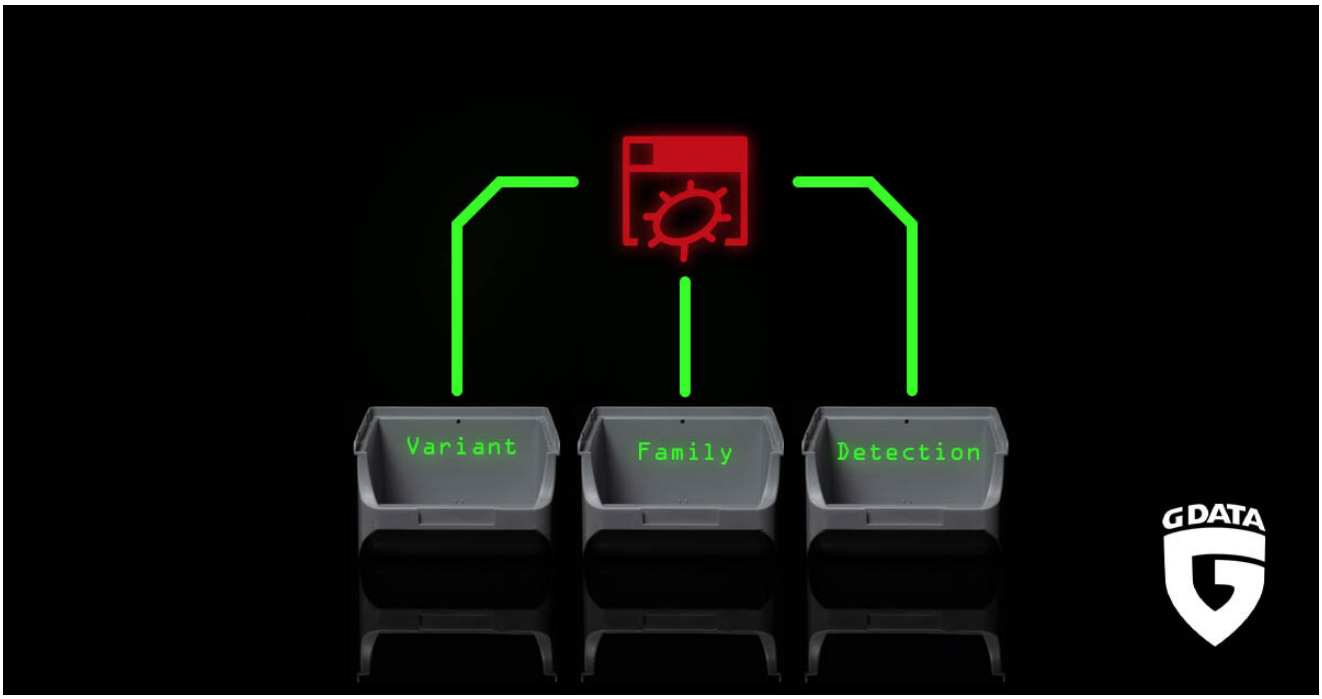
Malware family naming hell is our own fault

gdatasoftware.com/blog/malware-family-naming-hell



EternalPetya has more than 10 different names. Many do not realize that CryptoLocker is long dead. These are not isolated cases but symptoms of a systemic problem: The way we name malware does not work. Why does it happen and how can we solve it?

Current state of malware naming

Malware names are not clear. Neither the terms related to them have a common understanding, nor the names themselves. There is no common standard. There is no institution, database or organization that has an exhaustive list of malware names and their definition.

Our current use of malware names and their creation suffer from the following problems.

Problem	Examples
Malware families and variants have several names	EternalPetya probably holds the record. Some of its names (not exhaustive): NonPetya, NotPetya, Petna, ExPetr, Pnyetya, Nyetya, nPetya, BadRabbit, EternalBluePetya, BluePetya, petrWrap.
One name used simultaneously for several families	The ransomware JesusCrypt in 2019 and a different JesusCrypt in 2021. Both use the .NET framework.

