Software Forensics for Adversarial Authorship

by

Rodney Visser

A Proposal submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
[Insert date of graduation, e.g., Aug 1, 2023]

Keywords: Software Forensics, Threat TTPs, Digital Provenance, Machine Learning, Deep Neural Networks, Random Forrest Classifier, Threat Intelligence, Authorship Attribution

Abstract

Software forensics in computer science can often be used to determine intellectual property rights and it provides potential for threat attribution. Risks from malicious software and the software supply chain are becoming increasingly an issue as code gets more complex and attackers continuously outpace defenders in the digital domain. The software development process has changed significantly and threats have been on the bleeding edge of this maturity in terms of capability and obfuscation. The history and modern motivation for such analysis of software is present and includes commercial and government organizations. This research investigates current static and dynamic methods of performing software forensics and improves on these methods by utilizing multi-dimensional analysis to detect semantic differences between two versions of a similar software code and then applying models in order to predict future changes in behavior.

Table of Contents

List of Figures

Chapter 1

Introduction

A quickly growing use case of software forensics is authorship attribution, the process of attempting to identify the likely authorship of a software sample given a collection of software whose authorship is known. Convention points to the use of a subfield of digital forensics, software forensics, to apply authorial context to the providence of malicious software. The categorization and mapping of key properties extracted from a suspect artifact can be used to select candidate matches from a pool of legitimate artifacts.

Currently, reverse engineering software artifacts suspected of being malicious code remains a labor-intensive and manual task, despite advances in automated programming understanding. Static and dynamic analysis tools provide insight into software structure and behavior, but the ultimate determination of what is a malicious artifact relies heavily on the skills and experiences of individual forensic investigators. Further complicating the process, attackers have also begun to place false flags within the technical breadcrumbs that are commonly left behind after a cyber attack. The means of which threats deliver malicious code has changed along with complexity of false flags. Typically modern threats interface with how users interact with the internet and network around them.

A cost effective example of malware delivery is through the use of phishing emails, where the content of the email tries to entice the recipient to click a URL linking to a malicious web site or downloading a malicious attachment. Analysts attempting to provide intelligence on such activities quickly find that the ever increasing volume of phishing emails circulating daily is overwhelming and the most effective phishing techniques are often not caught because of their use of the proper application of backstopping. Therefore, intelligence gathered within this area is only representative of only a small sample at a single point in

time, not of the global picture of the problem. Additionally, analyses have revealed that authorship attribution on this type of malware delivery method have provided linkages to the same author, but that it is unlikely that the authorship-based clustering algorithms have managed to group together all spam produced by each group. Said in another way, this is not a sustainable answer to authorship attribution on this type of malware delivery method on it's own because of the low rate of recall within APT groups. This problem of high precision with low recall has been faced in past authorship research [Li et al. 2017]. In broader terms, the issue of authorship attribution of malicious code and the various delivery methods suffer from problems related to precision and recall when analyzed on their own. This problem we propose can be assisted by multi-dimensional analysis using a novel framework.

A key issue in authorship attribution for cyber attacks is the lack of specific knowledge about technical capabilities, motivation, policies, and laws that currently govern this area. The difficult technical side of attribution is further escalated by serious legal and policy questions about when and how to accuse governments of responsibility for cyber attacks. Within the area of technical capabilities and motivations for groups that have been labeled as Advanced Persistent Threats (APT)s by commercial cyber security firms, there are sweeping differences in the arrangement, categorization, and number of named threats from nation states or criminal enterprises [Romanosky and Boudreaux 2019]. Below in Table 1.1 we provide a snapshot of the disconnect between leading cyber threat intelligence firms that track and publish intelligence on named APTs.

Table 1.1: Named APTs

Country	**FireEye**	**CrowdStrike**	**Kaspersky**	**Dell SecureWorks**	**Cisco Talos**
China	20	40	N/A	5	9
Russia	3	3	N/A	6	3
Iran	3	7	N/A	1	3
North Korea	1	4	N/A	2	2
Criminal / Terrorist	10	29	N/A	3	1
Other	2	1	N/A	2	-
Unknown/Undisclosed	8	1	N/A	-	1
Total	47	85	36	19	19

Although politics may largely determine whether attributions are made public, there is a need for a framework in which cyber attacks are attributed to states in a means governed by legal standards [Eichensehr 2020]. Experts on the topic of cyber threat attribution are quick to call for greater levels of regulation and policy that govern the world's critical infrastructure and economies, but for the most part, worldwide leaders in key positions do not understand the issues presented to them, the proper information is not made available, or the terrain of navigating the geopolitical landscape on this issue in the public eye is deemed risky.

A prime example of state-sponsored cyber espionage and influence operations is the unauthorized access and disclosure of the Democratic party's email in 2016 by a Russian State backed hacker group commonly referred to as Fancy Bear. The main point of dissemination for this operation was a self proclaimed Romanian hacker called Guccifer 2.0. Toni Gidwani, director of research operations for ThreatConnect, said: "It would suggest to us that the operators of the Guccifer 2.0 persona were not the actors who breached the DNC. You're looking at the operations guys who don't have the same technical credibility as these very sophisticated actors who exploited these networks. You've got a lot of cooks in this kitchen here, not just one actor." [Zager and Zager 2016]

This type of misdirection campaign for state-sponsored hacking is considered common place and automated single-dimension attribution techniques are easily guided in the wrong direction. Recent application of technology in the realm of source code authorship attribution methods have accuracy above 88%, but adversarial learning attack methods drop the rates of attribution to around 1% [Abuhamad et al. 2018], [Caliskan-Islam et al. 2015], [Quiring et al. 2019].

For these reasons, attribution must be thought of as a multi-dimensional issue that draws on multiple sources of information available, including technical forensics, human intelligence, signals intelligence, history, and geopolitics, among others [Lin 2016]. If successful, the attribution of successful adversarial attacks from code left behind on an infected system

could enable software forensics to be leveraged in order to be fight advancing and evolving threats.

1.1 Software Forensics

Within the questions of what, when, where, who, and how for digital forensic investigations, answering the who question is frequently the most elusive. The strategic aim of this research is to assist in answering this important question. The tactical objectives include a survey of current state-of-the-art techniques in adversarial authorship using software forensics and the application of machine learning to automate and innovate analysis.

Software-assisted detection allows vast collections of artifacts to be compared to each other, making successful detection much more likely than traditional manual methods. We refer to these first generation software-assisted means of detection as Single Dimensional Software Correlation. This method can be thwarted by simple compiler optimizations and anti-forensic techniques. Differences in hashing and mapping methods can cause a significant increase in false negatives count.

The motivation to use more focused and robust software forensic techniques is growing at an expanded rate in order to keep up with the adversarial techniques malware authors use to obfuscate their code. Cyber threat actors for the most part are out pacing the defenders at an increasing rate, modern software forensics tools, techniques, and procedures are growing, but not fast enough to lessen the gap with threats. In order to keep pace with intermediate and advanced level threat actors, novel techniques must be employed to find, track, and project how threats are evolving. It is our belief that adversarial authorship using software forensics can allow us to do this.

Chapter 2

Literature Survey

Using software forensics to track and plot the actions and potential growth of cyber threat actors includes problematic factors such as: lowered barriers of entry in terms of technical competence, malware as a service platforms, increasing levels of sophistication, and an rapidly expanding attack surface. The development of a novel graph pruning technique has shown that software correlation can be used to understand how malicious code has evolved over the years, and in particular, how different instances of malicious code relate to one another [Gupta et al. 2009]. Researchers established an inheritance relationships between different instances of malware based on temporal information and key common phrases identified in the malware descriptions and an extension of this research has the potential to show that software correlation can be used to inspect known examples of malware in order to gauge how threats and their code bases are successfully attributed and evolving.

Authorship attribution for malware has roots in traditional author identification in books and software intellectual property rights. Research in this area focuses on the identification of key fingerprints of the author and searching for similar fingerprints in other works. Extending this research to assist in the Attribution of Cyber Threats has some potential important impacts for the future. The dynamic geopolitical environment with interconnected networks of computers and most nations embracing of the cyber domain in a similar way that air, land, sea, and space are talked about make this a area of focus for research. This survey provides a brief historical analysis of cyber threats, their effects, and the role applying attribution plays. This survey's conclusions assist in discerning the benefits and potential pitfalls with cyber threat attribution and can be used to support future research strategies in this area.

Risks presented from cyber threat actors are increasing as systems get more complex and attackers continuously outpace defenders in the digital domain. The number of tactical capabilities that have transitioned from being run on hardware-based disconnected systems, to on-software based networked and interoperable systems has amplified the level of exposure for threats to interface. Many nations have embraced the lower barrier of entry into state-sponsored or supported cyber threat capabilities. The history and need for more research on cyber threat attribution is growing. Gray space critical infrastructure, along with blue space tactical and logistical programs are prime targets for nation-states to interface within the Cyber Domain.

A report from Sandia National Laboratories in 2007 on the categorization of threats implements a framework for establishing a common vocabulary, see Figure 2.1. A key motivation for this effort was to address the increasingly converged nature of how cyber threats integrate with kinetic threats and embedded computers. Two families of attributes categorize threats into a threat level profile based on discrete characteristics, or distinguishing properties.

The commitment attribute family is used to quantify a threat's drive and willingness to pursue a goal. The threats with the highest level of commitment will stop at nothing in pursuit of the end goal. On the opposite end of the spectrum, threats with less overall commitment will be less willing to commit time and resources to this goal. The intensity threat attribute is used to measure what a threat is willing to risk to accomplish its goal. A High (H) level of intensity means that a threat is highly determined to pursue its goal and is willing to accept any and all negative consequences to meet it. In contrast, a Low (L) level of intensity is not willing to accept negative consequences. Stealth is the threat's ability to maintain a necessary level of secrecy throughout the pursuit of its goal and range from highly capable to not capable. The third attribute for commitment is time measured in days, weeks, months, years, and decades. A good example of a disruptive technology that takes years to decades of planning and developing to deploy is the Stuxnet Worm.

THREAT LEVEL	THREAT PROFILE						
	COMMITMENT			RESOURCES			
	INTENSITY	STEALTH	TIME	TECHNICAL PERSONNEL	KNOWLEDGE		ACCESS
					CYBER	KINETIC	
1	H	H	Years to Decades	Hundreds	H	H	H
2	H	H	Years to Decades	Tens of Tens	M	H	M
3	H	H	Months to Years	Tens of Tens	H	M	M
4	M	H	Weeks to Months	Tens	H	M	M
5	H	M	Weeks to Months	Tens	M	M	M
6	M	M	Weeks to Months	Ones	M	M	L
7	M	M	Months to Years	Tens	L	L	L
8	L	L	Days to Weeks	Ones	L	L	L

Figure 2.1: Threat Profile [Woodard et al. 2007]

The resources attribute family are sets of characteristics that measure resources in the number, abilities, and access of personnel. The technical personnel threat attribute describes the group size of technical personnel that the threat can use to pursue the goal. The scope of this attribute is focused in specific areas of knowledge or skills, such as kinetic or cyber, and those directly involved with the actual fabrication of the group's capabilities. This area is broken into four levels based on number of resources and ability to communicate. The amount of and ability for personnel to communicate is key to support innovative design and development, but this alone cannot change the speed at which innovation takes place. The knowledge threat attribute is the proficiency of, support structure for, the threat's capability to apply the technical personnel to the goal. It is broken into 2 sub-categories (cyber and kinetic), with a high, medium, low rating is applied to each. Access is the last threat attribute and can be described as the ability for a threat to place a group member within a restricted system at a specific level of access. There are three level to this attribute, placement with

unlimited access, placement with limited access, no placement. An example of this would be the use of Close Access Teams (CAT) to provide physical access to restricted systems.

From all these data points a Threat Level can be applied that describes the capability of a threat to reaching its goal. Threat Level 1 is the most capable, and Threat Level 8 being the least capable. Threat Level 8 threat may be able to have the impact that a level 1 threat, but important details on characteristics, supporting infrastructure, and timing would differ that could be useful in the application of applying threat attribution. It is impossible to capture and categorize each threat attribute with 100% correctness, but this framework can be used to assist organizations in categorize threats using a common vocabulary.

In review of some historical examples of attributed cyber threats and their impacts on military policy we use the generic threat matrix presented above. In this review there will be descriptions from different times, areas, and scopes. This is not meant to be an exhaustive survey, but a varied of examples picked over time from different areas.

The Morris Worm was the work of a 23-year-old Cornell University graduate student Robert Tappan Morris Jr. Around 8:30pm on the 2nd of November in 1988 this worm was released upon the ARPAnet from a computer at the Massachusetts Institute of Technology (MIT). The worm contained multiple vectors to propagate from one Unix-based operating systems to another.

