Anton Belousov

Malware Analyst
Positive Technologies Expert
Security Center (PT ESC)

Who's the author? How does the automated malware attribution engine work

HITB Bangkok, Aug 29, 2024

Agenda

1. Why attribution is important
2. Attribution engine structure
3. Analytical modules and features used
4. Examples
5. Test results
6. Perspectives

Attribution approaches

What is it ?

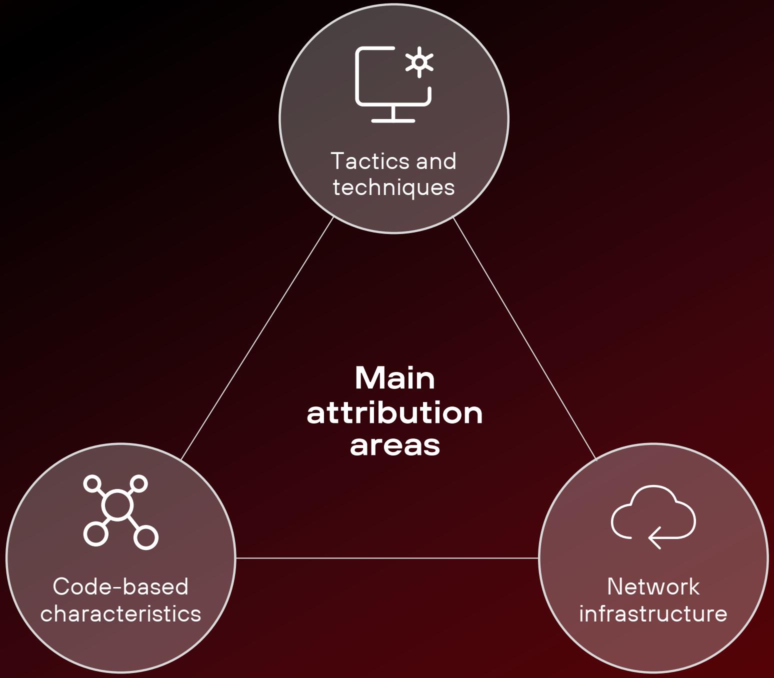
- Process of finding the authors and actors of cyber attacks
- Complex, fuzzy but helpful

Why is it important ?

- Increasing the efficiency of incident response
- Allowing for proactive protection

Why is the challenge ?

