Malware Naming Hell: Taming the mess of AV detection names

By Karsten Hahn

Published: 2020-03-02 · Archived: 2026-04-10 02:06:07 UTC

08/12/2019

Reading time: 9 min (2423 words)

Everyone who deals with malware will know this: Malware names are a convoluted mess. AV scanners will show different detection names for the same file. This confusion is also reflected in media coverage. Is there a way out of this mess?

Before we start our expedition into this muddled place, let's get the terminology right. "Malware name" might refer to one of the following:

1. AV detection name
    - Those are the names an Antivirus product will show in a pop-up or log screen if it found an infection on the system. Those are also the names you see on multi scanning services like Virustotal.com.
2. Malware family name
    - A malware family describes all malicious samples whose payloads have the same or similar source code as origin. There is no clear line when a derivation of the malware source code creates a new family or when it is another variant of the same family.
    - The family name can be, but doesn't have to be, part of the AV detection name.

The first part of our series examines Antivirus detection names. The second part is a dive into malware family names.

1. The past: CARO virus naming conventions (1991)

The first attempt to make malware naming consistent was in 1991, when a committee at CARO created A New Virus Naming Convention. This was a time where all or almost all existing malware was also a virus. The naming scheme has influenced today's detection names. Most AV vendors use the same or similar components that CARO suggested but often with their own terminology and ordering.

> The full name of a virus consists of up to four parts, desimited by points (‘.’). Any part may be missing, but at least one must be present. The general format is Family_Name.Group_Name.Major_Variant.Minor_Variant[[:Modifier]

(CARO, 1991, A New Virus Naming Convention)

This article will not describe all of these components in detail but highlight some points. The best description is in the conventions themselves on CARO's website.

The *Family_Name* portion of the detection name doesn't always denote an actual malware family. CARO's conventions provide four umbrella names for insignificant viruses:

1. "Trivial" for viruses smaller than 100 bytes of code. The infective length is appended as number to the *Family_Name*.

2. "Silly" for viruses that "do not contain anything particular that can be used to name them". Modifiers are appended to *Family_Name* to denote boot sector viruses or types of files that are infected by Silly, e.g., SillyRC for **r**esident viruses that infect **C**OM files, or SillyB for DOS **b**oot sector infectors
3. "HLLO" for overwriting viruses written in high-level languages.
4. "HLLC" for companion viruses written in high-level languages.

The *Group_Name* represents a sub group of malware in the same family employing the same rules and does and don'ts as the *Family_Name*.

The *Major_Variant* and *Minor_Variant* create even more specific sub groups. Often the infective length of the virus or consecutive letters or numbers are used for these components.

The Modifier is applied to the detection name if the threat actor used a third-party component to conceal their virus, e.g., a polymorphic engine or a compression tool.

CARO's naming conventions are well-conceived. The format pattern starts with the most generic information and becomes increasinlgy more specific. The does and don'ts listed for the *Family_Name* portion of the format make sure that the names are easy to remember, well-readable, don't interfere with each other, don't offend any third parties (companies, people), and don't use mutable characteristics to describe a family (e.g. activation dates). They were created with forethought and are still applicable today.

2. Detection naming conventions today

The malware landscape has changed since 1991, and so have the means of detecting malware, which might be one reason that the CARO conventions didn't work out in the long run.

Nowadays:

- viruses are rare, most malware does not infect files. Detection naming conventions must consider the currently wide-spread malware types.
- the file length has lost its meaning for malware identification and grouping, thus, it plays no part in detection names anymore.
- most malware is packed.
- more generic and heuristic methods are used to detect malware, e.g., instead of identifying a malware family, products may just identify malicous behavior. A lot of signatures are created automatically, resulting in detection names that contain information about the detection technology instead of describing the malware itself.

Every AV vendor has their own detection name conventions, but most of them use similar components. They usually include: platform, malware type, malware family, a variant component and additional information that is prepended or appended to the name.