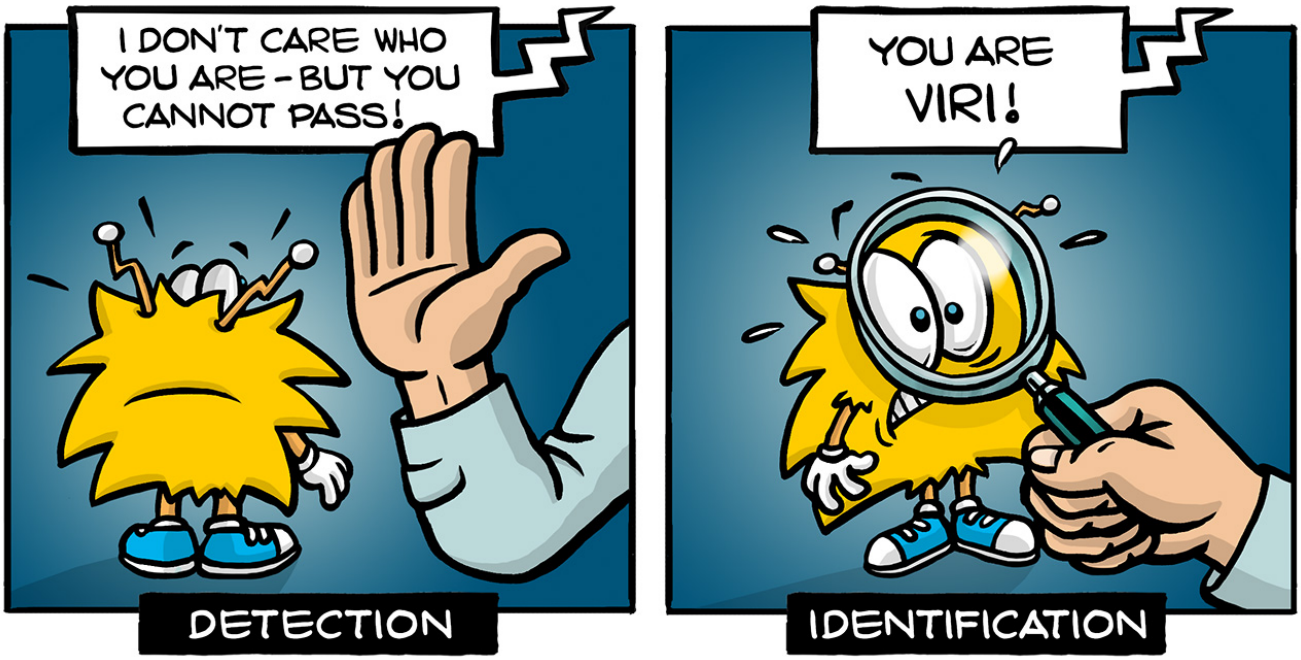
Problem	Examples
Malware families are conflated with their detection name	Malware prevalence reports by many antimalware companies write "malware family" but use detection names.
Malware families are conflated with their loader, downloader, spreading campaign, threat actor(s), or packer	Gootkit and its loader were conflated, although the loader also ships other families than Gootkit.
The meaning of a name can change over time	Nemucod was at first only a family and was later used to refer to malicious JScript downloaders in general.
The meaning of a name can depend on the person or organization using it	Artemis by McAfee does not refer to the malware family but to a detection technology.
The same name is used for the family as well as the malware type	CryptoLocker may refer to the ransomware family or file encrypting ransomware in general.

Terminology: Detection names are misunderstood

As I explained in my previous article about detection names, the CARO Virus Naming Convention back in 1991 were an attempt to streamline malware naming and taxonomy. However, the threat landscape changed which made CARO's naming convention outdated. As a result the antimalware industry adopted and modified the CARO's naming convention to their own needs; but the purpose of these malware names shifted from identification to detection.

Identification has the goalto determine the correct malware family and potentially also the variant.

Detection has the goal to distinguish between clean, potentially unwanted and malicious files, registry entries, settings, events, or requests, so that the user's systems can be protected from harm.



