Finger Pointing for Fun, Profit and War?

The importance of a technical attribution capability in an interconnected world.

Tom Parker (tom@rooted.net)

July 2010

Introduction

During the past two years, a number of incidents have given rise to significant amounts of speculation amongst industry, and media outlets as to the origins of many of the observed computer network based attacks today. For example, in no particular order of chronology or significance – these have included 2010 attacks against a number of large corporations (dubbed "Operation Aurora") the 2007 attacks against Estonian internet infrastructures[1], the 2008 South Ossetian government attacks[2] and GhostNet[3]; which purportedly compromised classified documents from both governmental and private organizations – namely the defense industrial base (DIB).

Many of these attacks have followed the same modus operandi of a majority of attacks observed since the inception of computer networks; reconnaissance, compromise; and finally to establish a persistent presence within the target environment. In spite of this, due to the high profile nature of many targets involved, many have developed a fascination with the origins of the attacks and issued bold statements, pointing fingers at nation states and organized crime rings amongst others.

In February of 1998, Department of Defense computers were attacked with a well known operating system vulnerability. The attacks originated from networks in Germany, Israel and the UAE. At around this same time, the United States and its allies were preparing for possible military action in Iraq, due to disputes over UN weapons inspections. It is without doubt that if this were to happen today, with a sharp jerk of the knee, the wires would be ablaze with speculation over nation state involvement and perhaps even the preemptive suggestion of a nuclear response from an ill informed politician. This incident was later attributed to an Israeli national and two California residents with no known ties to state sponsored programs.

Solar Sunrise (as this incident is now referred to) was technically inferior in contrast to many of today's standards. However, it serves to demonstrate the need for restraint when attempting to attribute an attack and the possible dangers of incorrect attribution – even more so during times when tensions with a foreign nation state are high.

While unlikely to ever be the soul motivation behind a declaration of war, it is a very real possibility that at some point in the near future a decision will be made to take military action based upon a number of factors. These include the technical analysis of a cyber incident and attribution of the actor(s) responsible.

Because of this fact alone, as an industry we need to become more accountable for our analysis of technical threats; and speculation alone can not be allowed to form public opinion of, or vilify an individual state without due cause and solid technical fact.

The Cyber Conflict Lexicon

One of the many issues we face today in the information security world is the formation of a universal community lexicon. Terms such as vulnerability, exploit and scan, are frequently confused and often lead to miscommunication and result in incorrect decisions being made. The cyber conflict space is no exception and terms such as cyber war are frequently bounded around with infrequent regard for their true meaning. An entire thesis could be written on the security lexicon, so for the purposes of this paper, we will focus on just one. The word "cyber war" is often used by media and industry to describe the actions of a broad range of adversarial activity – from organized crime groups, non-state sponsored sympathizers of state regimes to bored hackers waiting for World of War craft to update. To the many who have studied the topic of cyber conflict for some time, this is analogous to calling street crime "land war." While arguably technically accurate, it has connotations which many will find misleading. War is commonly defined as "a legal state created by a declaration of war and ended by official declaration during which the international rules of war apply." While some might find this is a literal translation in order to communicate with a broad cross section of the information security community, the lexicon must be compatible with existing terminology. For the purposes of the remainder of this paper, cyber war will mean just that: a declared state of warfare between two states, during which time the use of computer network attack is employed.

Attribution and Why We Care

Attribution is often viewed as a futile task, and challenged with a policy of positioning oneself with a generic cyber defense posture designed to fend off all threats regardless of origin. Although such a generic defensive posture may be plausible in certain environments it does not address the demands of justifying a measured response, or the ever growing demands of public opinion. The ability to attribute is also a crucial tool in the belt of many intelligence organizations who seek to profile the capabilities of noteworthy current and potential future adversaries and even track the interrelations between these groups. Further to this, the ability to attribute an attack is vital for successful law enforcement efforts, and the development of deterrents. In both the law enforcement, and intelligence worlds attribution also provides insights which allow the tracking of the capability supply chain. Associating field craft, tool markings and other attack meta data between attacks may often provide key information into how malicious software, and other attack technologies are shared, traded and co-developed between different actor groups.

When speaking of attribution, it is most often understood to mean attributing an attack at the individual level providing information regarding who they are, where they are located and other personal data. While this may often be an objective for law enforcement

attribution efforts, attribution analysis can take place at higher levels which may prove more useful as detailed data about the individual behind an attack. As a case in point, detailed data regarding the tool set utilized by a state funded actor group, including supply chain data, correlation between multiple attacks and the technical aptitude of the tool sets authors will in many cases be of more use for defensive purposes, than knowing the favorite color of one of the actors within that group. And so – when we talk about attribution, what we really mean is defining a set of characteristics, associated with an individual or group of individuals with one or more commonalities, such as the actors sharing common geography, an activist group, employment, or even loose associations with other actors of interest.