These vulnerabilities included:

1. A weakness in the sendmail program, which allowed the worm to send copies of itself to other computers via email.

2. A flaw in the fingerd program, which allowed the worm to execute arbitrary code on the targeted computer and gain access to user accounts.

3. A vulnerability in the rsh (remote shell) program, which allowed the worm to execute commands on other computers and propagate itself further.

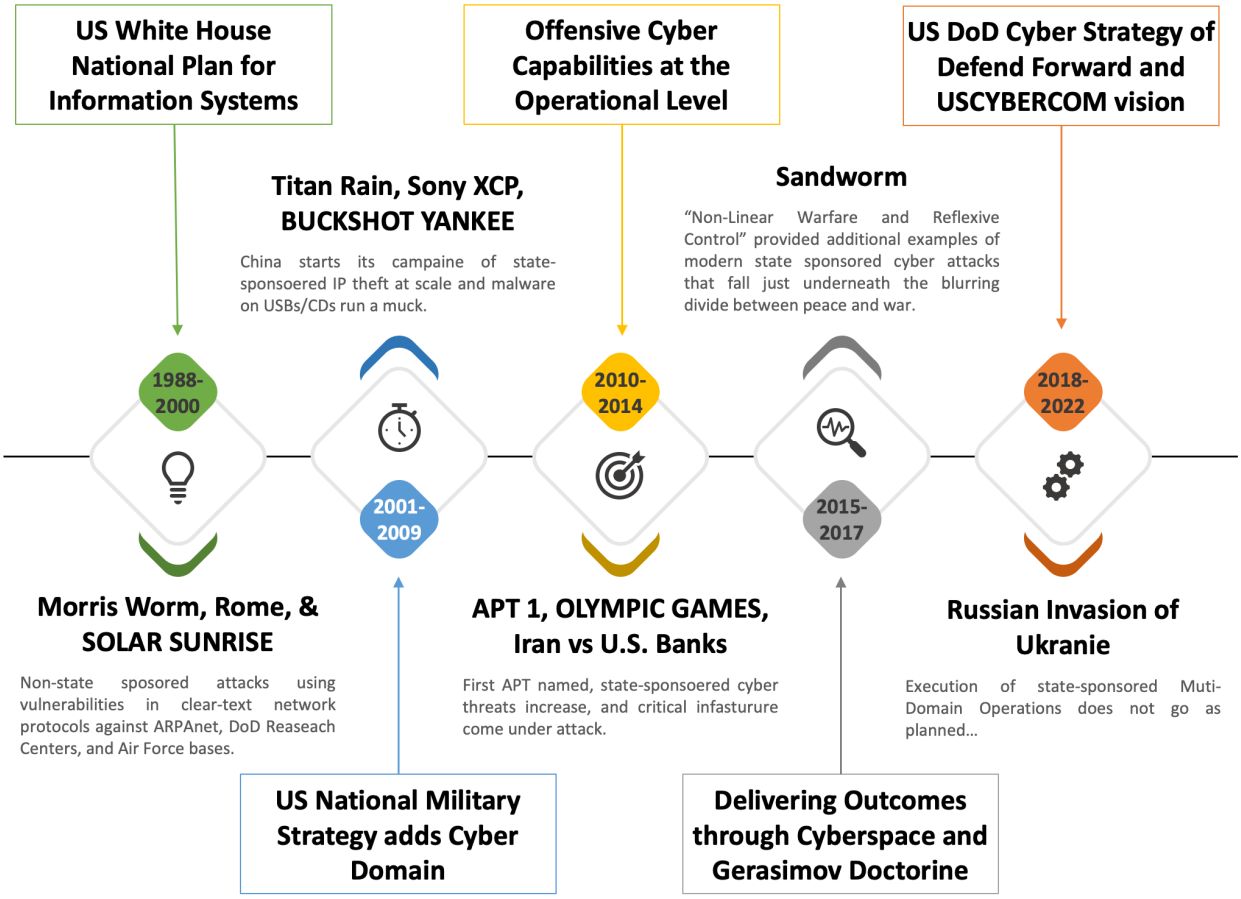

Figure 2.2: Timeline

4. A weakness in the password file system, which allowed the worm to bypass authentication and gain access to user accounts.

These vulnerabilities were not new, but the Morris worm was one of the first instances where they were exploited in a coordinated attack. The speed at which this happened was at the time unprecedented. Within 24 a hour time frame, an estimated 10% of the computers that were then connected to this network had been hit with it, around 6,000 of the 60,000 total at the time. The article in the New York Times that describes the incident was the first time that this periodical used the term " the internet" to describe the international group of computer communications networks that the worm used to propagate. Another interesting point about this event is that Mr. Morris's father was the chief scientist for a computer security arm of the National Security Agency.

Attribution for this attack was easily obtained through a combination of technical analysis and investigation by law enforcement agencies. Investigators were able to track the origin of the worm to a computer at MIT, which was owned by Morris. They also found evidence that Morris had tried to cover his tracks by modifying the code to remove references to his name and by using several different computers to launch the worm. Technical analysis of the worm's code also revealed that it contained a flaw that caused it to spread more rapidly than intended, which further implicated Morris as the creator. Morris eventually confessed to creating the worm and was convicted of violating the Computer Fraud and Abuse Act, becoming the first person to be prosecuted under the law. Specifically within the technical analysis, investigators at the Defense Advanced Research Projects Agency (DARPA) were able to analyze the network traffic generated by the worm and trace it back to its source. In addition to the technical analysis, investigators also conducted interviews with Morris's colleagues and acquaintances, which provided additional evidence linking him to the creation of the worm. Morris's own behavior, such as his attempts to cover his tracks and his evasiveness when questioned by investigators, further suggested his involvement in the creation of the worm. The threat profile using the generic threat matrix was an 8, because of the

low characteristics associated with stealth, time, number of technical personnel, and access. [Brassard]

In March 28th of 1994, system administrators at Rome Air Development Center, Griffiss Air Force Base in New York (Rome Labs) discovered their network had been penetrated and compromised using telephone modem connections. Initially, seven systems were found to be a part of the initial compromise that had a sniffer program installed on them that aided the attackers in gathering over 100 valid user account credentials. These credentials were used in the next 26 days to provide access to data on systems with in the US Air Force, Army, Navy, NASA, NATO, and even the South Korean Atomic Research Institute.

A compromise of this size of the defense industrial based had not been seen up to this point and was executed by two attackers with commercial off the shelf equipment. The hacker known online as the "Datastream Cowboy" was a 16 year old from the United Kingdom. He commonly used an internet provider based out of New York, mindvox.phantom.com, to establish a connection with Rome Labs. In order to mask his connection he frequently routed his communications through systems in multiple countries in Europe, South America, and Mexico. Datastream had launched these attacks with only a 25 MHZ, 486 SX desktop computer with a 170 Megabyte hard drive. This is a modestly powered system with limited storage capacity when compared to other consumer grade equipment at the time.

The second attacker, Kuji, was a far more sophisticated hacker than the 16 year old Datastream Cowboy. It took investigators more than two years after Datastream Cowboy's arrest to apprehend Matthew Bevan, the 21-year-old son of a local police officer. Kuji appears to have tutored Datastream Cowboy on how to break into networks and on what information to obtain. If an attack that Datastream Cowboy was conducting on a system failed in gaining him access, he would commonly engage in a 20-40 minute on-line "chat sessions" with Kuji. Following the on-line conversation, Datastream Cowboy was observed attack the same system again, but this time succeed. In return for mentoring Kuji commonly received a copy of the data the accessed. After Datastream Cowboy was apprehended,

he told investigators that he had never physically met Kuji and only communicated with him through the Internet or on the telephone. When interviewed, Kuji said that he was motivated by gaining "any information I could find relating to a conspiracy or cover-up of the UFO phenomenon." Additionally, he said he was bullied in school and the hacking world was pure escapism for him. Attribution for this attack required coordination between US DoD and UK authorities. The combination of technical analysis, forensic investigation, and law enforcement cooperation was needed to enable attribution. Firstly, investigators were able to trace the origin of the attacks to computers and networks associated with the two individuals. They also found evidence that the attackers had used a variety of hacking techniques and tools to gain unauthorized access to the Rome Labs network and exfiltrate sensitive information. In addition, forensic analysis of the compromised systems provided further clues as to the methods and techniques used by the attackers. One key piece of evidence that led investigators to Datastream Cowboy was his use of a specific hacking tool called "QAZ Trojan". This tool was found on the compromised systems at Rome Labs and was also used in other attacks that were attributed to Datastream Cowboy. Investigators were able to analyze the QAZ Trojan and identify unique characteristics that allowed them to link it to the same individual.

The QAZ Trojan had a number of capabilities that made it a powerful tool for hackers. These included:

1. Backdoor access: The Trojan provided a backdoor into a compromised system, allowing an attacker to access the system at any time without the user's knowledge.

2. Remote control: The Trojan allowed an attacker to remotely control a compromised system, including executing commands, uploading and downloading files, and modifying system settings.

3. Keylogging: The Trojan was capable of logging keystrokes, which allowed an attacker to capture passwords and other sensitive information.

4. Screen capture: The Trojan could take screenshots of a compromised system's desktop, which allowed an attacker to monitor the victim's activities.

5. File transfer: The Trojan allowed an attacker to transfer files to and from a compromised system, which could be used to exfiltrate sensitive data or upload additional malware.

A key piece of evidence that led investigators to Kuji was his use of a specific hacking tool called "Vorpal". This tool was found on the compromised systems at Rome Labs and was also used in other attacks that were attributed to Kuji. Investigators were able to analyze Vorpal and identify unique characteristics that allowed them to link it to the same individual.

Secondly, law enforcement agencies were able to gather intelligence on the activities of Datastream Cowboy and Kuji through their cooperation with foreign law enforcement and intelligence services. Both individuals were eventually arrested and charged with computer-related crimes, which further solidified the attribution of the attacks to them. Intelligence was gathered on both hackers activities and online presence. They identified online forums and chat rooms where they communicated with other hackers and discussed their activities. They also traced their use of various online pseudonyms and aliases, which helped to build profiles of their online activities. The threat profile using the generic threat matrix was an 6, because of the characteristics associated with stealth, intensity, number of technical personnel, and access. [Delibasis Fokas 2007]

The pattern of successful computer intrusions executed against the DoD by small groups of non-state-sponsored hackers sprinkled across the planet did not stop with the Datastream Cowboy and Kuji. A common theme within successful network intrusions at this time was the use of known vulnerabilities to exploit systems and the blanketed use of clear text network authentication protocols that enabled attackers to propagate across shared networks quickly. Typically, these attackers manually exploited systems using low to medium level cyber threat techniques.

This pattern continued and in January of 1998 tensions escalated between the United States, the United Nations, and Iraq because of the recent expulsion of UN weapon inspectors by Saddam Hussein. On Feb 3rd and 4th root-level compromises were detected across multiple US Air Forces Bases within the continental united states that was called SOLAR SUNRISE. The cyber-attack managed to get access to more than 500 computers which belonged to the US military, US government, and private defense sector. In the attack the hackers used a virus against systems running the Solaris operating system. At this point in time there was no known virus found on the Solaris or SunSolaris systems and this was the first attack of its kind against this operating system. The vulnerability that the attackers used to gain an initial foothold was a known *rpc* vulnerability called *statd*. This vulnerability allowed the attackers to remotely execute arbitrary commands of their choice using the privileges of the statd service. From there the attackers installed backdoors, keyloggers, and network sniffers to enable persistent and additional access to systems.

This computer hacking incident quickly turned into a multi-agency effort that saw unprecedented interagency cooperation between the Army, Navy, Air Force, FBI, NASA, CIA, NSA, and others. This unusual level of cooperation quickly resulted in the DoJ being able to obtain nine court orders in fewer than 10 days and a Title III wiretap that was approved in one day. In the end, this attack was not executed by Iraqi state-sponsored hackers, but two 16 year old hackers from Northern California and an 18 year old from Israel. They named themselves as "the Analyzers" and this incident gave the very first nation state level cyber-war alarm which proved to be nothing but a false-alarm. The investigation that led to the attribution of the Solar Sunrise attack to three teenagers, two from California and one from Israel, involved a combination of technical analysis and law enforcement techniques. The investigation was led by the US Air Force Office of Special Investigations (AFOSI) and involved collaboration with other US government agencies. Investigators used a range of techniques to trace the attack back to its source, including analyzing network traffic, examining logs and other forensic evidence, and interviewing personnel who may have had

knowledge of the attacks. A key breakthrough in the investigation came when investigators discovered that one of the attackers had used a personal email address that was associated with his real name. This led investigators to identify the individuals responsible for the attack as Dhananjay "Jay" R. Desai and Aaron D. Hesse, both from California, and Ehud "Tim" Tenenbaum from Israel. The attribution of the Solar Sunrise attack to these three individuals was seen as a significant achievement in the field of cyber-investigations, as it demonstrated that even sophisticated attacks on government computer systems could be traced back to their source and that perpetrators of such attacks could be held accountable for their actions. Discovering of this kind of vulnerability forced US Defense department to take strong steps to avoid such thing in the future. [Purohit] The threat profile using the generic threat matrix was an 8, because of the characteristics associated with stealth, intensity, number of technical personnel, and access.