The *platform*, if it is part of the detection name, specifies the execution environment for the malware. This may be the operating system and architecture (e.g., Win32), a framework (e.g., MSIL for malware using .NET framework), or a programming language (e.g. Powershell). Unspecific terms for unknown execution environments exist too, e.g. Script may denote any text-like file.

The *malware type* tells something about the behavior of the malware.

The *malware family* component in a detection name is either the actual family name of the malware or an umbrella name for a broad range of families if the family is not known.

The *variant* in a detection name is mostly just a counter to distinguish different signatures from each other whereas in the CARO conventions it was meant as an actual variant of the malware family. The variant may consist of numbers or letters or both. In automatically created detections the variant portion may also be generated from the file itself, e.g., via hashing.

The *modifier* component is optional. It adds additional information about the detection. It may further specify the malware's behavior or type or add a characteristic about the signature, e.g., it may state that the signature is heuristic or generic.

The table below lists some Antivirus vendors with links to their official detection naming conventions (if available) and their format using the components above.

AV vendor	**Format**	**Example**
Avast	Platform:Type1-Modifier \[Type2\]	VBS:Downloader-ARK [Trj]
AVG	Type Family.Variant	Trojan horse Crypt8.BHVG
Avira	Modifier/[Type.]Family.Variant	TR/Inject.xbbeicg TR/AD.SodinoRansom.wcoir
Bitdefender	[Modifier: [Platform.]]Type.Family[.Modifier].Variant	Gen:Trojan.Mresmon.Gen.1 DeepScan:Generic.Ransom.Sodinokibi.5460CDF8
ESET	[Modifier] Platform/[Type.]Family.Variant Type	a variant of MSIL/TrojanDropper.Agent.BPM trojan
G DATA	Platform.Type.Family.Variant[@Modifier]	MSIL.Backdoor.Yantac.A@susp
Kaspersky	[Modifier:]Type.Platform.Family[.Variant]	HEUR:Trojan.Win32.Nymaim.gen
McAfee	Platform/Family Type Platform/Family.Variant.Modifier	RDN/Generic BackDoor W32/HLLP.11042.gen
Microsoft	Type:Platform/Family.Variant[!Modifier]	Trojan:Win32/Gandcrab.AF
Trendmicro (old)	Type_Family.Variant	TROJ_GEN.R002C0WGH19
Trendmicro (new)	Type.Platform.Family.Variant.Modifier	
Symantec	PlatformOrType.Family.Variant	Trojan.Gen.MBT

3. Tips and tricks for interpreting AV detection names

AV detection names can contain useful information. However, everyone working with them must be aware that often they don't, and sometimes the information in them is wrong. So if you have a scan result listing of a service like Virustotal, how do you know which detection names can be trusted? A rule of thumb:

The more specific detections from different scanning engines exist that are consistent with each other, the more likely the information is correct.

The sections below examine how to differentiate specific from non-specific detections and explain how to interpret certain key words used in detection names.

Understood what you just read? We are looking for you!

3.1. Antivirus terminology is quirky but you need to know it

The inconsistency and shift of meaning for terms used by the antivirus industry makes communication difficult. A well-known example is the use of the term "virus". Initially this was the most common or only malware type that existed. Peter Szor defines a virus as follows:

> A virus recursively replicates itself by infecting or replacing other programs or modifying references to these programs to point to the virus code instead. A virus possibly mutates itself with new generations.

(cf. [Szor05, p.27, 36])

Antivirus products were true to their name because they protected systems from viruses. Nowadays, antivirus products mostly protect systems from malware regardless if they are viruses or not. Yet, the term "virus" has become synonymous with "malware" in mainstream media.

The word "trojan" is another example. The "trojan horse" describes a way malware can be delivered to the target system by pretending to be a useful and benign program or even providing useful functionality and tricking the user into execution (cf. [Szor05, p.37]). Thus, it describes a specific infection vector. It is in most cases not an inherent characteristic of a malware family but a way third parties make use of the malware. However, in detection names and in mainstream media the term "trojan" is used synonymously with "malware". We can often see that "trojan" is used as default value for the type component in unspecific detection names. Maybe that is why mainstream media picked up the term to describe any malware.