- High skill level
- Tons of non-obvious attributes



Main attribution areas

Tactics and techniques

Code-based characteristics

Network infrastructure

How to automate

Gather, extract, compare

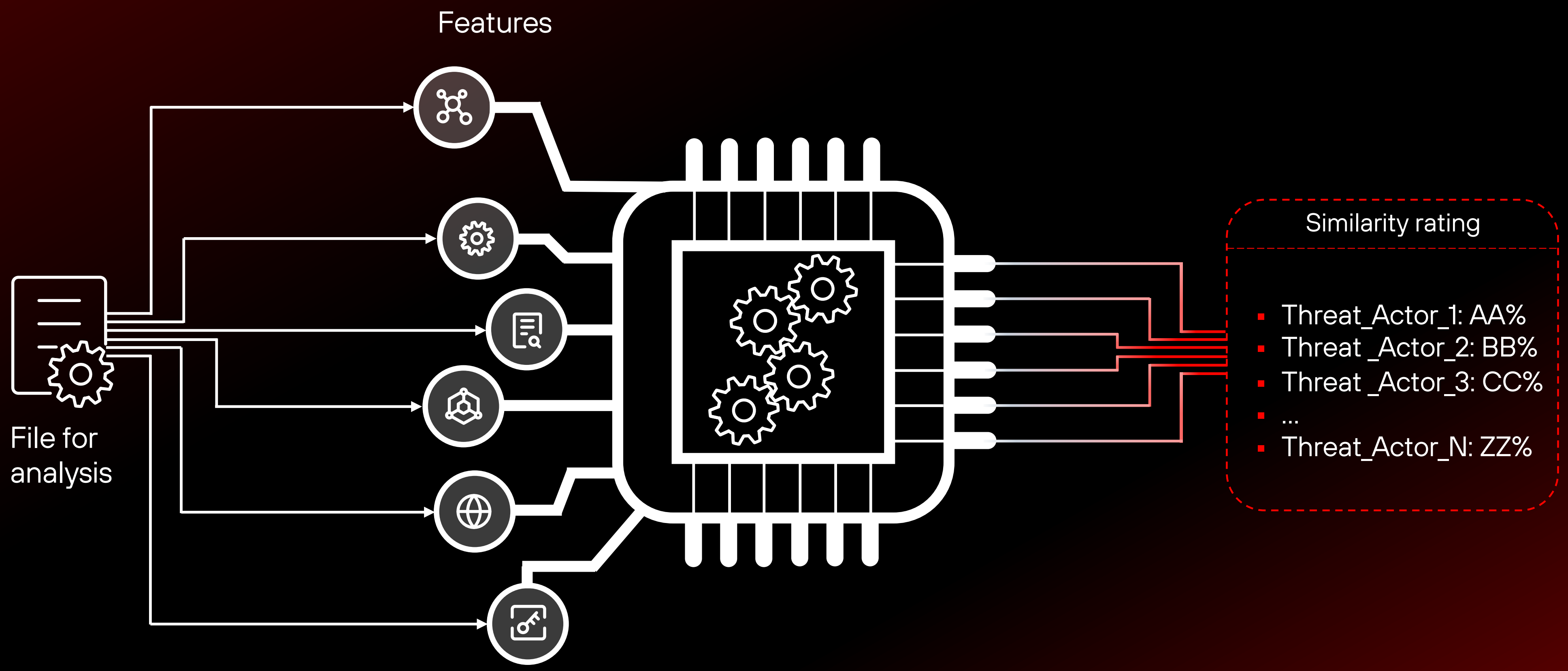
Gather attributed malware samples

- Analyze each file, extract features
- Save extracted data to DB

Obtain new malware sample

- Analyze it, extract features
- Compare with known files

Attribution engine architecture



- Threat_Actor_1: AA%
- Threat _Actor_2: BB%
- Threat _Actor_3: CC%
- ...
- Threat_Actor_N: ZZ%

Features

01

Basic file metadata

- Cryptographical hashes
- Fuzzy hashes
- Specific hashes (imphash, exphash, etc)

02

Code based features

- Unique code fragments
- Constants, magics, strings
- Control flow similarities

03

Tactics and techniques

- Suspicious API's
- Malware-specific code snippets

04

Dynamic analysis

- Behaviour-based detects
- Sandbox traces

Test sample

Main characteristics:

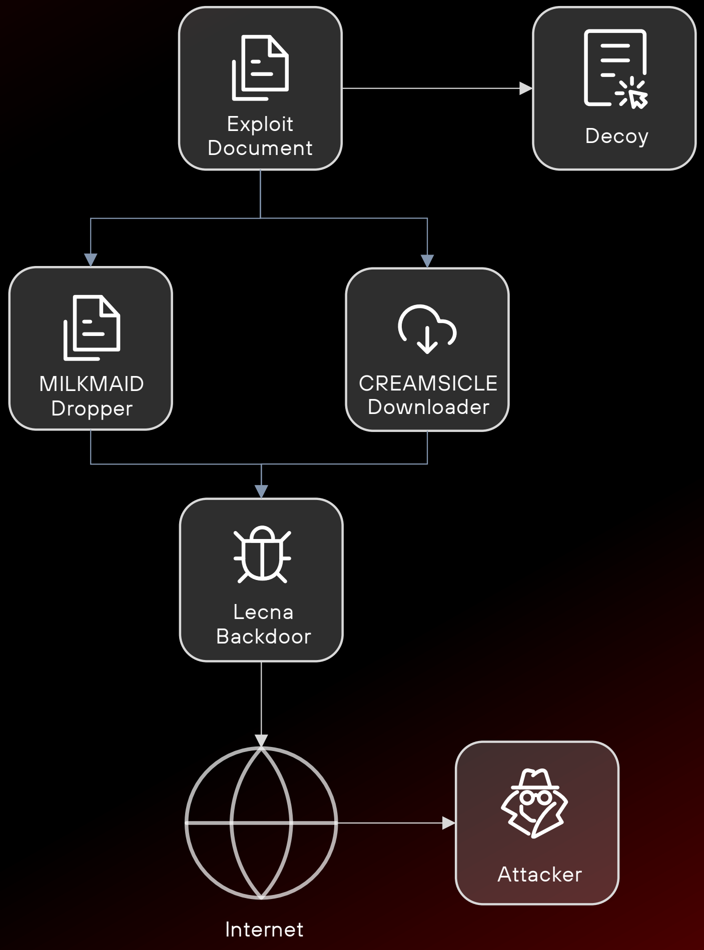
- Malware family: Lecna (BACKSPACE)
- Malware class: Backdoor
- Threat actor: APT30

