M

Metamorphic Malware: How It Rewrites Code and Evades Signatures

Metamorphic malware rewrites its own code while preserving malicious behavior, producing structurally different variants. Learn how it differs from polymorphic malware and how defenders detect it.

PUBLISHER
Gridinsoft Support

UPDATED
Oct 8, 2026

READ TIME
4 min

CONTENTS

In this guide

01 Metamorphic vs polymorphic and oligomorphic malware
02 How metamorphic code changes
03 Why exact signatures and hashes are insufficient
04 How defenders detect metamorphic malware
05 What to do after a detection
06 How to reduce risk
07 Frequently asked questions
08 Further reading

Metamorphic malware produces functionally equivalent versions of its code by rewriting instructions and structure instead of merely changing an encrypted wrapper. Each generation can have a different byte pattern and file hash while preserving the same malicious purpose. This makes simple exact-signature detection less reliable.

Metamorphism is an evasion technique, not a separate payload objective. A metamorphic program can be a virus, worm, Trojan, backdoor, or another malware type. The term should describe verified code transformation, not every sample that is packed, obfuscated, frequently updated, or assigned several hashes.

01 Metamorphic vs polymorphic and oligomorphic malware

Technique	What changes	What tends to remain
Encrypted malware	Payload is encrypted on disk	Decryptor and decrypted body may be stable
Oligomorphic malware	Selects from a limited set of decryptors or variants	Encrypted payload behavior and a finite variant set
Polymorphic malware	Encryption, packing, keys, and decryptor mutate	Core payload commonly decrypts to similar code in memory
Metamorphic malware	The body itself is transformed or regenerated	Semantics, control goals, and external behavior
Ordinary packed malware	Representation is compressed or wrapped	Unpacked payload may be identical

Real samples can combine packing, polymorphism, metamorphism, server-side generation, and behavior changes. Analysts should describe the observed mechanism rather than forcing every family into one box.

02 How metamorphic code changes

- **Register reassignment:** equivalent registers are substituted where semantics allow.
- **Instruction substitution:** an operation is replaced with a different sequence producing the same result.
- **Code reordering:** independent blocks or functions are rearranged with control flow adjusted.
- **Dead-code insertion:** instructions that do not affect the result are added to alter structure.
- **Control-flow transformation:** branches, jumps, and block layout are rewritten.
- **Code expansion or shrinking:** simple instructions become longer equivalents or vice versa.
- **Recompilation:** a higher-level representation is emitted as a new binary.

A transformation engine must preserve the program's intended behavior. Bugs in the engine can create broken variants, while advanced engines use disassembly, intermediate representations, or compilation techniques to avoid corrupting instructions and data.

03 Why exact signatures and hashes are insufficient

A cryptographic hash identifies one exact byte sequence. If metamorphic rewriting changes the file, the hash changes even though the threat is related. A narrow byte signature can also disappear. This does not make the malware invisible: it still needs to execute, persist, communicate, access credentials, or perform another objective.

Family labels from different security vendors may disagree because one detects the packer, another the delivery chain, and another the behavior. Multiple names do not prove metamorphism.

04 How defenders detect metamorphic malware

- **Behavior monitoring:** detect injection, persistence, credential access, encryption, and command-and-control actions.

- **Normalized code analysis:** remove superficial differences and compare instructions or intermediate representations.
- **Control-flow and call-graph analysis:** compare program structure at a higher semantic level.
- **Emulation and sandboxing:** observe runtime behavior and unpacked or generated stages.
- **Memory scanning:** inspect code after decryption, unpacking, or execution.
- **Machine learning and similarity:** combine many static or dynamic features instead of one byte string.
- **Network and identity telemetry:** correlate infrastructure, protocols, accounts, and follow-on actions.

Every method has limits. Malware can detect sandboxes, delay execution, require user input, mimic legitimate tools, or change its infrastructure. Strong detection combines layers and retains context.

05 What to do after a detection

1. Isolate the device when execution, persistence, credential theft, remote control, or spread is possible.
2. Preserve the file, hash, path, process tree, memory evidence, user, timestamps, persistence, and network connections according to policy.
3. Quarantine identified components and scan with current engines, including memory and offline scanning when appropriate.
4. Hunt for behavior and infrastructure across endpoints instead of relying only on the original hash.
5. Revoke sessions and rotate credentials and keys exposed to the affected process or account.
6. Reimage systems when high-impact malware ran, persistence remains, or trustworthy eradication cannot be demonstrated.

06 How to reduce risk

- Patch operating systems, browsers, document software, and internet-facing services.
- Use endpoint detection that combines signatures, reputation, heuristics, memory, and behavior.
- Restrict execution from user-writable locations and apply application control.
- Use least privilege, network segmentation, and separate administrator identities.
- Filter malicious attachments, scripts, archives, and downloads.
- Maintain tested offline or immutable backups and an incident-response plan.

07 Frequently asked questions

Does metamorphic malware change its behavior?

Its implementation changes while the core malicious semantics remain. Campaign operators may separately add or remove capabilities across versions.

Is every changing malware sample metamorphic?

No. Server-side builds, recompilation by the author, packing, configuration changes, and polymorphic encryption can all produce new hashes without an internal metamorphic engine.

Can antivirus detect metamorphic malware?

Yes. Exact hashes are weak against it, but behavioral, structural, memory, reputation, network, and machine-learning detections can identify related activity.

08 Further reading

- Survey: Evolution and Detection of Polymorphic and Metamorphic Malware

- Oligomorphic, polymorphic, and metamorphic malware comparison

NEED MORE HELP?

We are here when the guide is not enough.

Visit the Gridinsoft Support Center for current product guidance or submit a support request with the details of your case.

Open this guide online at support.gridinsoft.com