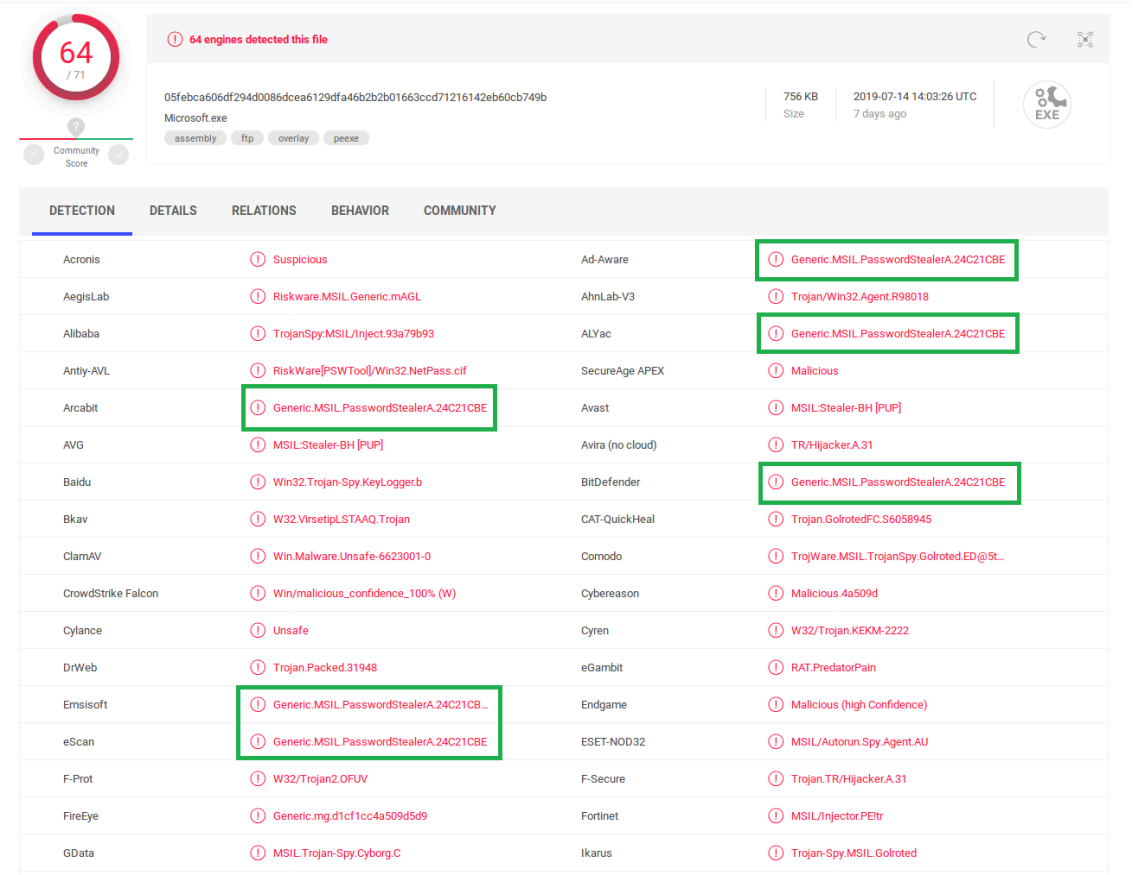
Apart from these quirks there are also certain key words in the malware family component of detection names which have specific meanings. Umbrella names that include several malware families based on common characteristics will be explained in the second part of this series. However, some keywords in the malware family component don't have anything to do with the characteristics of the malware family itself. They are default values, detection technologies, or describe the malware's protection mechanism that was added by a third party. These are in the table below.