While the 42nd President of the United States was in office in January of 2000, the White House published a unclassified paper titled "Defending America's Cyberspace: National Plan for Information Systems". A key motivation for publication of the plan was to facilitate public-private sector cooperation on information systems protection. This plan outlined a number of defensive milestones and accomplishments, but did not address the application of offensive network capabilities to the DoD or applying attribution to attacks against information systems. [Clarke 2000]

In 2003, before cyberspace was declared to be a war fighting domain, there was a series of attributed computer network intrusions labeled TITAN RAIN that US Defense Department investigators believed to have originated from the People's Liberation Army of China. These series of intrusions marked a turning point for the DoD in which the attributed activities were shown to have not been conducted by any free-lance hackers but long term operations conducted by state-sponsored hackers. The sensitive information that was gained from this series of attacks came from a variety of US defense contractor computer networks, Sandia National Laboratories, US Army Space and Strategic Defense Command, FBI, NASA, and

the United Kingdom's Ministry of Defense. The attribution of the cyber threat attacks called TITAN RAIN was based on a combination of technical analysis, intelligence gathering, and collaboration between multiple organizations.

Some of the key information that aided in the attribution of the attacks included the following.

1. Technical indicators: Analysts identified a set of unique technical indicators associated with the TITAN RAIN attacks, such as specific malware samples, command-and-control infrastructure, and methods of exploitation.

2. Network traffic analysis: Investigators analyzed network traffic logs from affected organizations to identify the source of the attacks and the tactics, techniques, and procedures used by the attackers.

3. Intelligence gathering: Intelligence agencies used a range of sources and methods to collect information about the identity and location of the attackers, including human intelligence, signals intelligence, and imagery intelligence.

4. Collaboration between organizations: Multiple organizations, including the FBI, the National Security Agency (NSA), and private security firms, worked together to share information and coordinate their efforts in investigating the attacks.

5. Attribution to specific threat actors: Ultimately, the TITAN RAIN attacks were attributed to a group of Chinese state-sponsored hackers known as the "Byzantine Candor" group, based on a combination of technical and intelligence analysis.

In a similar vein to the Chinese government's response to the Mandiant report in 2010 and 2013, China refuted the charges of being involved in the TITAN RAIN attacks and suggested instead that the network security of Chinese computers and websites were lacking in such a severe way that they could have been exploited by any random hackers across the

world. This was a convenient answer, but did not line up with evidence of the size, complexity, patience, coordination, discipline, and professional manner in which the attacks were carried out. The Director of Research of SANS Institute at the time, Adam Paller, stated that "the attacks were from individuals with intense discipline and no other organization except military could do this". [Cyware Hacker News 2016]

This was a key turning point in the domain of cyber and how threats influenced military strategy across the globe. The era of advanced persistent threats began and nation-states started building resources to gain enhancing capabilities in this area to assist in a variety of national priorities that ranged from areas of intelligence, military capabilities, economic growth, and gaining geo-political status. The threat profile using the generic threat matrix was an 3, because of the intense application of resources and level of commitment that was not common place at the time.

In the 2004 National Military Strategy, the Chairman of the Joint Chiefs of Staff Richard Myers declared cyberspace a "domain" of conflict alongside the air, land, sea, and space domains. [Myers 2004] It was noted that the US DoD must maintain its ability to defend against and to engage enemy actors in this new domain. That same year the Secretary of Defense Donald Rumsfeld divided Joint Task Force – Computer Network Operations (JTF–CNO) into defensive and offensive components. [Department of Defense 2011] The component that held the responsibility for defense of the cyberspace domain was the Joint Task Force – Global Network Operations (JTF–GNO) and the component that held the responsibility for offensive cyberspace operations planning was the Joint Functional Component Command – Network Warfare (JFCC–NW).

The commercial company Madiant started to provide detailed reporting on what it called Advanced Persistent Threat (APT) 1 in 2010. The position of Mandiant was that "The Chinese government may authorize this activity, but there's no way to determine the extent of its involvement." That changed in 2013 when they reported that now we have the evidence required to change our assessment to say that the groups conducting these

activities are based primarily in China and that the Chinese Government is aware of them. [Mandiant 2013] In response to this update, the Chinese Defense Ministry provided this "It is unprofessional and groundless to accuse the Chinese military of launching cyber attacks without any conclusive evidence." [Lewis 2005]

Operation BUCKSHOT YAHKEE was a response plan executed in 2008 to clean up a worm from DoD networks called agent.btz. Unlike earlier worms described in this survey, the primary means of propagation this worm leveraged the windows auto-run feature and the abundant use of usb thumb drives to move data between systems. Agent.btz was variant of the SillyFDC worm that had been observed being used by Russian hackers before. Some of its capabilities included abilities to scan computers for data, open backdoors, and send through those backdoors to a remote command and control server. One unique feature of SillyFDC is its ability to modify its code and behavior in response to changing system conditions. For example, it can detect if it is running in a virtual machine environment and modify its behavior to avoid detection. This family of malware was unique in that it also attempted to disable antivirus software and other security measures on the infected system, making it more difficult to detect and remove. The agent.btz worm was particularly challenging to detect and remove due to its use of advanced encryption techniques to hide its activity and evade traditional antivirus and security tools. It was also designed to remain dormant for extended periods, allowing it to evade detection and continue to spread across networks. It is believed that initial infection was facilitated by a foreign intelligence agency that left an infected thumb drive in the parking lot of a DoD base in the Middle East. From there it to was observed spreading across unclassified and classified information systems at US CENTCOM. Attribution of the agent.btz worm being used against CENTCOM is not entirely clear, but it is believed to have been the work of a foreign intelligence agency, possibly Russia or China. This incident highlighted the vulnerability of even the most secure military networks to sophisticated cyber attacks.

Three years before this, in 2005 Sony BMG had found itself in trouble becasue of the development and opperational use of the windows auto-run feature and a piece of Digital Rights Management (DRM) software called the Extended Copy Protection (XCP) rootkit. Sony had been losing revenue because of peer-to-peer file sharing software. Statements by Sony Pictures Entertainment US Senior VP Steve Heckler foreshadowed these events when he told attendees of the Americas Conference on Information Systems in 2000 that "The industry will take whatever steps it needs to protect itself and protect its revenue streams... It will not lose that revenue stream, no matter what... Sony is going to take aggressive steps to stop this. We will develop technology that transcends the individual user. We will firewall Napster at source – we will block it at your cable company. We will block it at your phone company. We will block it at your ISP. We will firewall it at your PC... These strategies are being aggressively pursued because there is simply too much at stake." The combination of this aggressive stance and Sony's application of a malicious software in their products did not help their cause.

The DoD did not effectively learn from Sony's mistake, but did take measures to clean the infection and install security controls to help ensure this particular attack method was mitigated going forward. The it took the Pentagon nearly 14 months to intercept the covert channel that this worm used and clean the infection from military networks. Many changes were implemented in terms of people, processes, and technology that directly effected American military strategy and good deal of lessons were learned in having better situational awareness of military networks. The threat profile using the generic threat matrix was an 8, because of the characteristics associated with stealth, intensity, number of technical personnel, and access, but to this date public attribution has not been confirmed. [Lynn III 2010]

In June of 2009, the Iranian nuclear power enrichment plant in Natanz came under attack by an updated version of what would be called the world's first digital weapon, Stuxnet. In a similar lane to the Agent.btz worm, Stuxnet used usb thumb drives as the primary

prorogation method, but the similarities in commitment towards the goal and resources applied stop there. [Zetter 2014] This worm was designed to target industrial control systems, specifically those used to control centrifuges at an Iranian nuclear facility. The complexity and sophistication of the worm led many experts to believe that it was the work of a nation-state.

The primary differences in the commitment and resources attribute families project a sharp contrast from previous references in this survey. Evidence in the amount of time that this piece of malware was under development can date back to around 2005. Lines of code is also a significant outlier when compared to other malware samples collected within the same time frame. Sophos, a cybersecurity company specializing in malware analysis and enterprise security occasionally publishes some of their analysis on trends found within malware. In 2010 the analysis provided by Sophos in this area described most malware contained around 125 lines of code. Stuxnet was shown to contain more than 15,000 lines of code. Another significant difference was the use of 4 different zero day vulnerabilities used to get the malware to its intended destination.

Direct public attribution has not been definitively verified, but most researchers agree that the evidence points to a collaborative effort by the US and Israeli governments within Operation Olympic Games. It has been reported that the development of Stuxnet began under the George W. Bush administration and continued under the Obama administration. The operation is considered a landmark in the history of cyber warfare, as it demonstrated the potential for cyber attacks to cause physical damage to critical infrastructure. The threat profile using the generic threat matrix was an 1, because of the unprecedented characteristics associated with stealth, intensity, time, number of technical personnel, access, and kinetic effect that the Stuxnet worm had. [Kamiński et al. 2020]

In response to the Stuxnet worm and ongoing economic sanctions the Iranian government made a retaliating strike on the US. In another turn of events that seems to be a modern trend for state-sponsored cyber attacks, this attack was pointed not to the Israeli

Mossad, the US Central Intelligence Agency, or the US military, but instead they attacked important pieces of gray space critical infrastructure, including banks, oil manufacturing, and cellular service providers.

In Feburary of 2012 the Director of National Intelligence James R. Clapper Jr. told the US Congress that "Iran's intelligence operations against the United States, including cyber capabilities, have dramatically increased in recent years in depth and complexity." Specifically, Iran's Ministry of Intelligence and Security and a special unit within Iran's Revolutionary Guard Corps called the Quds Force are suspected of carry out this and other attacks against critical infrastructure. Most of these attacks were simple distributed denial-of-service attacks designed to overload an organization's web site and block legitimate access to the servers. The threat profile using the generic threat matrix was a 7, because of the characteristics associated with stealth, intensity, number of technical personnel.

The Center for Strategic & International Studies (CSIS) and Georgia Tech Research Institute (GTRI) published a pap er in 2013 titled "Offensive Cyber Capabilities at the Operational Level, The Way Ahead". The paper examines in greater depth whether the Defense Department should make a more deliberate effort to explore the potential of offensive cyber tools at below that of a combatant command. It details the growing need for offensive cyber capabilities at echelons below Combatant Commands (COCOM)s and the struggle of limited resources, lack of understanding of the capabilities, proper conditions of use, not well-developed requirements, and immature tool sets. In the end this paper presents the opinion of "At present, neither the procedures nor the tools are sufficiently robust to merit a delegation of offensive cyber authorities beyond the very limited ways in which they have been utilized thus far." [Nakashima 2012]

A research paper published by the NATO Defense College in November of 2015 titled "Russia's Renewed Military Thinking: Non-Linear Warfare and Reflexive Control" provided additional examples of modern state-sponsored cyber attacks that fall just underneath the blurring divide between peace and war in the 21st century threat landscape. In this paper,

evidence of the adoption of the "Gerasimov Doctrine" is presented. Additionally, the book SANDWORM by Andy Greenberg provides a detailed timeline of cyber attacks executed by the Kremlin.

To support this new shift in military doctrine, General Valery Gerasimov, the Chief of the General Staff of the Russian Federation's Armed Forces, argues that "the role of non-military means of achieving political and strategic goals has grown, and, in many cases, they have exceeded the power of force of weapons in their effectiveness." The doctrine describes ways Russia is leveraging cyber attacks and organizing resources within unified informational spheres towards their mission goals. Examples of these efforts are the state-sponsored attacks against Estonia and Georgia that support Russia's goal of necessary expansionism. The threat profile using the generic threat matrix was a 5, because of the characteristics associated with stealth, intensity, and balance between cyber/kinetic/access that aligns with the Gerasimov Doctrine.

The cyber attack referred to as the NotPetya worm is an example of the greater level of need for advancing software forensic tools in order to better help determining adversarial attribution. The NotPetya worm was created by a group of Russian military intelligence hackers known as Sandworm, and was intended as a climactic strike against Ukraine in the years-long cyberwar Russia had carried out against its southwestern neighbor. This worm presents an interesting case because of the unique combination of using stolen National Security Agency hacking capabilities, an open-source credential-harvesting tool called *Mimikatz*, and the hijacking of a update system within a common piece of Ukrainian accounting software that was used used by practically every company that filed taxes or had business ties in the country. [Greenberg 2018]

This lowering of resources needed to build and integrate this capability, in combination with nation states conducting advanced offensive cyber operations, in different instances, at scale and with precision, below the threshold of traditional war requires new ways of thinking. Specifically, when focusing in on the previous history of applying cyber threat

attribution either incorrectly or not in a timely manner, there is significant risk to critical infrastructure. New ways of applying attribution in this domain is required. Within this timeline of events there are opportunities for advancement within the areas of analysis that can be automated and correlated for the benefit of accuracy in cyber threat attribution.