Main techniques:

- Additional modules downloading
- Infected system manipulation
- System reconnaissance, information stealing
- Reverse-shell

SHA256: 017f4349170bd50e0abe565cd96ce7c65cf9a8308f76a20a0a7f391f73390012

APT30 and the mechanics of a long-running cyber espionage operation



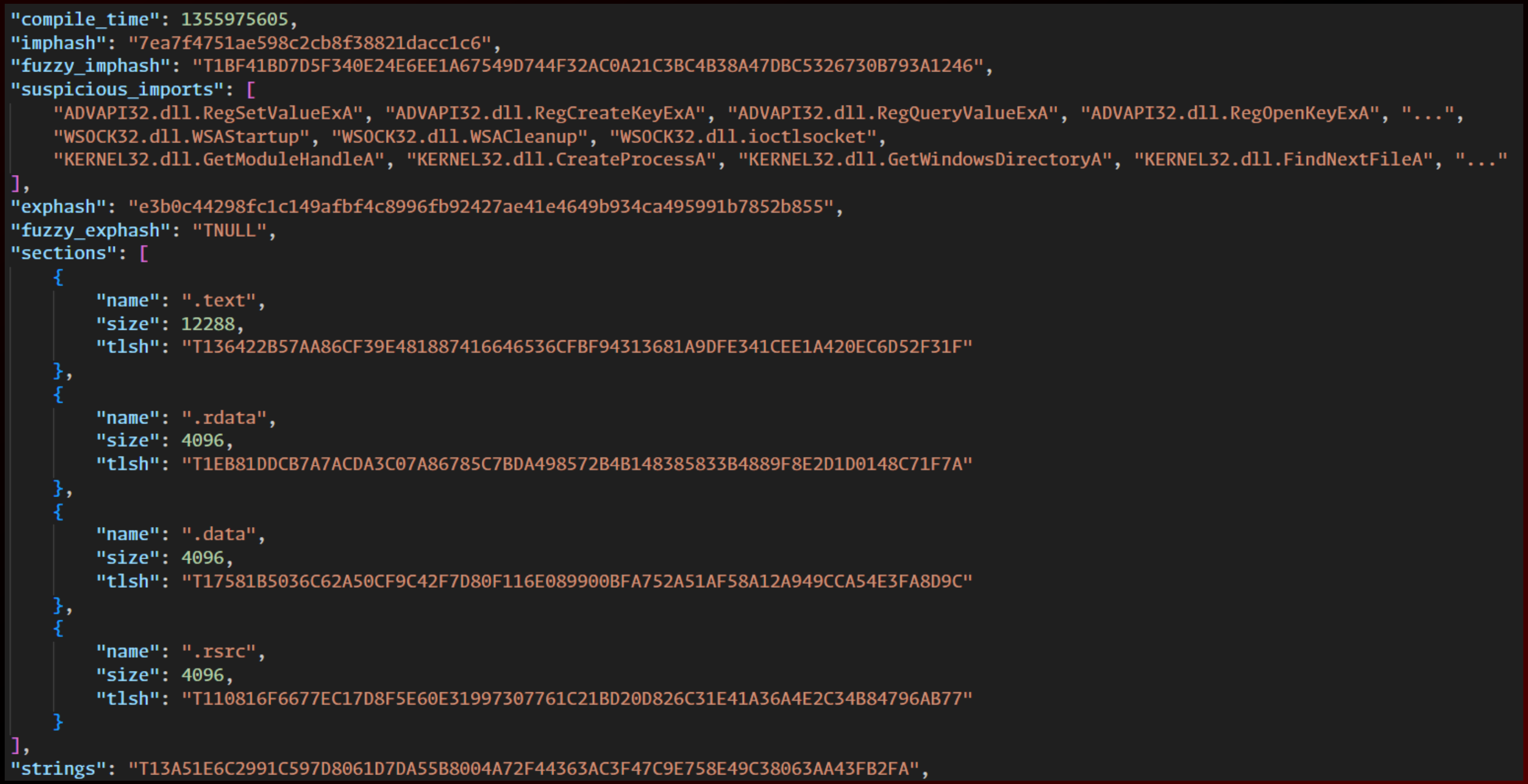