"Detection" and "Identification" are often used synonymously, but there is an important difference between the two.
Big testing organizations like AV Test and AV Comparatives therefore test the detection capabilities of antimalware products, not their correct identification of malware. For antimalware products there is not much incentive to program their scanners for identification. They look like they identify and classify malware but they are doing a pretty bad job at this.

A **malware family** is a group of malware samples that have a common code base.

A **malware variant** is a subgroup of a malware family. Different malware variants have notable derivations from the code base of the family. One malware variant contains all samples that have the same derivation.

An example for a malware family is Petya, whereas GoldenPetya and GreenPetya are variants of the Petya family. The most notable difference of these Petya variants is the color of their ransom note text. The terms malware family and malware variant should **not** be confused with the *Family* and *Variant* component in detection names.

Detection namesare readable names that map to certain detection signatures or technologies. Detection names are used by antimalware products and vendors.

Because the purpose of detection names is not identification, they are not viable to be used that way. But as we can see in many malware prevalence reports, detection names are still assumed to represent malware variants, leading to confusing and wrong statements by media and news.

Yes, detection names can contain malware family names in their *Family* component. However, it is not necessarily the correct one, nor is it clear if that part of the detection name is actually a family or an umbrella term or something else (see Detection naming conventions

today).

Yes, detection names contain a *Variant* component, but this component does not represent a malware variant. It is rather used either as a counter that increments with every added detection signature or it represents a hash value which might be a different one for every sample. The *Variant* component in detection names is crucial to the mapping for the detection technologies and signatures and used to identify and maintain them. Thus, they are an internal information that is only useful for the antimalware vendor.

Now that the terms used in this article are clear, we will focus on how and by whom malware family names are created in order to answer these questions: What is actually the issue with this process? How can we make it better?

Creation of malware family names

Malware family names are mainly created by malware analysts. The general procedure is as follows.

1.Cross-checking with known malware families

A malware analyst gets a sample to analyse. A primary analysis may tell them if the sample is a known malware family, e.g., because the analyst has seen this family before and recognizes strings in the binary, the code or the behaviour.

If the malware analyst doesn't recognize the family, they search for clues on what it could be. This can be done via the following methods:

- The analyst extracts strings from the binary. They research those strings that seem unique, e.g., a mutex, unique file path, strings that contain typos
- Sometimes the malware developer's project name is part of the binary, e.g., via the PDB path. Many malware family names are based on the project name.
- Sometimes the malware developer includes their own nickname or social media handle, e.g., for Twitter, Telegram, Discord. Researching their accounts often leads to postings or videos about their malware which include the developer's malware name
- The analyst checks the hash on various sample databases and comments by other analysts, e.g., on Virustotal.
- The analyst checks detection names by other vendors, e.g., on Virustotal, preferably on the unpacked sample. If certain family names appear in many different scanners, the chance is higher that this might be the family.
- The analyst looks up general behaviour and extracts interesting data via automated analysis systems, e.g., Any.Run, Hybrid-Analysis, VMRay, ... These may also be used for cross-checking.
- Internal as well as public analysis systems may provide similarity analysis to known families. A good example is Intezer.

- Asking colleagues. A tweet may provide more than hours of research.

Ideally the malware analyst now has assumptions what family it might be. They now cross-check their assumption with public analysis reports. This includes searching for aliases of the same family and checking them as well. Public analysis reports are crucial at this point.

If no malware family can be found or if the name is not suitable for the analyst's requirements, they create a new family name.

2. Inventing or deriving a new family name

Family name creation is a pet peeve of many malware analysts, especially those who analyse lots of samples a day to create detection signatures. They often just need quickly a name to tie it on their detection signature. Time is crucial, especially if there is a current malware outbreak on customer's systems. Remember: detection names are nothing more than mappings to the detection signature. The family part of the detection name can feel particularly frustrating if it is holding one back to commit the signature. How often did I end up in this situation:

1. I find a name that sounds great. I type it into Google, and it turns out to be a **city**.
2. I change my name by mixing up some of its letters. Google now tells me this new name is a **company's name**.
3. I reverse the string. Google tells me it is a **person's family name**.
4. At this point I use my fail-safe method to create a completely unique name: I roll my head on the keyboard while making frustrated noises. I call this the "I give up" method. Allegedly some analysts let their cat sit or walk over the keyboard while others give it to their toddler to play with. This "random" generated word turns out to be an **offensive word** in a few languages that I don't speak.