Current US policy for conflicts in the cyber domain is an evolving area that include a number of key stakeholders. US Cyber Command (USCYBERCOM) released an updated vision in April of 2018 that built upon the USCYBERCOM commander's vision released in 2015. The vision in 2015 was titled, Delivering Outcomes through Cyberspace. The updated vision described the growing need, dynamic nature, and interconnection between cyber and other warfighting domains. "The cyberspace domain that existed at the creation of US Cyber Command has changed. Our adversaries have exploited the velocity and volume of data and events in cyberspace to make the domain more hostile." A key point of this change lies within what is described as a new normal way of interacting with the United States in cyberspace. Adversaries continuously operate against the US below the threshold of armed conflict without fear of legal or military consequences. Similar issues with cyber threat attribution are still present in 2018 that Madinat attempted to overtly call out in 2010. [U.S. Cyber Command 2018]

An important shift in focus of this document was apply the concept of "defend forward" in order to disrupt or halt malicious cyber activity at its source, including activity that falls below the level of armed conflict. This is a step in the right direction to help create policy that better address state-sponsored cyber threat actors attack on gray space infrastructure, but greater levels of focus are needed on policy and efforts for the DoD to better apply attribution to cyber threats and provide Offensive Cyber Capabilities at the Operational Level. [Goldsmith 2022]

In July of 2019 there was a hearing to consider the nomination of General Mark A. Milley to be the Chairman of the Joint Chiefs of Staff at the 116th US Congress. Within this nomination hearing there were many questions related to cyber threats and the change

from a counter-insurgency focused military to a near-peer battle. Chairman of the Senate Armed Services Committee, U.S. Sen. Jim Inhofe set the stage with "The National Defense Strategy makes it clear that strategic competition with China and Russia, not terrorism, is now our primary national security concern. China and Russia have passed us in key areas and are catching up with others." This are key points of reference regarding the change in military strategy and in influence the cyber domain has in modern military. When General Milley was asked questions on the theory of deterrence within the cyber domain, a parallel to a particular phase of American football was used, "a good offense is critical and that is the best defense".

This emphasis on the lethality of the U.S. military's offensive capabilities, especially in the context of the "defend forward" concept is concerning because it still suffers from similar drawbacks that CSIS and GTRI report outline in 2013. To overmatch and disrupt or halt malicious cyber activity at its source, attribution needs to be better addressed and capabilities need to be pushed towards the tactical edge to maintain superiority in the cyberspace domain. Russia's invasion of the Ukraine in February of 2022 illustrates the interseation of modern warfare domains. There could be a great number of reasons for this result to be taking place. On March 8th, 2022, during a House Permanent Select Committee on Intelligence hearing on worldwide threats, Director of the Central Intelligence Agency William Burns describes four false assumptions that the Russian dictator has made:

"First: That Ukraine, in his view, was weak and easily intimidated. Second: That the Europeans, especially the French and Germans, were distracted by elections in France and a leadership succession in Germany and risk averse. Third: He believed he had sanctions-proofed his economy in the sense of creating a large war chest to foreign currency reserves. And fourth: He was confident that he had modernized his military and they were capable of a quick, decisive victory at minimal cost."

The fourth assumption plays a key role in potentially applying assistance in what and how a modernized military need operate to be prepared for a conflict with a near-peer.

For the U.S. DoD to defend froward and apply overmatch with cyber in this new normal environment, research is needed to investigate our ability to apply cyber threat attribution. Improved performance of software forensics utilizing multi-dimensional analysis to detect semantic differences between known and unknown cyber threat actors can assist in this endeavor of better applying authorship attribution.

In order to use software forensics to track advancing threats like that of NotPetya, the software forensics engineer needs to break the inputs into objects suitable for comparison and apply structure to these objects. This structure can then be used to determine which objects exist across various versions and which do not. This is not a simple task since the differences in high-level language constructs and structure, including code comments, are commonly not found in the compiled versions of the malicious software. Techniques used to identify the fingerprints of malicious software authors come from a variety of sources and modern authorship identification techniques can be used to highlight patterns. Previous methods of manual analysis do not meet the scale of which current malware is being manufactured.

2.1 Single Dimensional Software Correlation

Single Dimensional Software Correlation takes a single input source and uses that source to as it's basis for analysis. Historically this is how advanced cyber threats are identified and tracked. This method of correlation is a labor intensive task of subject matter experts. Depending on the experience and skill of the analysts, accuracy can be very high, but does not scale and can be influenced though the placement of false flags. Some parts of this analysis can be automated, other parts are not easily automated. This analysis determines the similarity of source or binary code, by comparing raw text, cryptographic hashes, and tokens. The result is a single measure that indicates the degree of similarity. This measure is not accurate enough to be admissible in court of law because they are not definitive. Questions arise because the algorithms can be fooled by, for example, simple substitutions

in the code with representative fixes. Similarly, the problem of instruction reordering can be solved though grouping using the variable tracing and sorting method [Oh 2009].

[Zeidman 2012] developed algorithms that look at the basic elements of code in order to find which elements are correlated to each other. He proposes an algorithmic method that extracts program elements that are common in the samples being compared. These methods keep each of the data structures that are found to have entries corresponding to program elements of a distinct program element type represented by program strings or program identifiers. From that, elements are extracted based on a program object code file using a text converter to transform the program object code to text sequences by solely determining byte-by-byte whether the sequences represent characters. Then a calculation of a correlation score based on the similar entries is done that is comprised of a number of similar strings and a number of similar identifiers.

Another single dimensional software correlation technique is analysis of code comments. Once analysis has been performed using a single dimension procedure, an analyst then looks at subjective evidence such as comments in the code. Fake copyright notices, open source notifications, or programmer names added to source code after copying took place, in order to disguise the copying, are not uncommon in real-world cases of code theft [Zeidman 2011]. Challenges arise in the ways that code comments are written using natural language, complicating automated means to understand the intent of comments even when armed with the latest Natural Language Processing (NLP) techniques [Bird et al. 2015]. As a benefit, code comments contain a rich amount of information that can be leveraged to improve attribution. Analyzing free-form and semi-structured code comments can assist in the unique identification of characteristics and content of code comments.

The value of code comment analysis is improved when presented with strong pieces of supplemental data. Information such as how an organization distributes its code, details of the software development life-cycle can possibly emerge. This gives the potential to identify the programmers who authored the code.

There is also work being done on data sets of different programming languages (Java or C++) of varying difficulty (6 to 30 candidate authors) to demonstrate the effectiveness of the Source Code Author Profiles (SCAP) approach. It involves analyzing characteristics of source code to identify patterns and unique attributes that can help identify the author or authors of the code. This approach takes into account various factors such as the use of certain programming languages, coding styles, and coding conventions. By analyzing these characteristics, SCAP can help identify the individual or group responsible for creating the code, which can be helpful in identifying cyber threats or potential attacks. This is based on analysis of byte-level contiguous sequence profiles in order to represent a source code author's style. [Frantzeskou et al. 2006] This method has potential to show effectiveness of the model and how it is not severely affected by the absence of comments in the source code, a condition usually met in cyber-crime cases.

There have been several instances where the SCAP approach has been used to aid in cyber threat attribution.

1. Attribution of the 2017 NotPetya attack: In 2018, the UK and US governments attributed the 2017 NotPetya attack to Russia based on evidence gathered using the SCAP approach. The UK government's National Cyber Security Centre (NCSC) analyzed the malware's source code and used the SCAP approach to link it to a specific Russian military intelligence unit.

2. Attribution of the Lazarus Group: The SCAP approach has also been used to link the Lazarus Group to a specific North Korean research institution. In 2018, the US Department of Justice indicted a North Korean programmer who was believed to be a member of the Lazarus Group. The indictment included details of the SCAP approach being used to identify the individual based on his code and development patterns.

3. Attribution of the Anthem data breach: In 2015, the US health insurer Anthem suffered a data breach that exposed the personal information of millions of customers. The

SCAP approach was used to link the attack to a specific Chinese hacking group known as Deep Panda. The attribution was based on similarities between the Anthem attack and previous attacks attributed to Deep Panda, including the use of specific code libraries and development patterns.

The analysis of malicious code authorship and true functionality without the assistance of the underlying source code makes single dimensional software correlation difficult and a prime target for deception. Comparing code functionality is a difficult problem that has yet to be effectively shown by any algorithm within a reasonable time. For this reason, the process of applying attribution to a code's author is still mostly manual.

2.2 Multidimensional Software Correlation

Multidimensional software correlation is the process of dividing attributing features into discrete categories in order to determine which elements assist the most in attribution. One of the key elements to applying this correlation is the ability to filter and interpret the relationships in order to find the best features and eliminate false positives. The state of research of semantic data models being leveraged for machine learning has shown promise in helping identify similar elements for correlation. One example of this work is the concept of traceability being applied using machine learning within safety-critical domains for source code and other artifacts using Word Embedding and Recurrent Neural Network (RNN) models to generate trace links. RNN models use word vectors to learn the sentence semantics of requirements artifacts. An RNN system that uses Bidirectional Gated Recurrent Units (BI-GRU) has shown promise for the tracing task. [Saha et al. 2019] BI-GRU, in this instance, significantly out-performed state-of-the-art tracing methods including the Vector Space Model and Latent Semantic Indexing. This provides us an example of a solution that leverages deep learning to incorporate artifact semantics into a novel solution. [Guo et al. 2017] This method stands in stark contrast to the labor intensive job of manual expert selection done by an analyst described in previous section on Single Dimensional Software

Correlation. In most modern implementations of Multidimensional Software Correlation to aid in cyber threat authorship attribution all available features are used and little manual work or checks and balances are leveraged in terms of expert selection.

An important area of focus for addressing attribution using software correlation are the range of features available to apply efficient and effective attribution. Research being done within the area of time-series forecasting methods for cyber attacks, from network telescopes, honeypots, and automated intrusion detection/prevention systems is limited to awareness of predesignated items within a temporal scope that can be easy to miss and hard to collect. "To enhance awareness about specific threats, it is vital to uncover associated and, ideally, causal factors for cyber attacks" [Bakdash et al. 2017]. For our research endeavor, this data can be used to assist in attribution based on past observation of transit data path, means, and capabilities in combination with other features already identified using expert selection. These supporting factors can be used in order to further judge breadth, depth, and positioning of future attacks in traditional and new environments that threats are working towards having similar capabilities that exist in client-server environments.

Accurately detecting malware signatures and anomalies aids in the process of applying attribution. A short review of malware detection is given to draw similarities to features that can be used to guide better results within authorship attribution. We describe this research in two main different types, syntactic vs semantic analysis. An easy way to describe the differences in these techniques is a pattern matching approach and a trait detecting approach.

Syntactic detection of modern malware is fraught with issues, a key one being the use of self-mutating software. A good example that describes the cat and mouse game of malware detection is syntactic analysis of payload generation and encoding capabilities of the open source exploitation framework *metasploit*. Shikata-Ga-Nai is a Japanese language phrase meaning "it cannot be helped" or "nothing can be done about it" and for many years this was an accurate description of the ability of syntactic analysis tools to trigger

on this specific method of encoding that is built into the metasploit framework. At it's core, it is a polymorphic XOR additive feedback encoder. This encoder offers three features that provide advanced protection when combined. First, the decoder stub generator uses metamorphic techniques, through code reordering and substitution, to produce varied output each time it is used. This makes triggering on just the shellcode very difficult, especially if unique parameters are leveraged. This is implemented using loops with a user definable counter. Second, it uses a chained self modifying key through additive feedback. This means that if the decoding input or keys are incorrect at any iteration then all subsequent output will be incorrect. Third, the decoder stub is itself partially obfuscated via self-modifying of the current basic block as well as armored against emulation using Floating-Point Unit instructions.

APT20, a suspected Chinese nation state-sponsored threat group, leverages the Shikata-Ga-Nai encoder in their payloads. This group has a primary focus on stealing data, specifically intellectual properties. Other named groups include APT41 and FIN6. APT41 has been seen using this encoder within custom developed backdoors. APT41 is a Chinese cyber threat group that has been observed carrying out financially motivated missions coinciding with cyber-espionage operations. The financially focused threat group FIN6 also uses this encoding method to carry out some of their missions, and they have historically relied upon various publicly available tools. These missions largely involve theft of payment card data from point-of-sale systems.

Static Malware Detection by the use of mechanisms to detect string signatures, byte-sequences, n-grams, syntactic library calls, control flow graph and opcode frequency distribution of known malware have helped keep pace with obfuscation, but some of these measure are trait detecting approaches. Semantic malware detection measures that examine intermediate languages have been shown to provide greater levels of resilience against the polymorphic capabilities of modern malware [Christodorescu et al. 2005], [Ranjan et al.