Basic static profile

Common file characteristics without deep diving:

- cryptographic hashes
- TLSH
- AV-detects
- embedded strings TLSH
- imphash, exphash TLSH
- section-wise TLSH
- resources TLSH

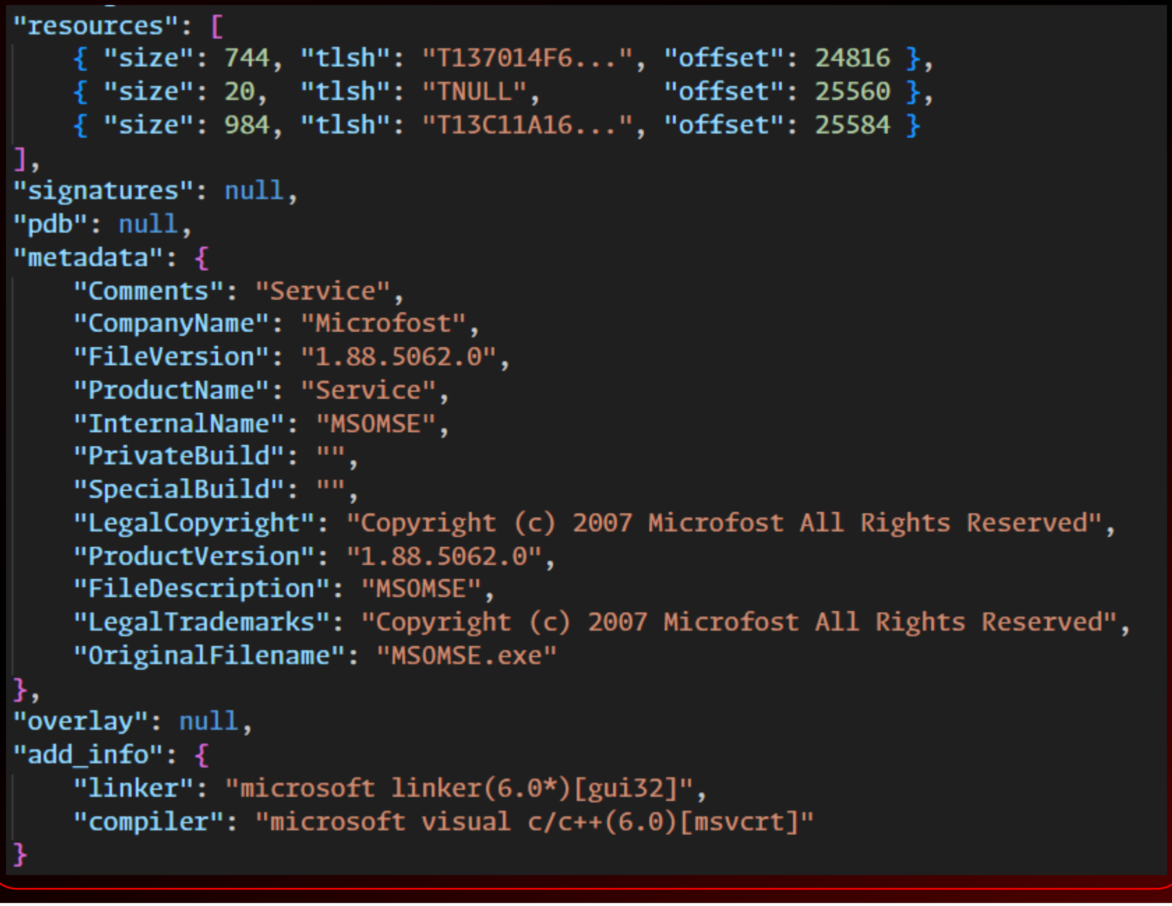
- metadata
- signatures
- suspicious imports
- overlay TLSH
- pdb-path
- compilation timestamp