All of the mentioned names, companies, persons and offensive words, are not suitable to be used as malware family names. Additionally trademarks, products and common words are not permitted. No person or organization would like to see themselves associated to a malware. Side note: Company or product names may show up in detection names if it is a PUP (=potentially unwanted software) detection. But that's **not** the same as a malware family name. In many antimalware companies it is also not permitted to use the name that the malware author intended. The reason is that the malware author should not get any fame for their product.

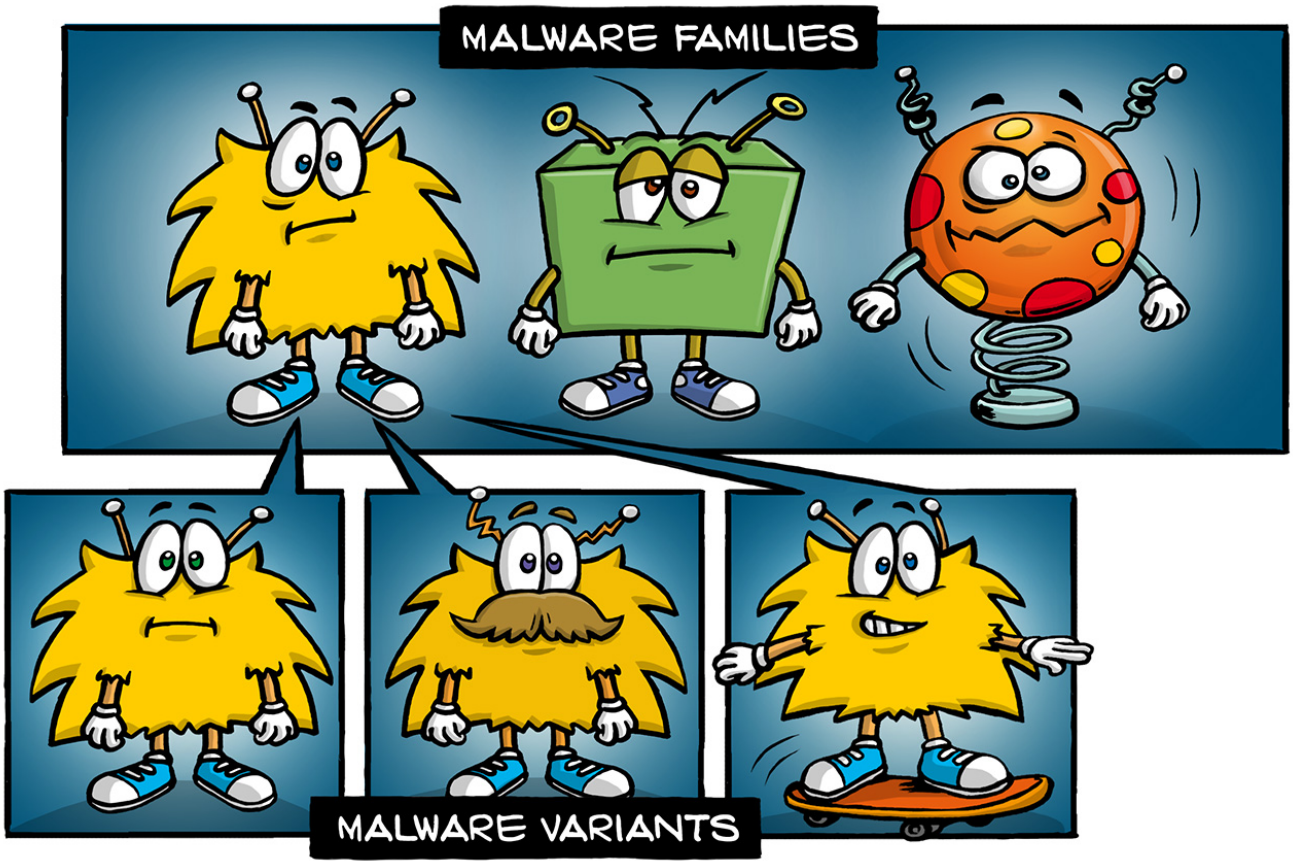
Apart from this process, what strings are actually used to create or derive a family name?