2016]. These methods use LLVM generators to convert machine code to simplified forms in order to aid in semantic analysis that can be leveraged in identification of authorship traits.

Provenance, a word that originates from art, refers to the chronology of location and ownership. A detailed provenance can be used to establish that a piece of art is not a forgery or has not been stolen. Recently, the term provenance has also been adopted and applied to other fields, including computer science, where it refers to having knowledge of all the steps involved in producing a scientific result, from experiment design through acquisition of raw data and all the subsequent steps of data selection, analysis, and visualization [Davison 2011].

The concept of provenance can also be applied to multidimensional software correlation. The tracing of provenance in software development has become increasingly more important to tracking vulnerabilities and threats. Models of provenance developed from observing software during its natural life cycle can be used to educate researchers on how code evolves and mutates over time. The relation between a large file and LOC objects is very sparse and indexing can allow the relations to mapped as clusters of LOC and their containing files. Using these observations, an analyst is able to observe statistical trends of modifications made to files over time and see changes in sparse structures and natural growth patterns throughout a software development life cycle. In a typical scenario, files exhibit rapid initial growth with code being added in large chunks, and then when the changes of the software are in place, small bug fixes are the majority of modifications thereafter. By tracking the unique relationship between file size and lines of code, an analyst can map codelets as code patches with co-migratory patterns in files. [Davison 2014] For instance, a generative model that uses a Bayesian game to assist in Active Malware Analysis (AMA) has shown to have strong statistical results in identifying relations in software provenance between malware families [Sartea et al. 2020].

As an extension of this concept of provenance, experiments have shown benefits in the selection of informative features using genetic algorithms (GA). Specifically, authorship

attribution of social media and literary texts using machine learning methods in combination with GA has shown the ability to reduce the time costs by half in comparison with deep Neural Networks alone in comparable accuracy. [Fedotova et al. 2021] In addition to an improvement of time, cost though the implementation of Neural Networks and GA, there have also been a series of comprehensive experiments that present results that significantly increase the rate of accuracy in terms of recognition. Focusing feature selection on syntactic attributes for authorship attribution using multi-objective genetic algorithm and a Support Vector Machine (SVM) classifier has been shown to increase the recognition rate around 15 percentage points. [Varela et al. 2011]

Traditional grammar-based analysis techniques used to identify authors have been adapted for use in software source code ([Kustanto and Liem 2009], [Lesner et al. 2010], and [Jadalla and Elnagar 2008]). Code, like speech, consists of patterns that can link the author to the product. The langage Prolog offers advantages for a thorough analysis in two main parts. In one part, it natively provides versatile options to efficiently process tree or graph data structures. In another part, Prolog's non-determinism and backtracking eases tests of different variations of the program flow without a large level of effort. A rule-based approach with Prolog allows the characterization of verification goals in a concise and declarative way. This is why the Prolog programming language is currently being used to do source code verification for embedded aerospace systems [Flederer et al. 2017].

Multidimensional software correlation can be leveraged in order to collectively analyze more data inputs that single dimensional software correlation, but this does not necessarily mean that the results will be more useful or efficient. Issues exist in the accuracy of the data gained, but on a grander scale because of the increase in the number of inputs. In terms of efficiency, there are also issues in terms of computing and analyst time required to get though the raw data and explore the results to find the applicable truths.

2.3 Summary

All single dimensional software correlation techniques have in inherent weaknesses within the scalability of authorship attribution for malicious software. The state of the science needs to be provided a framework of multi-dimensional software correlation based on expert selection across a priorized sets of features. This can help spread the sources of attribution across areas of analysis in order to better apply augmented intelligence and allow analysts to navigate the maze of false positives at scale.

Chapter 3

Proposed Solution

This research proposes a novel approach to malware authorship attribution by identifying those features within a larger feature set of malicious software samples in such a way as to maximize accuracy. Specifically, we will input a given attributed malware dataset into an existing framework that performs static and dynamic malware analysis. The results of the analysis are applied to machine learning classifiers that result in all possible feature sets. We propose to determine the smallest subset of features that maximize accuracy.

3.1 Research Goals

The overarching goal of this research is to identify the minimum number of features required to attribute a malware sample to its author. The objectives are 1) to reduce the noise inherent with a large feature set, 2) to increase accuracy, and 3) to reduce the computation resources required to perform authorship attribution.

3.2 Research Plan

We plan to conduct this research by first extracting features from a repository of malware samples that are attributed and in common use by the cybersecurity research community, then selecting those features that provide the greatest accuracy for the least computational cost.

3.2.1 Feature Extraction

The feature extraction phase consists of selecting the attributed malware dataset, profiling each artifact in the dataset through static and dynamic analysis, and identifying indicators of malicious action.

Our background research uncovered three malware datasets that contain samples attributed to a specific author: Laurenza [Laurenza and Lazzeretti 2019], cyber-research [Radnus 2022b], and Haddadpajouh [Haddadpajouh et al. 2020]. The Laurenza dataset consists of 19 APT groups and over 2000 malware samples. The cyber-research dataset contains 3594 malware samples which are related to twelve APT groups and five different nation-states. The Haddadpajouh dataset consists of roughly 2000 samples and overlaps the cyber-research dataset. We propose using the cyber-research dataset as it contains the largest number of attributed samples. Table 3.1 overviews the cyber-research data with the number of malware samples attributed to APT Groups within respective countries.

Table 3.1: APT Malware Dataset

Country	**APT Group**	**Family**	**Sample Size**
China	APT 1		405
China	APT 10	i.a. Plugx	244
China	APT 19	Derusbi	32
China	APT 21	TravNet	106
Russia	APT 28	*Bears*	214
Russia	APT 29	*Dukes*	281
China	APT 30		164
North-Korea	DarkHotel	DarkHotel	273
Russia	Energetic Bear	Havex	132
USA	Equation Group	Fannyworm	395
Pakistan	Gorgon Group	Different RATs	961
China	Winnti		387
Total			3594

In researching malware analysis tools available that could be used to support the effort of preforming feature extraction of malicious samples, three products were investigated:

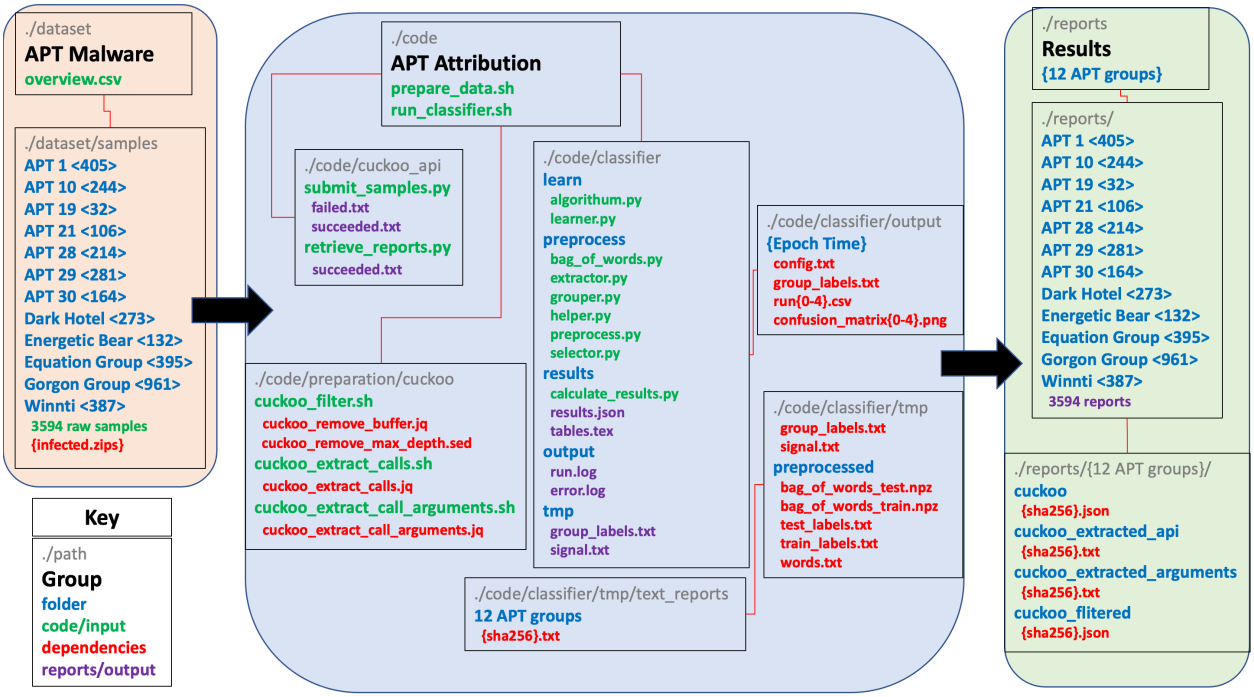

Figure 3.1: Feature extraction workflow [Radnus 2022a]

Cuckoo, Remnux, and Bro. Cuckoo is a open source platform that automates malicious file analysis for Windows, OS X, Linux, and Android binaries. Remnux is a Linux toolkit that assists analysts in reverse engineering malware samples. Bro is a network-based analysis framework. We propose using Cuckoo because it encompasses many of Remnux' and Bro's features, has an API that can be interfaced with for automation, and is open source.

The software code base that controls the Cuckoo sandbox interacts in the following ways: First, it manages the virtual machine image in which the malware is run. Second, it executes each malware sample in the sandboxed virtual machine environment. Third, it collects a variety of information from the sandbox, such as strings, PE sections, identification of packers, shell code, API calls, DLLs accessed, images of UI interaction, network connections, file manipulation, registry hives, and running processes. Fourth, the analysis data is captured in JSON-formatted files. Figure 3.1 provides a graphic description of the workflow as the dataset is ingested by the software code base to produce the reports.

In order to prepare the results to be analyzed, cleansing needs to be executed on the raw JSON report files for each sample. The cleansing consists of two steps: 1) removing encrypted data that is specific to the malware instance but not generally useful to characterizing the malware sample and 2) extracting API calls and arguments. Following the cleansing process, the json formatted reports are ready for feature pruning.

3.2.2 Feature Pruning

The feature pruning phase identifies the most meaningful combinations of features required for attribution. We propose doing this by using a combination of features identified by expert selection, then apply feature selection algorithm that improves attribution.

The identification and use of data modeling classification algorithms on the features will support data modeling classification as a structured method for validating the results. Additionally, the aim is to also to promote future analysis by adding additional inputs into this modular framework. In a comparison of data modeling techniques used in the research of applying attribution to threats the Random Forrest and Deep Neural Network Algorithms excelled [Gray et al. 2021]. Within the exploration of more than two classification algorithms, Random Forest (RF) was found to be the most suitable candidates for solving the problem due to their enhanced performance against the other techniques they tested [Hong et al. 2018]. These results agree with additional research in the field: [Hendrikse 2017], [Caliskan et al. 2015], [Kalgutkar et al. 2018], [Meng 2016], [Gonzalez et al. 2018]. The parameter values chosen for this data modeling classification algorithm is 100 estimators, meaning that 100 decision trees will run for this model. Within the exploration other data modeling techniques, Deep Neural Network's showed usefulness for classifying binaries to authors better than Support Vector Machine (SVM) and Conditional Random Fields (CRFs): [Meng and Miller 2018], [Rosenberg et al. 2017], [Rosenberg et al. 2018], [Alrabaee et al. 2019a], [Alrabaee et al. 2019b]. The parameters values chosen for this data modeling classification algorithm is 7 layers of that range from 2000 to 500 nodes, that have a max of 250 iterations through

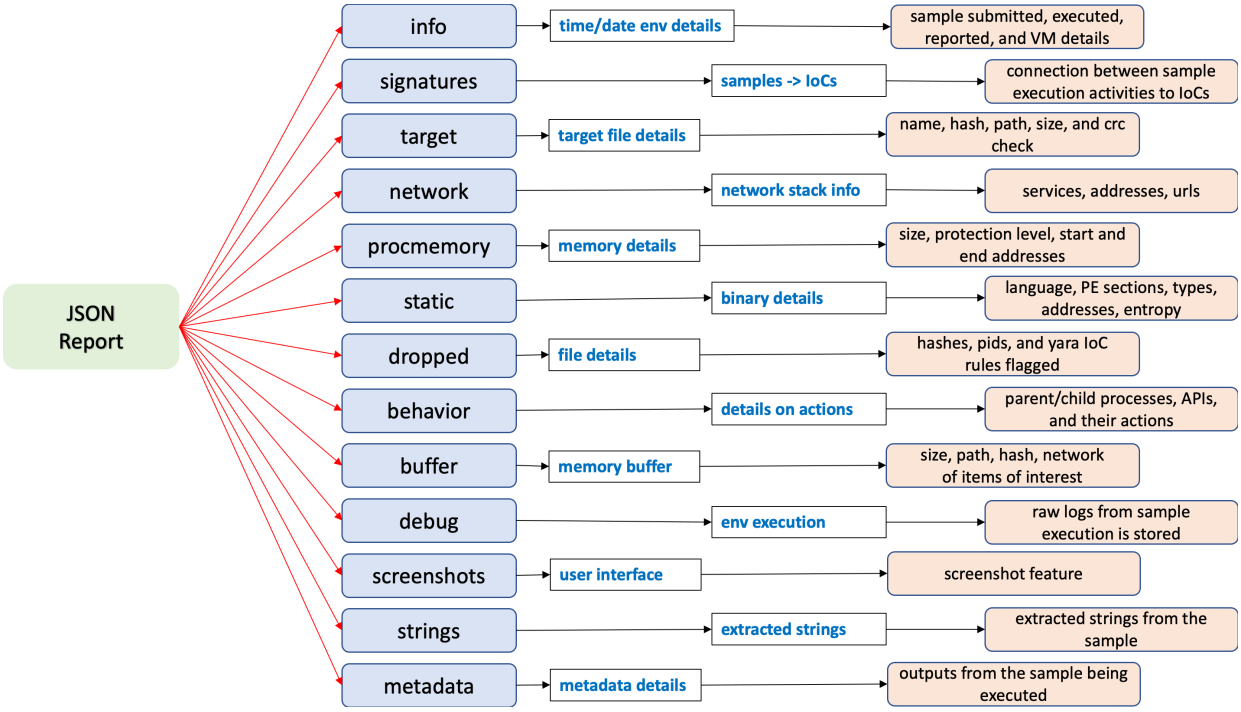

Figure 3.2: JSON Structure

which the data will be run. These two data modeling classification algorithms show promise in assisting in this research. Additionally, using more than one data modeling classification algorithm helps to fight potential bias present in the algorithms.

