More Than Malware Families: Retooling Our Approach to Tracking Software

vertex.link/blogs/more-than-malware-families/

by savage | 2025-01-14

One way we track suspicious cyber activity involves identifying the tools used and categorizing them into what we colloquially refer to as malware families. However, while shorthand has its place, as analysts, we need to be as precise as possible when making and explaining our assessments. When we say that a file is part of a malware family, what do we mean? How do we define that malware family? Does it consist of just the backdoor, or does it also include the file that launched the backdoor? In this blog, we'll introduce the methodology that The Vertex Project analysts use when analyzing software, and explain the choices we've made in an effort to more accurately capture and communicate our findings.

Why Identify and Track Tools?

Naming, categorizing, and tracking tools can help with:

- **Identification**: Defining and naming the tool for ourselves makes it easier for us to track it, as we now have a way to refer to and evaluate the tool when we come across potential samples.
- **Detection**: Identifying and tracking tools often plays a critical role in detecting and responding to malicious activity. We can use our understanding of a tool to develop more effective detections, as well as triage resulting alerts based on the risk to our organization.
- **Categorizing sourcing**: Information about a tool's developer and sourcing can shed light on its intended use and availability, as well as provide insight into the operators' resources and potential connections to others. The use of the same privately developed tool, for example, may indicate a partnership or shared supply chain among threat actors.
- **Characterizing activity**: We can use insights gleaned from identifying tools to better characterize suspicious activity. In some cases, knowing what tools a threat actor is using can help us to assess the actors' likely aims - are they seeking to steal data? Deploy ransomware? Both? - and for incident responders to react accordingly.

These are just a few of the benefits that can stem from tracking tools. However, as with analysis as a whole, the accuracy and the resulting utility of doing so depends on our methodology.

More Than Malware

When discussing potential approaches for identifying tools, The Vertex Project analysts all agreed that we wanted a process that would grant us both a broad view of software, as well as a more detailed understanding of how different tools are structured and work together. For us, this meant:

- Moving away from the concept of malicious software and threat actor tools to consider software in general, and,
- Accounting for the relationships between different kinds of software, such as different files that are used in conjunction with one another (like an installer and a backdoor).

With these goals in mind, we decided that rather than focusing solely on malware - tools created for use in malicious operations - we would focus instead on tracking software in general. We could then apply a consistent methodology to native utilities and commercially available tools, as well as to those developed by or for threat actors. Threat actors can and do abuse a range of tools in their operations, including commercially available remote administrative tools and native utilities that allow them to "live off the land" rather than deploy their own custom tooling.

However, it can also be helpful to identify common utilities and software in general, regardless of their potential use for malicious purposes. The more tools we are familiar with, the more easily we can recognize and filter out known software and activity. We can then focus our investigative efforts on the unfamiliar, which may include known tools employed in unfamiliar ways.

In addition to expanding our focus to software in general, we also sought to better define and scope the categories into which we sort that software so that we can capture relationships between tools that may support or operate in conjunction with one another. We don’t just want to know what a tool is - we also want to know how it may relate to other tools and resources we might come across. Therefore, we created three levels of groupings: code families, software suites, and software ecosystems.

Code Families

Code families are the most foundational element of this organizational structure and consist of individual files based on the same or highly similar source code. Although comparable to the idea of a malware family, code families differ in both scope and application. Analysts

base code families on what they determine to be an overlapping code base or subset of key components that are unique to or strongly representative of the code family.

Analysts may create code families for native utilities and commercially available files, regardless of their use in malicious operations. A code family may represent Cobalt Strike's Beacon backdoor, for example, while another may document Microsoft's PsExec. As code families are granular by design, software composed of multiple components may have several associated code families, each representing a different module's unique code base.

While we use code families to identify individual files, these files often work in concert with other resources. We can document these relationships as Software Suites and Software Ecosystems.

Software Suite

A software suite consists of files that are dependent on each other in order for the overall software to operate correctly. An analyst may categorize the various files comprising Microsoft Office, for example, as a single software suite. We can consider a software suite as a specific subset of a Software Ecosystem.

Software Ecosytem

A software ecosystem consists of at least one code family and additional, related resources that are typically used in conjunction with one another. For example, an analyst may create a code family for a specific backdoor, but then also want to document a file used as a launcher, as well as the C2 infrastructure that the backdoor communicates with. We can document the relationship between these elements - backdoor, launcher (which may have its own code family), and C2 infrastructure - by grouping them into a single software ecosystem.

Categorizing Software to Aid Analysis

With these three levels of categorization - code family, software suite, and software ecosystem - analysts can categorize and track a specific piece of software or unique code as well as a grouping of files and software resources used as part of a tool or specific program. This lets us answer a variety of questions, such as:

- What is this file?
- What are these files together a part of?
- What are the required components of this program?
- What additional resources are related to this software? What does it require to run? Communicate?

- How can I detect or identify this one file, component, or related activity?

Retooling Our Approach

In accompanying blogs, we'll elaborate further on our approach to identifying and grouping tools, beginning with how we define and use code families within our research and analysis. We'll describe the methods available for identifying and tracking code families, and the pros and cons of each, before walking through some examples. Finally, we'll discuss how we can then represent our findings within Synapse.

We intend to share our approach as both an example and a conversation starter. As always, the best approach for you and your team is going to depend on your mission and resources - what works for us or for another organization may not be appropriate for your needs.