Key words	**Meaning**
Kryptik, Krypt, Cryptik, Crypt, Packed	Packed file
Obfus	Obfuscated file, mostly used for malicious script files
Injector, Inject	Packed file that injects into a process, usually via RunPE

Key words	Meaning
AntiXY	Protection mechanism against XY, e.g. AntiAV means the file might incapacitate AV programmes, AntiVM means it might refuse to run in a Virtual Machine
FakeXY, XYFake	The file imitates XY, e.g. FakeAdobe imitates an Adobe product. This is often done via third party tools that change the icon and version information of the file
Corrupt, Corrupted, Malformed	The file is corrupt.
Patched	The file was modified which makes it suspicious.
Agent	Default name for unknown or insignificant malware family
Razy, Kazy, Zusy, Graftor	Unknown malware family, detected by certain Bitdefender technology
WisdomEyes	Unknown malware family, detected by certain Baidu technology
Artemis	Unknown malware family, detected by certain McAfee technology

3.2. Be aware of third party scanning engines

If you evaluate a sample in Virustotal, you might see an image similar to the one below.



There are at least six scanners that have the exact same detection for this file, including the same variant portion which is usually a vendor specific value. The reason for this is simple: The results stem from one and the same engine -- in this case Bitdefender's engine.

AV products may often incorporate engines of other AV vendors additionally to their own. AV Comparatives has a list of AV vendors and the third party engines that they use. Bitdefender, Kaspersky and Avira appear as third party engines in this list.

That means if you evaluate multiscanner results, and you see that several vendors have the exact same detection name, including the variant portion of the name, it is probably just one and the same scanner. The detection in question should rather count as one detection instead of six (like in the sample above) while considering how valuable and informative the results are. E.g., a sample that has six detections from one and the same engine is more likely a false positive than a sample with six detections of six different engines.

3.3. Prefer specific over unspecific detection names

The more specific a detection name is, the more information you can draw from it. Unspecific names are those with default components. Descriptive components and umbrella names are a bit more specific. Actual identification of a malware family is the most specific.

Let's take a look at this packed Ursnif sample[1].

Ad-Aware	Trojan.Agent.DGAT	ALYac	Spyware.Ursnif
Arcabit	Trojan.Agent.DGAT	Avast	Win32:DangerousSig [Trj]
AVG	Win32:DangerousSig [Trj]	BitDefender	Trojan.Agent.DGAT
CAT-QuickHeal	Trojan.Mauvaise.SL1	Comodo	Malware@#1lvkk1eb6fihf
CrowdStrike Falcon	Malicious_confidence_100% (W)	Cylance	Unsafe
Cyren	W32/Trojan.QCQR-8401	DrWeb	Trojan.Gozi.345
eGambit	Unsafe.AI_Score_100%	Emsisoft	Trojan.Agent.DGAT (B)
Endgame	Malicious (high Confidence)	eScan	Trojan.Agent.DGAT
ESET-NOD32	Win32/Spy.Ursnif.BP	F-Secure	Trojan.Agent.DGAT
Fortinet	W32/Agent.FFF1!tr	GData	Trojan.Agent.DGAT
Ikarus	Trojan.Win32.Krypt	K7AntiVirus	Spyware (0052a9701)
K7GW	Spyware (0052a9701)	Kaspersky	Trojan-Spy.Win32.Ursnif.aahi
Malwarebytes	Trojan.FakeMS	McAfee	Artemis!15B2A3D1E076
McAfee-GW-Edition	Artemis!Trojan	Microsoft	TrojanSpy:Win32/Wastenif
NANO-Antivirus	Trojan.Win32.Ursnif.fisrkm	Palo Alto Networks	Generic.ml
Panda	Trj/CI.A	Qihoo-360	Win32/Trojan.Spy.fd1
Rising	Spyware.Ursnif!8.1DEF (CLOUD)	Sophos AV	Troj/Agent-AZXV
Sophos ML	Heuristic	Symantec	Trojan Horse
Tencent	Win32.Trojan-spy.Ursnif.Dzag	Trapmine	Malicious.high.ml.score
TrendMicro	TROJ_GEN.F0C2C00J518	TrendMicro-HouseCall	TROJ_GEN.F0C2C00J518
Yandex	TrojanSpy.Ursnif!SgNrrf7pyRI	ZoneAlarm by Check Point	Trojan-Spy.Win32.Ursnif.aahi