Once the data modeling classification algorithms are selected they will be executed against all filtered and extracted results. These results are collected that analyzed the accuracy of the classifiers correctly identifying the correct APT group when all features are present. This give us a control state in which we can experiment using various combinations of feature selection. Then we will prune features in order to identify the best combination that provided a more efficient solution than currently recognized methods. This method of data pruning is done though command line jq scripts on the .json formatted results. An example of this pruning would be iterate over all json formatted results file to separate the data for the WriteProcessMemory API using this piped query string: .behavior.processes[].calls[] — select(.api == "WriteProcessMemory").api.

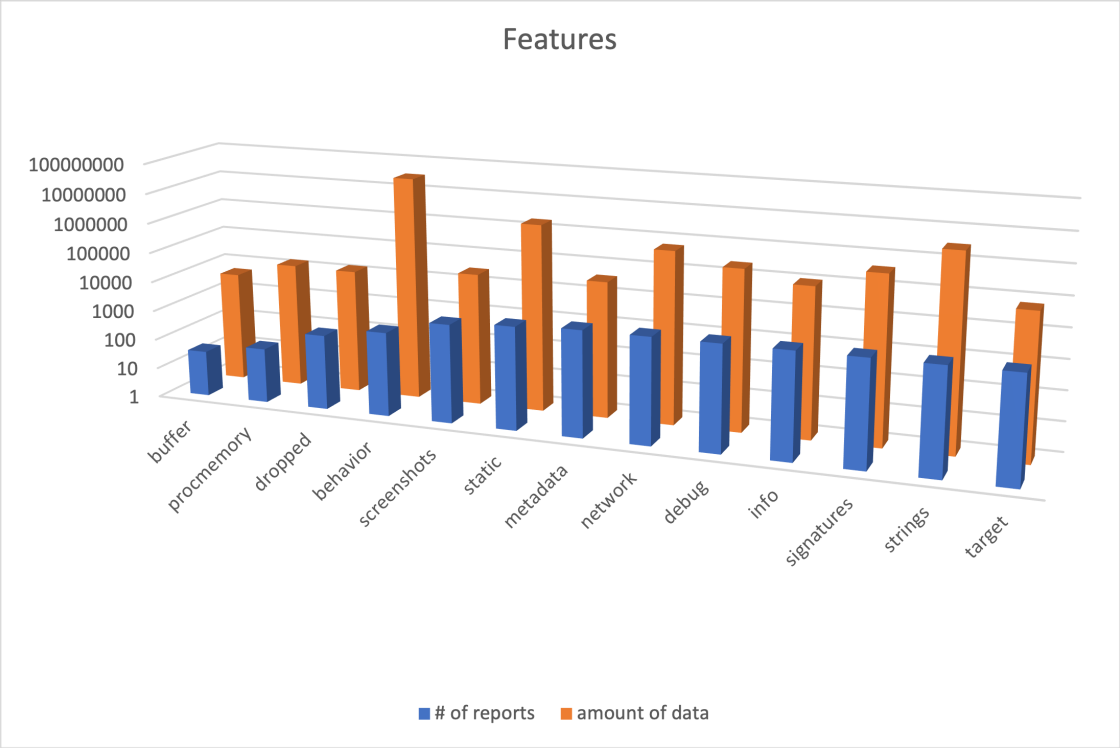

Figure 3.3: Features

The ontology of these results is structured into 13 areas described in Figure 3.2. The info feature contains information on the time/date that the sample submitted, executed, and reported on. Also, it provides details on Virtual Machine that sample was run on. Signatures provides a connection between sample execution activities to potential Indicators of Compromise (IoC). The target group of features gives target file details, like: name, hash, path, size, and crc checks. The collection of network information that emanated from the execution of the sample can be viewed in the network feature. Procmemory shows the details about different regions of the memory with their size, protection level, start and end addresses. The static feature displays information of the binary and its parts. This information includes timestamp of when compiled, type of binary, resources used, language, PE sections, PE sections types, memory addresses, and level of entropy. The dropped group of features provides information on dropped files, hashes, pids, and yara IoC rules flagged. The behavior group collects information on parent/child processes, their relationships, APIs

used, and their actions. Information on the size, path, hash, network of items of interest in volatile memory is contained within the buffer group of features. Debug is where raw logs from sample execution is stored. Information about the user interface at different points during sample run can be referenced in the screenshot feature. Strings contains extracted string from the sample. Metadata includes information about various outputs from the sample being executed. The scope of these json results include more than 38 million items, 1.35 million being unique, and an upper bound of 2.2 million instances. The frequency of the total amount of reports and data point present in these reports for these thirteen different feature groups is displayed in Figure 3.3.

Expert selection will be used as the starting point of iteration of features in order to apply attribution. This expert selection contains two steps, first step is to identify unique features that assist in attribution, second step is to eliminated duplicate data from the reports that will be given to the classifiers. Based on the ontology of the data contained within the reports there are 6 different areas that expert selection points to in Cyber Threat Intelligence reports. Two primary references are used in building this expert selection, the source column of the overview.csv file contained within the dataset selected for this research and additional supplemental reporting. Most of the supplemental reporting has been collected from publicly-available papers and blogs of malicious campaigns/activity/software that have been associated with vendor-defined APT that are sorted by year and their source details stored on github. [Bandla and Castro 2022]

APT groups 19, 21, Gorgon, and Winnti all have uniquely identify items contained within the filepath area that support attribution based on expert opinion in CTI reporting [CrowdStrike 2013], [Huss 2016], [Perigaud 2015], [Cherian 2013]. APT groups 1, 28, and 29 have information contained within function name arguments and strings that aid in applying attribution [Mandiant 2013], [GReAT 2018], [Artturi 2015]. APT 10 and the Energetic Bear groups are identifiable by looking for references to specific regkey read attempts [PWC 2017], [Kaspersky 2014a]. Operations by APT 30 can be identified thought the use of mutant name

argument [FireEye 2015]. The APT group known as Dark Hotel can be identified thought the used of a specific process name [Kaspersky 2014b]. Equation Group can be identified though the use of an identifiable string [Mila 2015]. The 6 areas for expert selection are using these combined sources are: function name, regkey read, filepath, mutant name, process name, and strings.

Iterations of different combinations of features will be generated in order to find the most effective way to apply attribution. Confusion matrices generated from the data classification algorithms will be used to compare accuracy and drive the iteration process of identifying features that have the greatest rate of positive change for attribution accuracy. The Proof of Concept section describes a specific scenario of feature pruning, analysis, and results. Algorithm 1 is a pseudocode description of a novel methodology for combining Expert Selection with Genetic Algorithms (ESGA) for feature selection.

The pseudocode in Algorithm 1 starts with the initialization of each individual data point in the feature collection. Then the correlation coefficients between expert selected features and the contribution of each feature in all feature sets are calculated to obtain the improved initial value of the individual and the feature collection, as defined by lines 7 and 8. The amount of correlation between these two parts is configured to 95% similarity in this version of the algorithm. The search process will stop and output the improved position of the feature collection, if the process comes to a stopping condition; otherwise, it will execute circularly. The repeat conditions is defined based on control runs with all feature sets preset and configured to stop when the a decrease in accuracy exceeds the maximum allowable threshold. This threshold is configured to be 3 consecutive instances of a greater than 1% decrease in accuracy. Lines 10–21 define the detailed cyclic process to get the improved feature subset, which contains updating the individual data points' velocity and position, calculating the fitness value of the individual data point using classifiers, and updating the historical best position of the individual data point and the improved position of the feature

Algorithm 1 Pseudocode of correlation-based ESGA method for feature selection

Input F: feature, where feature = [data, velocity, dimension, weight]
Input E: Let E_{expert} = e : F — human_approved(e)
Extract keywords from E_{expert}
E_0 = Use keywords to filter F
Let E_{new} = deduplicate(E_0)
corr_matrix = corr(F,E_{new})
Select Select corr_matrix.where(corr_matrix.shape), k=1(bool)
E_{new} = Drop column for column in corr_matrix.columns for any column > 0.95
E_{old} = null
while notImproving(E_{old},E_{new}): **do**
    Repeat notImproving $i = 0$ 3 where population.accuracy -1%
    Let $confusionMatrix_1$ = classifier(E_{new})
    Let $E_{changed}$ = geneticize(E_{new}, F)
    feature_optimization_matrix = best population.accuracy(individual.X, individual.Y) from (E_{new}, F)
    Apply GA to k=1(feature)
    Let $E_{changed}$ Select feature_optimization_martix.where(population.accuracy is greatest)
    Let $confusionMatrix_2$ = classifier($E_{changed}$)
    confusionMatrix = best($confusionMatrix_1$, $confusionMatrix_2$)
    $E_{old} = E_{new}$
    E_{new} = best(E_{new}, $E_{changed}$)
end while
Output E_{new}

collection. The most improved feature subset will be selected after finishing this cyclical process.

3.3 Proof of Concept

The sandboxed virtual machine environment is comprised of a Intel i9 CPU fitted into a Socket LGA 2066 motherboard with 32 GigaBytes of DDR4 RAM. The RAM is configured to run at a speed of 3600 Megahertz. Two EVGA GeForce GTX 1080 Ti FTW3 HYBRID cooled GPUs are connected together using an SLI bridge to provide video output and computational support using cuda. Additionally, two 250 GigaByte Samasung 850 Pro Solid State Drives connected in a RAID 0 are formated for the boot partition and storage of the raw results. To illustrate the feasibility of our approach, we implemented Algorithm 1 on a limited subset of data. We started by ingesting the cyber-research [Radnus 2022b] dataset into the Cuckoo Sandbox software tool. The tool required roughly 700 hours to analyze the 3594 samples of the dataset and produce the raw results described in Figure 3.2. We applied two classification algorithms – Random Forrest Classifier (RFC) and Deep Neural Net (DNN) classifier – to the raw results to establish an analysis baseline that takes into consideration the entire dataset. We also applied the classifier algorithms to a subset of the raw results so as to demonstrate how to select and measure combinations of features that might reduce the noise of the entire data set and lead to better performance. We refer to the former analysis as the "control results" and the latter analysis as the "extracted results."