Conventional Analysis Data Sources

Today's attribution efforts are typically conducted, through leveraging a small handful of data sources including: runtime, and static analysis of code; forensic analysis of the run-time state of a compromised system (including memory and file system forensics); and the analysis of attack vectors, such as vulnerability triggers and exploit payloads. These methods, while often effective often involve many hours of pain staking research and analysis – often to discover that the subject is of little interest – and identical to other, recently analyzed targets.

There will always be a need for manual analysis efforts, however there is a clear need for those charged with responding to attacks to be able to quickly ascertain whether a piece of malicious software is of note and therefore worth investing more time to analyze or if it can be connected to an existing body of research. Such a model, also lends itself well to a collaborative environment involving large numbers of analysts, possibly across multiple organizations, that are required to share attack data with one another, in a normalized, and timely manner.

A common problem with many analysis tools and techniques today is that they are simply not designed for purposes of attribution.

Existing Automated Analysis Technologies

There are two primary groups of widely available automated analysis software today, which each represent a significant amount research dollars, but does not address the problem of attribution. Anti-virus technologies have been around for a long time, and to date, primarily rely upon a signature based approach in order to identify the presence of a threat. While all modern anti-virus technologies feature some level of heuristic analysis (such as bloodhound), these provide little data concerning the threat and often involve proprietary identification algorithms making it almost impossible to understand why an analysis target may have tested positively. The past five years have seen the proliferation of several 'sand-box' technologies, whose purpose in life is to automatically simulate a attack targets environment, and provide technical data regarding what it is that the code does. In their own right, these technologies are well matured, and do exactly what they were designed to do – which is to answer the questions of:

- How did the code get in?
- What did it do once was in?
- Is it still there?
- What data was lost?
- Can it be prevented in the future?

These are questions frequently put to incident response groups and as such, the automated analysis tool set of today has evolved around the need to answer them. This data however, does not paint a complete picture and is therefore not an appropriate tool set, to tackle the problem of attribution, with a high level of accuracy.

Static and Runtime Analysis

Similarly, almost all manual, and partially automated static and runtime analysis technologies, are designed to (out of the box), answer the question of, "what does the code do?" Typically, the component of most interest to the malware analyst revolves around what changes are made to the target system, to provide an understanding of how an incident can be recovered from and what (if any) other systems were impacted. This will often involve a report detailing items such as which registry keys and files were modified and any modifications that may have been made to memory. This is all useful information, however it still fails to draw us closer to knowing more about the adversary responsible for the attack. Further, much malicious software in the wild today will result in very similar reports when analyzing code functionality.

Developing the Tool Set

As we have seen, many technologies exist today to assist with the analysis of malware, however none fully process and present data in a manner which is fully conducive to the attribution effort. Through leveraging new and existing technologies to develop a tool set specifically aimed at determining adversarial characteristics, we both bring ourselves closer to a methodology with which various levels of attribution is possible and address many of the challenges of modern incident response organizations, such as being able to quickly identify a needle in an otherwise uninteresting haystack of code.

Code Isomorphism

The term isomorphism – borrowed from the abstract algebra world -- comes from the Greek word "isos" meaning equal and "morphe" meaning shape or taken together, equal shape. Informally, it is used to refer to similarities between objects and has subsequently been adopted within the information security community as a way of establishing similarities between computer programs and other binary images, which themselves are highly structured. Code isomorphism techniques are an important tool for establishing lineage between related code (particularly malware). If performed correctly, this provides some immediate value to through an ability to identifying derivatives of known code samples. This in itself allows us to potentially:

- Identify related malware strains;

- Quickly identify modifications to known code samples; and
- Track sharing / distribution of malware between adversarial groups.