Basic static profile example 1



```
"compile_time": 1355975605,
"imphash": "7ea7f4751ae598c2cb8f38821dacc1c6",
"fuzzy_imphash": "T1BF41BD7D5F340E24E6EE1A67549D744F32AC0A21C3BC4B38A47DBC5326730B793A1246",
"suspicious_imports": [
    "ADVAPI32.dll.RegSetValueExA", "ADVAPI32.dll.RegCreateKeyExA", "ADVAPI32.dll.RegQueryValueExA", "ADVAPI32.dll.RegOpenKeyExA", "...",
    "WSOCK32.dll.WSAStartup", "WSOCK32.dll.WSACleanup", "WSOCK32.dll.ioctlsocket",
    "KERNEL32.dll.GetModuleHandleA", "KERNEL32.dll.CreateProcessA", "KERNEL32.dll.GetWindowsDirectoryA", "KERNEL32.dll.FindNextFileA", "..."
],
"exphash": "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
"fuzzy_exphash": "TNULL",
"sections": [
    {
        "name": ".text",
        "size": 12288,
        "tlsh": "T136422B57AA86CF39E481887416646536CFBF94313681A9DFE341CEE1A420EC6D52F31F"
    },
    {
        "name": ".rdata",
        "size": 4096,
        "tlsh": "T1EB81DDCB7A7ACDA3C07A86785C7BDA498572B4B148385833B4889F8E2D1D0148C71F7A"
    },
    {
        "name": ".data",
        "size": 4096,
        "tlsh": "T17581B5036C62A50CF9C42F7D80F116E089900BFA752A51AF58A12A949CCA54E3FA8D9C"
    },
    {
        "name": ".rsrc",
        "size": 4096,
        "tlsh": "T110816F6677EC17D8F5E60E31997307761C21BD20D826C31E41A36A4E2C34B84796AB77"
    }
],
"strings": "T13A51E6C2991C597D8061D7DA55B8004A72F44363AC3F47C9E758E49C38063AA43FB2FA",
```

Basic static profile example 2



```
"resources": [
    { "size": 744, "tlsh": "T137014F6...", "offset": 24816 },
    { "size": 20,  "tlsh": "TNULL",        "offset": 25560 },
    { "size": 984, "tlsh": "T13C11A16...", "offset": 25584 }
],
"signatures": null,
"pdb": null,
"metadata": {
    "Comments": "Service",
    "CompanyName": "Microfost",
    "FileVersion": "1.88.5062.0",
    "ProductName": "Service",
    "InternalName": "MSOMSE",
    "PrivateBuild": "",
    "SpecialBuild": "",
    "LegalCopyright": "Copyright (c) 2007 Microfost All Rights Reserved",
    "ProductVersion": "1.88.5062.0",
    "FileDescription": "MSOMSE",
    "LegalTrademarks": "Copyright (c) 2007 Microfost All Rights Reserved",
    "OriginalFilename": "MSOMSE.exe"
},
"overlay": null,
"add_info": {
    "linker": "microsoft linker(6.0*)[gui32]",
    "compiler": "microsoft visual c/c++(6.0)[msvcrt]"
}
```