Table 3.2 shows the accuracy of control results obtained from the RFC and DNN classification algorithms. These control results are used to describe the accuracy of applying attribution using all available features from the dataset results. These results include features from static/dynamic software analysis, volatile memory, and network actions. These features are used by the classifiers to apply attribution across the twelve identified Advanced Persistent Threats in table 3.1. There are 3 specific group comparisons that are preformed in this analysis. The first, APTGrouper is focused on grouping attributed samples from

specific APTs. The second is focused on grouping attributed samples from specific countries. The third is a combination of countries separated along with grouping based on the malware family. Both RFC and DNN are applied to each of these groupers to measure the accuracy of applying attribution within these groups. The Dataset column describes the source of the results. Within the Dataset column, the "Cuckoo" label designates all filtered results were used and "Cuckoo*" designates the pruned results were used. The next three columns represent accuracy results and the standard deviation of those results described in the form of a lowercase Sigma. In order to adjust for bias within the Groupers Unbalanced, Undersampling, and Oversampling accuracy are recorded and displayed. The problem of having imperfect or unbalanced data will always be present, but can be measured to provide guidance in the correct direction. For the chosen dataset there is a clear imbalance between the Country Group China having 1338 samples out of the 3594 total samples in table 3.1. These results amount to around 37% of the total samples within the five different Countries of this group. This matters because building a classifier using the data as it is, would in most cases give us a prediction model that is always biased towards the Chine Country Group class. The classifier used would be biased without preprocessing. There are three things to do when dealing with imbalanced data:

- Ignoring the problem.
  - Building a classifier using the data as it is, would in most cases give us a prediction model that always returns the majority class. The classifier would be biased and the world is not a better place.
- Undersampling the majority class.
  - One of the most common and simplest strategies to handle imbalanced data is to undersample the majority class. While different techniques have been proposed in the past, typically using more advanced methods (e.g. undersampling specific samples, for examples the ones "further away from the decision boundary")

[Japkowicz 2000] did not bring any improvement with respect to simply selecting samples at random. For this analysis we selected n samples at random from the majority class, where n is the number of samples for the minority class, and use them during training phase. These samples will be excluded for use in validation.

- Oversampling the minority class.
  - Oversampling the minority class can result in overfitting problems if we oversample before cross-validating. A currently well recognized method for this is the Synthetic Minority Oversampling Technique (SMOTE) technique [Chawla et al. 2002]. SMOTE is a method that instead of simply duplicating entries creates entries that are interpolations of the minority class, as well as under samples the majority class. Normally when we duplicate data points the classifiers get very convinced about a specific data point with small boundaries around it, as the only point where the minority class is valid, instead of generalizing from it. SMOTE however effectively forces the decision region of the minority class to become more general, partially solving the generalization problem. In simplest terms, oversampling using SMOTE improves the decision boundaries in imbalanced data.

Multiple controls runs were executed and results were within 1.5% accuracy and 0.02 standard deviation.

The second part of the proof of concept was the application of feature pruning using expert selection. Expert selection was preformed on the collected control results of the most maliciously scored samples collected. There were a total of 17 malware sample reports that fell into the highest threat level range of 7-10 and were categorized as very malicious. Within these 17 high scoring results, 5 distinguishing types of feature sets were contained within these samples. These feature sets are categorized as: APIs, arguments, PE sections, entropy, and file details. In total, there were 167 API calls in the 17 samples that were captured. There were 17 instances of the API WriteProcessMemory being used in reports for this

Table 3.2: Control Results

APTGrouper

	Dataset	Unbalanced	Undersampling	Oversampling
RFC	Cuckoo	0.40 (σ: 0.01)	0.17 (σ: 0.03)	0.22 (σ: 0.02)
DNN	Cuckoo	0.40 (σ: 0.01)	0.13 (σ: 0.07)	0.21 (σ: 0.03)

CountryGrouper

	Dataset	Unbalanced	Undersampling	Oversampling
RFC	Cuckoo	0.44 (σ: 0.01)	0.28 (σ: 0.01)	0.28 (σ: 0.01)
DNN	Cuckoo	0.44 (σ: 0.00)	0.27 (σ: 0.01)	0.28 (σ: 0.01)

CountrySeparatedGroupAndFamiliesGrouper

	Dataset	Unbalanced	Undersampling	Oversampling
RFC	Cuckoo	0.67 (σ: 0.01)	0.35 (σ: 0.01)	0.34 (σ: 0.01)
DNN	Cuckoo	0.67 (σ: 0.01)	0.34 (σ: 0.02)	0.33 (σ: 0.01)

threat level. Data pruning was executed to isolate on the API WriteProcessMemory calls and associated API arguments from this round of classification.

Table 3.3 describes the pruned analysis results that show the rate of change in attribution based on two changes to the API WriteProcessMemory pruned results. The accuracy and standard deviation calculations in red have shown the change in applying attribution using only the API WriteProcessMemory calls and associated API arguments.

Table 3.3: API WriteProcessMemory Extracted Results

APTGrouper

	Dataset	Unbalanced	Undersampling	Oversampling
RFC	Cuckoo*	0.28 (σ: 0.01)	0.16 (σ: 0.09)	0.05 (σ: 0.01)
DNN	Cuckoo*	0.27 (σ: 0.00)	0.07 (σ: 0.00)	0.08 (σ: 0.03)

CountryGrouper

	Dataset	Unbalanced	Undersampling	Oversampling
RFC	Cuckoo*	0.38 (σ: 0.00)	0.17 (σ: 0.06)	0.15 (σ: 0.03)
DNN	Cuckoo*	0.37 (σ: 0.00)	0.21 (σ: 0.10)	0.17 (σ: 0.03)

CountrySeparatedGroupAndFamiliesGrouper

	Dataset	Unbalanced	Undersampling	Oversampling
RFC	Cuckoo*	0.70 (σ: 0.00)	0.30 (σ: 0.00)	0.30 (σ: 0.00)
DNN	Cuckoo*	0.70 (σ: 0.00)	0.38 (σ: 0.16)	0.46 (σ: 0.19)

The accuracy results from the API WriteProcessMemory Extracted Results show an increase in accuracy for the unbalanced measurement of the Country Separated Group And

Families Grouper highlighted in green. Additionally, there are significant changes in accuracy results for the both classifiers executed in the APTGrouper and CountryGrouper runs. The average rate of change between the positive accuracy of the Control and API WriteProcessMemory Extracted Results with both classifiers are within 8-11% for the APT and Country Groupers. Significant change was also observed in standard deviation within the extracted results for Undersampling in all groups, except for Random Forrest in the Country Separated Group And Families Grouper. Additionally, the largest difference in standard deviation was found in Oversampling of the Deep Neural Network Classification in the Country Separated Group And Families Grouper. These results lead to the conclusion that the API WriteProcessMemory features have a strong connection to apply attribution within this dataset, especially in the Country Separated Group And Families Grouper. The research plan section describes the plan to extending this research in order to show the efficiency of applying attribution using expert selection on additional extracted data input. Measurements within these results of accuracy between the control and pruned results will be the validation mechanism for this research.

3.4 Research Hypothesis

The research hypothesis is "the expert selection correlation mapping technique of feature selection using genetic algorithms is be more efficient than current recognized methods in applying cyber threat attribution". The null hypothesis of this research is "the expert selection correlation mapping technique of feature selection using genetic algorithms is as efficient or less efficient than current recognized methods in applying cyber threat attribution". More and less efficient is defined as recent applications of technology in the realm of authorship attribution methods that have an accuracy of 88%.

3.5 Research Plan

Using the same dataset, results, and software code base the research will follow the same path a described in the proof of concept section. A control run of random features will be used to create the control results for comparison. Then features identified by expert selection will be chosen by using cyber threat intelligence reports for APTs in the chosen dataset. Key words, phrases, and numbers will be picked from the reports to be used in selecting features. Second, data de-duplication will be done between the features selected and additional features available from the dataset reports. A comparison function will be used to find duplicate data that is similar within the data in selected features. A configurable variable of 95% is planned to be used as the starting point of this research, but can be modified to meet the needs of other research efforts. The reason for this is to reduce unneeded duplication of data that will not assist with the task of accuracy of authorship attribution. The third part is the initialization and recurrence of measurements for cyber threat attribution accuracy based on a genetic algorithm being applied to the de-duplicated data. The genetic algorithms that are planned for comparison and correlation of expert selection are: Genetic & Evolutionary Feature Selection (GEFeS), Particle Swarm Optimization (PSO), Artificial Bee Colony Optimization (ABCO), Ant System Optimization (ASO), and Glowworm Swarm Optimization (GSO). In the same way that accuracy was measured in the proof of concept, accuracy and standard deviation will be the primary measurements of the research.

A configurable repeat until clause stop the recursion based on measurements gain from multiple control runs on all features being available. For this research this clause is configured to trigger when 3 consecutive runs happen in which a -1% accuracy is observed though the application of the genetic algorithm. The last part is the building and reporting of a confusion matrix in order to graphically display the results to humans and allow for further research to be developed and tested.

Results will be captured in three independent groupers: Named APT Groups, Country based Groups, and the combination of Country Separated Groups and Malware Family

Groups. Named APT Groups is the broadest group because it includes indirect state-sponsored threat actors, like intermediate level threat actors aimed at financial and educational institutions. Country Separated Groups are further restricted to state-sponsored threat actors that are directly attributable to a country and commonly attack infrastructure based on a military, economic, or geopolitical goals. The Country Separated Malware Family Groups further restricts classification down to the differences between the indirectly and directly sponsored, along with the differences in unique identifiers based on samples obtained.

The other guiding force to measure the results will be to calculate the cost of the genetic algorithms' assistance in feature selection within the 13 different date types to be compared. These features and the frequency of the data contained within this dataset are detailed in Figure 3.3, titled Features. With the cost results captured, statistical analysis on the dissimilar features included in this correlation will be scaled to larger datasets in order to find a use-able balance between expert selection and genetic algorithm feature enumeration of dissimilar results.

The hypothesis will be validated using the correlation-based ESGA method for feature selection algorithm described in the feature pruning section. This research will be validated by measuring the accuracy of attribution within the chosen feature sets. The environment for this research will be a combination of three main items: a dataset of attributed malware, an automated analytical platform, and algorithms for classification. The dataset includes 3594 samples from 12 different APTs. [Radnus 2022b] These samples were chosen because they are from different APTs supporting different countries and contain a variety of unique information that can be used to test the efficiency of applying cyber threat attribution. Details on the samples and traceability to cyber threat intelligence report applying attribution is contained within the overview.csv file within the root directory of the dataset. The automated analytical platform is a malware sandbox and software codebase to manage actions.

Examples of actions managed by the codebase is interactions with the sandbox api to submit and collect samples, pruning of results, execution of NLP, and reporting of results.

The timeline for research defense, code changes, data collection, analysis, and reporting on this research is planned for in the following timeline.

- Defend Research Proposal, April 2023
- Implement Code Changes for Research, May 2023
- Collect and Verify Data, June 2023
- Document Results, July 2023
- Defend Research, August, 2022

Bibliography

[Abuhamad et al. 2018] Mohammed Abuhamad, Tamer AbuHmed, Aziz Mohaisen, and DaeHun Nyang. 2018. Large-scale and language-oblivious code authorship identification. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 101–114.

[Alrabaee et al. 2019a] Saed Alrabaee, Mourad Debbabi, and Lingyu Wang. 2019a. On the feasibility of binary authorship characterization. *Digital Investigation* 28 (2019), S3–S11.

[Alrabaee et al. 2019b] Saed Alrabaee, ElMouatez Billah Karbab, Lingyu Wang, and Mourad Debbabi. 2019b. BinEye: towards efficient binary authorship characterization using deep learning. In *European Symposium on Research in Computer Security*. Springer, 47–67.

[Artturi 2015] Artturi. 2015. Duke APT group's latest tools: cloud services and Linux support. (2015). `https://www.f-secure.com/weblog/archives/00002822.html`

[Bakdash et al. 2017] Jonathan Z. Bakdash, Steve Hutchinson, Erin G. Zaroukian, Laura R. Marusich, Saravanan Thirumuruganathan, Char Sample, Blaine Hoffman, and Gautam Das. 2017. Malware in the Future? Forecasting Analyst Detection of Cyber Events. *CoRR* abs/1707.03243 (2017). `http://arxiv.org/abs/1707.03243`

[Bandla and Castro 2022] K. Bandla and S. Castro. 2022. APT CTI Reports. (2022). `https://github.com/aptnotes/data`

[Bird et al. 2015] Christian Bird, Tim Menzies, and Thomas Zimmermann. 2015. *The Art and Science of Analyzing Software Data*. Elsevier.

[Brassard] Ana Brassard. The Morris Worm (1988). (????).

[Caliskan et al. 2015] Aylin Caliskan, Fabian Yamaguchi, Edwin Dauber, Richard Harang, Konrad Rieck, Rachel Greenstadt, and Arvind Narayanan. 2015. When coding style survives compilation: De-anonymizing programmers from executable binaries. *arXiv preprint arXiv:1512.08546* (2015).

[Caliskan-Islam et al. 2015] Aylin Caliskan-Islam, Richard Harang, Andrew Liu, Arvind Narayanan, Clare Voss, Fabian Yamaguchi, and Rachel Greenstadt. 2015. De-anonymizing programmers via code stylometry. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*. 255–270.

[Chawla et al. 2002] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[Cherian 2013] R. Murthy A. Cherian. 2013. Inside Report – APT Attacks on Indian Cyber Space. (2013). https://airbus-cyber-security.com/newcomers-derusbi-family/

[Christodorescu et al. 2005] Mihai Christodorescu, Somesh Jha, Sanjit A Seshia, Dawn Song, and R andal E Bryant. 2005. Semantics-aware malware detection. In *2005 IEEE Symposium on Security and Privacy (S&P'05)*. IEEE, 32–46.