As I mentioned above the analyst cross-checks **unique strings** as well as **project names** with known family names. If no known family names come up, these strings are often used to derive the new family name. In some cases they are used verbatim, in others they are

modified to fit to the naming policies. Malware authors often use racist, sexist, homomisic, ableist or plain offensive language that may bleed into their project names. It is obviously not suitbale for a family name.

Apart from unique strings, analysts may also derive names from the sample's file path, its programming environment or language, its malware type and behaviour. E.g., PyCrypter is a mixture of **Py**thon and a file en**crypt**ing ransomware. The worm Conficker is a pun for "configure" and the German word "Ficker" which means "fucker" in English. In this case the rule to not use offensive words has been violated, but the name stuck.

Malware analysts do not only use puns to derive names. They literally like to **reverse** (pun intended). E.g., the first versions of Nemucod loader used download URLs that contained "document.php". **Nemucod** is "documen"(t) backwards. The "t" was likely removed to make it pronouncable.



Case study: How I misnamed GooLoad aka GootLoader

Gootkit is a banker that was shipped with a very specific combination of PowerShell, JScript and .NET assembly to allow fileless persistence on the infected system. This loader, later dubbed *GootLoader* and *GooLoad*, was described in previous Gootkit articles but not specifically named.

At some point the loader started shipping malware other than Gootkit, such as Gozi which is also a banker. Unaware of the situation, one of our malware analysts correctly identified such a sample as Gozi and wrote a signature that included patterns of GooLoad as well as Gozi. Due to the nature of the infection, fileless persistence via registry stuffing, our team created specific cleaning procedures for samples detected by this particular signature—Gozi in combination with GootLoader.

New infections appeared later that shipped a different malware via GootLoader. The Gozi signature did not match anymore, but we identified many common strings in that signature for the old and new infections. Subsequent signatures contained only GooLoad strings but were also named Gozi.

At this point I realized that team members had added the very same registry cleaning algorithm for certain Gootkit signatures as they did with Gozi. At that point I started to question the malware family identification. Most of our Gozi signatures contained GooLoad strings at that point, yet these strings were described in Gootkit articles. A few tweets and lots of research later it dawned on me that this malware did not have a name yet.

I renamed the related Gootkit and Gozi signatures to *GooLoad*. I chose this name because "goo" means sticky substance and *GooLoad* is somewhat close to Gootkit loader but still different enough to not associate it as loader for Gootkit alone. While I was working on a blog article to clear up the confusion, Sophos researchers settled on the name GootLoader at the same time.

So yes, I am at fault for spreading false information about Gozi (which I later corrected) and also for creating yet another name for GootLoader aka GooLoad malware. The reason for not reverting the name to GootLoader is simple: It would require too many changes in too many signatures and cleaning algorithms and none of them improve the protection of our product. This is why coupling malware taxonomy with detection names is not a good idea.

Underlying causes of misnaming

Some of the causes might have already popped up in your mind while reading the malware name creation section.

Malware analysts who create malware names often don't have the time nor the incentive to be accurate. Even if analysts want to be accurate, it is a tedious task that may still not result in the correct name. Analysts create new family names if they can't find any that fits.

Mistakes are often not noticed because no testing is done for identification. Even if naming mistakes are noticed they are often not corrected: Doing it consistently requires a lot of effort and doesn't improve protection.

Names that have already been established may not be applicable for every party involved. There may be policies that prevent the use of these names, and such policies are different in every organization. Therefore, new names are created to abide by these policies. The names may also need to fit PR purposes, e.g., Sodinokibi is hard to pronounce and to remember, but its alias REvil is much better suited for blog articles and news.

We have no common agreement on malware naming, nor is there any review procedure or institution to oversee it. Wrong identification or use of names is usually not even noticed.

