*, pages 784–786. Springer Berlin / Heidelberg.

Moore, R. (2008). Prospects for Cyber Deterrence. Master's thesis, Naval Postgraduate School, Monterey, California.

Morgan, P. (1977). *Deterrence : A Conceptual Analysis*. Sage Publications, Inc., Thousand Oaks, California.

Murdoch, S. J. and Zielinski, P. (2007). Sampled traffic analysis by internet-exchange-level adversaries. In *Proceedings of the 7th international conference on Privacy enhancing technologies*, PET'07, pages 167–183, Berlin, Heidelberg. Springer-Verlag.

National Defense Strategy of the United States (2005). *The National Defense Strategy of the United States of America*. United States Department of Defense, Washington, D.C.

National Defense Strategy of the United States (2008). *The National Defense Strategy of the United States of America*. United States Department of Defense, Washington, D.C.

NIST SP 800-39 (2010). NIST SP 800-39. Integrated Enterprise-Wide Risk Management: Organization, Mission, and Information System View.

Oman, P. and Cook, C. (1989). Programming style authorship analysis. In *Proceedings of the 17th conference on ACM Annual Computer Science Conference*, pages 320–326.

Osborn, A. (1910). *Questioned Documents*. The Lawyers' Cooperative Publishing Co., Rochester, NY.

Pouget, F. and Dacier, M. (2004). Honeypot-based forensics. In *AusCERT Asia Pacific Information Technology Security Conference 2004*.

Pras, A., Sperotto, A., Moura, G. C. M., Drago, I., Barbosa, R., Sadre, R., Schmidt, R., and Hofsted, R. (2010). Attacks by anonymous wikileaks proponents not anonymous. Technical report, Design and Analysis of Communication Systems Group (DACS); University of Twente, Enschede, The Netherlands.

Sinks, M. M. (2008). Cyber Warfare and International Law. Master's thesis, Air Command and Staff College, Air University, Maxwell Air Force Base, Alabama.

Spangler, R. (2003). Analysis of Remote Active Operating System Fingerprinting Tools. Technical report, University of Wisconsin - Whitewater.

Steinbruner, J. (1976). Beyond Rational Deterrence: The Struggle for New Conceptions. *World Politics*, 28(2):223–245.

Taipale, K. (2010). Cyber-Deterrence. *Law, Policy and Technology: Cyberterrorism, Information Warfare, Digital and Internet Immobilization*.

Tehranipoor, M. and Koushanfar, F. (2010). A survey of hardware trojan taxonomy and detection. *Design & Test of Computers, IEEE*, 27(1):10–25.

Thornburgh, N. (2005). The Invasion of the Chinese Cyberspies. *Time Magazine*.

Wheeler, D. and Larsen, G. (2003). Techniques for Cyber Attack Attribution. Technical report, Institute for Defense Analysis.

Zhuang, L., Zhou, F., and Tygar, J. D. (2009). Keyboard Acoustic Emanations Revisited. *ACM Transactions on Information Systems Security*, 13(1).

Zimmerman, J., Clark, A., Mohay, G., Pouget, F., and Dacier, M. (2005). The Use of Packet Inter-Arrival Times for Investigating Unsolicited Internet Traffic. In *Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering.*