[Clarke 2000] Richard A. Clarke. 2000. National Plan for Information Systems Protection. (2000). https://irp.fas.org/offdocs/pdd/CIP-plan.pdf

[CrowdStrike 2013] CrowdStrike. 2013. Deep Panda. (2013). http://cybercampaigns.net/wp-content/uploads/2013/06/Deep-Panda.pdf

[Cyware Hacker News 2016] Cyware Hacker News. 2016. Remembering Operation Titan Rain. (2016). https://cyware.com/news/remembering-operation-titan-rain-c54ad3e4

[Davison 2011] Andrew P. Davison. 2011. Provenance tracking — Best practices for data management. (2011). http://rrcns.readthedocs.io/en/latest/provenance_tracking.html

[Davison 2014] Andrew P. Davison. 2014. Provenance Inference in Software. (2014). https://insights.sei.cmu.edu/sei_blog/2014/02/provenance-inference-in-software.html

[Delibasis Fokas 2007] Demetrius Delibasis Fokas. 2007. Modern Information Warfare Operations and the Weakness of the Current Regulatory Regime. *Philosophy of International Law (https://www.electronicpublications.org/catalogue.php* (2007).

[Department of Defense 2011] Department of Defense. 2011. Department of Defense Strategy for Operating in Cyberspace. (2011).

[Eichensehr 2020] Kristen Eichensehr. 2020. The Law & Politics of Cyberattack Attribution. *UCLA Law Review* 67 (2020).

[Fedotova et al. 2021] Anastasia Fedotova, Aleksandr Romanov, Anna Kurtukova, and Alexander Shelupanov. 2021. Authorship attribution of social media and literary Russian-language texts using machine learning methods and feature selection. *Future Internet* 14, 1 (2021), 4.

[FireEye 2015] FireEye. 2015. APT30IoCs. (2015). https://github.com/fireeye/iocs/blob/master/APT30/eeffc8e8-caee-4fe1-8ace-7a994b5d893f.ioc

[Flederer et al. 2017] Frank Flederer, Ludwig Ostermayer, Dietmar Seipel, and Sergio Montenegro. 2017. Source Code Verification for Embedded Systems using Prolog. *arXiv preprint arXiv:1701.00630* (2017).

[Frantzeskou et al. 2006] Georgia Frantzeskou, Efstathios Stamatatos, Stefanos Gritzalis, and Sokratis Katsikas. 2006. Source code author identification based on n-gram author profiles. *Artificial Intelligence Applications and Innovations* (2006), 508–515.

[Goldsmith 2022] Jack Goldsmith. 2022. *The United States' Defend Forward Cyber Strategy: A Comprehensive Legal Assessment*. Oxford University Press.

[Gonzalez et al. 2018] Hugo Gonzalez, Natalia Stakhanova, and Ali A Ghorbani. 2018. Authorship attribution of android apps. In *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*. 277–286.

[Gray et al. 2021] Jason Gray, Daniele Sgandurra, and Lorenzo Cavallaro. 2021. Identifying authorship style in malicious binaries: techniques, challenges & datasets. *arXiv preprint arXiv:2101.06124* (2021).

[GReAT 2018] GReAT. 2018. A Slice of 2017 Sofacy Activity. (2018). `https://securelist.com/a-slice-of-2017-sofacy-activity/83930/`

[Greenberg 2018] Andy Greenberg. 2018. The untold story of NotPetya, the most devastating cyberattack in history. *Wired, August* 22 (2018).

[Guo et al. 2017] Jin Guo, Jinghui Cheng, and Jane Cleland-Huang. 2017. Semantically enhanced software traceability using deep learning techniques. In *Proceedings of the 39th International Conference on Software Engineering*. IEEE Press, 3–14.

[Gupta et al. 2009] Archit Gupta, Pavan Kuppili, Aditya Akella, and Paul Barford. 2009. An empirical study of malware evolution. In *Communication Systems and Networks and Workshops, 2009. COMSNETS 2009. First International*. IEEE, 1–10.

[Haddadpajouh et al. 2020] Hamed Haddadpajouh, Amin Azmoodeh, Ali Dehghantanha, and Reza M Parizi. 2020. MVFCC: A multi-view fuzzy consensus clustering model for malware threat attribution. *IEEE Access* 8 (2020), 139188–139198.

[Hendrikse 2017] Steven Hendrikse. 2017. *The effect of code obfuscation on authorship attribution of binary computer files*. Ph.D. Dissertation. Nova Southeastern University.

[Hong et al. 2018] Jiwon Hong, Sanghyun Park, Sang-Wook Kim, Dongphil Kim, and Wonho Kim. 2018. Classifying malwares for identification of author groups. *Concurrency and Computation: Practice and Experience* 30, 3 (2018), e4197.

[Huss 2016] D. Huss. 2016. Exploring Bergard: Old Malware with New Tricks. (2016). `https://proofpoint.com/us/exploring-bergard-old-malware-new-tricks`

[Jadalla and Elnagar 2008] Ameera Jadalla and Ashraf Elnagar. 2008. PDE4Java: Plagiarism Detection Engine For Java, Source Code: A Clustering Approach. *IJBIDM* 3, 2 (2008), 121–135.

[Japkowicz 2000] Nathalie Japkowicz. 2000. The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on artificial intelligence*, Vol. 56. 111–117.

[Kalgutkar et al. 2018] Vaibhavi Kalgutkar, Natalia Stakhanova, Paul Cook, and Alina Matyukhina. 2018. Android authorship attribution through string analysis. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*. 1–10.

[Kamiński et al. 2020] Mariusz Antoni Kamiński and others. 2020. Operation "Olympic Games." Cyber-sabotage as a tool of American intelligence aimed at counteracting the development of Iran's nuclear programme. *Security and Defence Quarterly* 29, 2 (2020), 63–71.

[Kaspersky 2014a] Kaspersky. 2014a. Crouching Yeti — Appendixes. (2014). `https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/08080840/Kaspersky_Lab_crouching_yeti_appendixes_eng_final.pdf`

[Kaspersky 2014b] Kaspersky. 2014b. DarkHotelIoCs. (2014). `https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/08070901/darkhotelappendixindicators_kl.pdf`

[Kustanto and Liem 2009] Cynthia Kustanto and Inggriani Liem. 2009. Automatic source code plagiarism detection. In *2009 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing*. IEEE, 481–486.

[Laurenza and Lazzeretti 2019] Giuseppe Laurenza and Riccardo Lazzeretti. 2019. dAPTaset: A comprehensive mapping of APT-related data. In *Computer Security*. Springer, 217–225.

[Lesner et al. 2010] Boris Lesner, Romain Brixtel, Cyril Bazin, and Guillaume Bagan. 2010. A novel framework to detect source code plagiarism: now, students have to work for real!. In *Proceedings of the 2010 ACM Symposium on Applied Computing*. 57–58.

[Lewis 2005] James A Lewis. 2005. Computer Espionage, Titan Rain and China. *Center for Strategic and International Studies-Technology and Public Policy Program* 1 (2005).

[Li et al. 2017] Qiang Li, Zeming Yang, Zhengwei Jiang, Baoxu Liu, and Yuxia Fu. 2017. Association Analysis Of Cyber-Attack Attribution Based On Threat Intelligence. In *2017 2nd Joint International Information Technology, Mechanical and Electronic Engineering Conference (JIMEC 2017)*. Atlantis Press.

[Lin 2016] Herbert Lin. 2016. Attribution of malicious cyber incidents: From soup to nuts. *Journal of International Affairs* 70, 1 (2016), 75–137.

[Lynn III 2010] William F Lynn III. 2010. Defending a new domain-the Pentagon's cyberstrategy. *Foreign Aff.* 89 (2010), 97.

[Mandiant 2013] Mandiant. 2013. APT1 Exposing One of China's Cyber Espionage Units. (2013). `https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf`

[Meng 2016] Xiaozhu Meng. 2016. Fine-grained binary code authorship identification. In *Proceedings of the 2016 24th ACM SIGSOFT international symposium on foundations of software engineering*. 1097–1099.

[Meng and Miller 2018] Xiaozhu Meng and Barton P Miller. 2018. Binary code multi-author identification in multi-toolchain scenarios. *Under Submission* (2018).

[Mila 2015] Mila. 2015. EquationSamples. (2015). `https://cantagiodump.blogspot.com/2015/02/equation-samples-from-kaspersky-report.html`

[Myers 2004] Richard Myers. 2004. National Military Strategy of the United States of America 2004. (2004).

[Nakashima 2012] Ellen Nakashima. 2012. Iran blamed for cyberattacks on US banks and companies. *Washington Post* 21 (2012).

[Oh 2009] Jeongwook Oh. 2009. Fight against 1-day exploits: Diffing binaries vs anti-diffing binaries. *Black Hat. Black Hat* (2009).

[Perigaud 2015] F. Perigaud. 2015. Newcomers in the Derusbi family. (2015). `https://airbus-cyber-security.com/newcomers-derusbi-family/`

[Purohit] Rohit Sharma1 Dr Mona Purohit. Cyber Attacks That Shook the World. (????).

[PWC 2017] PWC. 2017. Operation Cloud Hopper Indicators of Compromise. (2017). `https://www.pwc.co.uk/cyber-security/pdf/cloud-hopper-indicators-of-compromise-v3.pdf`

[Quiring et al. 2019] Erwin Quiring, Alwin Maier, and Konrad Rieck. 2019. Misleading Authorship Attribution of Source Code using Adversarial Learning. *arXiv preprint arXiv:1905.12386* (2019).

[2022a] Radnus. 2022a. APTAttribution. `https://github.com/Radnus/APTAttribution`. (2022).

[2022b] Radnus. 2022b. APTMalware. `https://github.com/Radnus/APTMalware`. (2022).

[Ranjan et al. 2016] Aditya Kaushal Ranjan, Raja Ali, Vijay Kumar, and Minoo Hosseinzadeh. 2016. Boolean Signatures for Metamorphic Malware. *Procedia Computer Science* 78 (2016), 255–262.

[Romanosky and Boudreaux 2019] Sasha Romanosky and Benjamin Boudreaux. 2019. Private Sector Attribution of Cyber Incidents. (2019).

[Rosenberg et al. 2017] Ishai Rosenberg, Guillaume Sicard, and Eli Omid David. 2017. DeepAPT: nation-state APT attribution using end-to-end deep neural networks. In *International Conference on Artificial Neural Networks*. Springer, 91–99.

[Rosenberg et al. 2018] Ishai Rosenberg, Guillaume Sicard, and Eli Omid David. 2018. End-to-end deep neural networks and transfer learning for automatic analysis of nation-state malware. *Entropy* 20, 5 (2018), 390.

[Saha et al. 2019] Sreya Saha, Md Mahbubul Hafiz, and Chanchal Kumar Roy. 2019. Integrating Requirements and Code Artifacts through Traceability Recovery using Bidirectional Gated Recurrent Units. In *Proceedings of the 27th IEEE International Requirements Engineering Conference (RE'19)*. IEEE, 87–98.

[Sartea et al. 2020] Riccardo Sartea, Georgios Chalkiadakis, Alessandro Farinelli, and Matteo Murari. 2020. Bayesian Active Malware Analysis. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. 1206–1214.

[U.S. Cyber Command 2018] U.S. Cyber Command. 2018. USCYBERCOM Vision. `https://www.cybercom.mil/Portals/56/Documents/USCYBERCOM%20Vision%20April%202018.pdf`. (2018). Accessed on 2023-03-23.

[Varela et al. 2011] Paulo Varela, Edson Justino, and Luiz S Oliveira. 2011. Selecting syntactic attributes for authorship attribution. In *The 2011 International Joint Conference on Neural Networks*. IEEE, 167–172.

[Woodard et al. 2007] Laura Woodard, Cynthia K Veitch, Sherry Reede Thomas, and David Patrick Duggan. 2007. *Categorizing threat: building and using a generic threat matrix.* Technical Report. Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA ....

[Zager and Zager 2016] Robert Zager and John Zager. 2016. A Response to? Cyber Proficient Force 2015 & Beyond?: Why We Will Continue to Lose the Cyber War. *Journal Article— Sep* 13, 1 (2016), 34am.

[Zeidman 2011] Bob Zeidman. 2011. *The Software IP Detective's Handbook: Measurement, Comparison, and Infringement Detection*. Prentice Hall Professional.

[Zeidman 2012] Robert Zeidman. 2012. Detecting copied computer source code by examining computer object code. (Aug. 28 2012). US Patent 8,255,885.

[Zetter 2014] Kim Zetter. 2014. *Countdown to zero day: Stuxnet and the launch of the world's first digital weapon*. Broadway Books.