Noteworthy outbreaks of malware infections require fast reaction times by malware analysts as well as the news media. This results in simultaneous creation of new malware names for the same family or variant. It is the reason why the Petya variant EternalPetya has so many names—its first outbreak was pandemic and all antimalware companies had been working on it on the same day.

Malware detection is already difficult, mathematically it is an *undecidable problem* (see Fred Cohen, 1987, Computer Viruses, p. 28). Identification more so because it requires additional steps. If I identify a malware family, the detection of its malicious components is a prerequisite.

Solutions to malware naming hell

The antivirus industry focuses on protection, not identification. Yet, their employees are the main creators of malware names, most of the time indirectly via detection names that are picked up by publications and news. As a result we have at least as many malware naming conventions as there are antivirus companies, and no common understanding.

Step 1: Detach detection names from malware taxonomy

If detection names did not attempt or pretend to be a malware taxonomy, many misunderstandings, misuses and wrong identifications can be prevented. So what we need is a **decoupling of detection names and malware taxonomy**. An alternative way to create detection names is described in the IceWater project.

If any product seemingly identifies malware, it should be **tested for identification capabilities**; not only for detection. It must be made clear for consumers of such products whether a product detects or identifies. Pretending to do one thing while actually doing another is unfortunately how detection names are presented currently.

Step 2: Create a common taxonomy and quality process

Once there is a decoupling (see Step 1), the antimalware industry and sciences may be open to agree on a **common taxonomy, policies to deal with name conflicts, and a process to ensure quality**. That's because this agreement doesn't directly affect

antimalware company's detection names anymore which are necessary for their operational work. Without that detachment, antimalware companies would not be able nor willing to unify malware family naming or the naming procedure.

We might adopt **scientific peer-review procedures** for malware analysis papers and new malware names just as it is common in academia to peer-review research papers.

Step 3: Build a vetted, public malware database

We have public sample and malware databases, but they are not suited for this purpose yet. E.g., there is Malpedia by Fraunhofer FKIE. It is a malware database of currently 2023 families with short descriptions, some aliases, links to blog articles and Yara rules.

However, Malpedia does not comment on the linked articles which naturally contain contradicting information. It merely acts as a reference collection. The Yara rules of Malpedia do not suffice for proper identification of malware families as many of them suffer from false positive and false negative matches as well as conflation of a family with its packer, loader or other detached components. This is not necessarily the result of bad rule writing. Rather the methodology of using signature-based detection is not best suited to identify families.

Intezer has a more adequate methodology for identification. It compares similarities of code and strings to reference files of a malware family. It does this on the sample itself as well as the memory contents while running it. But Intezer is not meant as a database to look up families, aliases, and their common defining behaviour as well as capabilities.

We need a **public malware database** that includes:

- malware families, including their aliases, but with *one* official name
- representative, non-packed samples for each family
- a detailed description of what makes up the family, and potential border cases to other families that might look similar
- a code and string based comparison of the family's main body to other input samples (like Intezer); this must not include its loader, downloader or unpacking stub, unless the family itself is a loader or downloader
- vetting of new entries

Feasibility

The first step is probably the hardest. The same inflexibility that causes the inaccuracy of malware naming also makes it difficult move away from a pretense malware taxonomy in detection names. Especially because there is no immediate gain by doing this step. On the contrary, loosing the pretense malware identification might seem like a loss at first. An easier transition might be possible by just adapting new detection names and technologies or only changing what is shown to the user of the antimalware product.

You might ask if such a database is possible with the new appearance of threats every day. I am confident that it is! The huge threat counts we see in malware prevalence reports are based on sample counts, or infection attempts, but *not* families. Malpedia currently lists only 2023 families, and during my daily work I usually find what I am looking for. There are certainly a few thousand families more than those listed on Malpedia, but the magnitude is easy enough to handle. The number of families can be reduced if we concentrate on those that are at least active for a few months.

The gains in the long run would be huge. If we actually made improvements in malware family naming, it would be easier to find information about malware, mistakes were less likely and work would less likely be done twice (e.g., because a family is already known but the analyst did not find the information). That in turn improves detection signature writing, reponse times to malware incidents, adequate treatment of such incidents, threat prevention and malware research time and quality.