A significant amount of research into code isomorphism exists including noteworthy contributions from Pedram Amini and Halvar Flake[4]. In addition for being used for vulnerability analysis, and more recently malware analysis, the anti-code theft world has made use of code isomorphism techniques, to identify commercial products that may have resulted from code stolen from a competitor. In their white paper[5] entitled "A Static Birthmark of Binary Executables Based on API Call Structure," Seokwoo Choi, and Heewan Park, et al, discuss ways in which to apply statistical algorithms to identify programs that are likely derived from one another. Other tried and tested methods for studying the isomorphic properties of two binary images include: *small prime product* (SPP) calculation, where unique prime numbers are assigned to each opcode (or instruction) such that each function is now assigned a unique, large prime number. This technique is useful because it is resistant to instruction re-ordering. *API Structure analysis* is where analysis of API calls used by a given function can be utilized to automatically establish relationships between 'similar' functions in other programs. *Function checksums* are similar in concept to SPP, however they often utilize a standard checksum algorithm such as MD5. *Constant tracking* is constant, local or global values utilized in programs can be used to provide additional evidence that a program is related to another.

In all cases, when trying to establish a relationship between two or more programs for purposes of attribution, no single technique should be involved in determining if code has been shared. As we are seeking the highest possible level of confidence to prove or disprove the presence of related programs, all of the above techniques should be leveraged to corroborate one another – as none are likely to provide this answer by themselves.

Exploit Analysis

Exploits used by malicious programs to initially compromise a target, and propagate further are another avenue which we can perform automated analysis of to determine a number of factors. Exploits used by professionally written malware, will frequently differ from their counterparts released on the internet to prove the concept behind a vulnerability. First and foremost, the reliability of exploits used by malware to propagate is of great important to most – whether an organized crime group seeking to build a botnet, or the efforts of a state funded organization. To each, a detected attack represents a lost opportunity to meet an objective, and the possible compromise of a valuable vulnerability, and the code intended to provide persistent access. As such, code is often reengineered to assure a high level of reliability and acceptable behavior during failure (i.e. that the host does not encounter a GPF, or other fault). Analysis of exploit triggers, and payloads can help quickly identify if a known flaw is being leveraged and if the exploitation process matches that of a known (public or private) exploit. Further to this, data regarding unique aspects of the reengineering process will provide insights into the technical capabilities of the author.

Black Axon

Black Axon is a concept project, developed to demonstrate how automated analysis of a binary can provide insights into the author(s). Through leveraging a database of known code profiles we are able to quickly establish similarity to known malware samples, and identify techniques that we know to be associated with certain actor groups. While it is always of interest to understand what a malware sample *does*, for the purposes of profiling the authors, more interest is placed around ***how a given function is achieved***.

In its current implementation, we leverage int3 debuger breakpoints to identify the execution of a predefined set of API calls. On execution of a function of interest, we attempt to establish the context of its execution, through proximity to other operations and API calls, return value, and parameters. We also have the opportunity at this point to establish any isomorphic relationships to functions in other known samples. By identifying the context of the call, we are able to understand its use, and therefore identify specific techniques used to implement a given function within the program.

When combined with a scoring matrix, we can quickly identify the use of certain techniques, in unknown malware samples which can then be used for purposes of attribution as follows:

- Establishing the likely technical aptitude of the author(s);
- Identifying the use of known and unknown techniques; and
- Matching technique use with the modus operandi of known actors.

Today, many of these outcomes require hours of manual analysis, and are subject to human error. Through automating this process, not only can we significantly improve the efficiency of the analysis effort, but also provide a model through which definitions of known tools and techniques can be shared between collaborating teams.

Currently, due to its use of conventional debugger breakpoints, it is unlikely that the tool would scale well or prove effective against many of the anti-debugging techniques utilized by modern malware. Debugger breakpoints were the method of choice due to the ease at which the context of a call may be established within the debugger environment (in this case IDA Pro). Future adaptations may leverage DETOURS or Kernel level hooks in order to identify the use of an API call of interest, while reducing the likelihood of being detected by the code in question. Additionally, API hooks provide broader coverage over the operating system environment with which possibly compromised processes can be monitored for the presence of malicious code, injected via dynamic link library/process injection and exploitation.

Conclusions

The current tool set available to those performing analysis of malicious software are not suitable for purposes of attribution but are being used in this way in the absence of an alternative. Through adapting well tried and tested analysis techniques, we can build tools, and data models that *can* provide assistance in the attribution process, providing a solid, broadly understood methodology which can be used to back up what may have previously been poorly supported attribution theories. In many cases, this will help disprove the knee-jerk, pedestrian attribution efforts that many in recent years have fallen foul of.

References

[1] http://en.wikipedia.org/wiki/2007_cyberattacks_on_Estonia
[2] http://en.wikipedia.org/wiki/2008_South_Ossetia_war
[3] http://en.wikipedia.org/wiki/GhostNet
[4] http://www.zynamics.com/vxclass.html
[5] http://plus.kaist.ac.kr/~han/ASIAN07.pdf