Green: specific detection names with malware identification; blue: descriptive detection names without identification

The detection names that are marked green identify the malware family Ursnif aka Gozi. "Wastenif" used by Microsoft seems to be an alias for Ursnif as well. The blue detection names don't provide the malware family but other information.

- **Trojan.FakeMS**: "Trojan" is here a default value for unspecified malware. "FakeMS" is a file that pretends to be a Microsoft file. If you look at the "File Version Information" (e.g. in the "Details" tab in VirusTotal), you will see that the file has indeed "Microsoft Corporation" in the "Copyright" metadata.

- **Trojan.Win32.Krypt**: "Trojan" is again unspecified malware. "Win32" usually stands for 32-bit Windows PE files. "Krypt" tells us that the file is packed.
- **Win32/Trojan.Spy.fd1** and **Spyware (0052a9701)**: "Trojan" is unspecified malware. "Spy" or "Spyware" stands for a malware that monitors the system for useful information like entered passwords and sends this information to the threat actor. The variant portion of the name is of no use for us.

All of the other detection names don't provide any information about the malware except the platform (Win32) in some cases.

- **Trojan.Agent.DGAT**: This detection name appears several times because it from Bitdefender. "Agent" is the default family name for any unidentified malware family.
- **Artemis!15B2A3D1E076**: "Artemis" sounds like a malware family, but has a special meaning for McAfee detections. "Artemis" was the previous name of the Global Threat Intelligence Technology (GTI) by McAfee. Detections which are created by GTI contain the name "Artemis". An explanation is also on this site.

3.4. Results are better for non-packed files

Packed malware has generally less detections and less specific detection names than their non-packed counterparts. This makes sense because packing is used by threat actors to evade AV detection.

But even if the packed file is fully detected, results for the unpacked file usually provide more accurate information about the malware identity. The scanners on Virustotal don't use the full technology set that is available for the actual antivirus product. They mostly rely on detection signatures that hit on the file without executing it, whereas in-memory scanning technologies like DeepRay aren't involved. Identification for packed files is often not possible or difficult if they aren't executed.

Therefore it is recommended to unpack the malware before scanning to get better results. An example is in the pictures below.

The packed file[2] is detected by only two engines. Both detection names indicate that it is a script file which is packed. The unpacked file[3] in the second image is detected by ten engines and seven of them state it is a downloader written in VBScript. F-Secure claims it is a JScript which will download Teslacrypt ransomware. That means the unpacked file's detection names reveal its actual behavior -- downloading other malware. Small downloaders like this one often don't get their own malware family name.

Referenced samples

Description	Detection name	SHA256
[1] Packed Ursnif	Trojan.Agent.DGAT	7a8e75dd59b0c9633c9976bb3f07a53fe5fdbd3330ce8ba90153697edff4fff1
[2] Packed VBScript malware	Script.Malware.KryptikPuf.A	c6e7f80c08b456f2786454c59dc0f3138c1c17b070d5f02b6acc814ac740d128

Description	Detection name	SHA256
[3] Unpacked VBScript malware	Script.Trojan-Downloader.VBS.TIOAOR	f43748608c9e904dd67ea5eb5650c15d0b62a1d3b1f917defef09ef946a0f553

Referenced book

[Szor05] Peter Szor. *The Art of Computer Virus Research and Defense*. Addison Wesley Professional, February 2005.

Related articles:

Karsten Hahn

Principal Malware Researcher

Content

Topics

- Tips and tricks
- Techblog
- Malware
- Security products
- Microsoft Windows