Tools mapping

Some malware families have already been linked to known threat actors so we can use this knowledge

- Try to determine where the new file belongs to known malware family using
  - PT Feeds
  - PT Sandbox
- Check if this malware family is used by any known threat actor

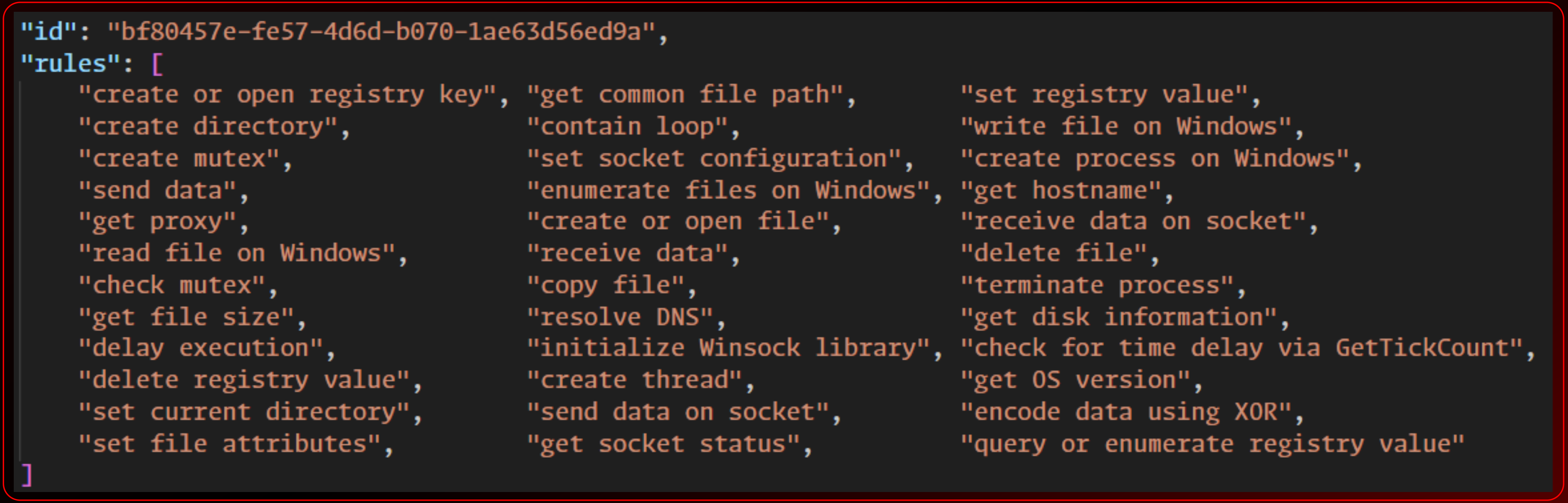
Static techniques profile

Analyze each file using heuristic engine to determine it's functionality.

Heuristic rules detect code snippets or some data signatures that implement different suspicious actions:

- Sandbox checking
- Obfuscation
- Anti-debugging, anti-disassembling techniques
- Network communication
- Host reconnaissance

Static techniques profile example



```
"id": "bf80457e-fe57-4d6d-b070-1ae63d56ed9a",
"rules": [
    "create or open registry key", "get common file path",       "set registry value",
    "create directory",            "contain loop",               "write file on Windows",
    "create mutex",                "set socket configuration",   "create process on Windows",
    "send data",                   "enumerate files on Windows", "get hostname",
    "get proxy",                   "create or open file",        "receive data on socket",
    "read file on Windows",        "receive data",               "delete file",
    "check mutex",                 "copy file",                  "terminate process",
    "get file size",               "resolve DNS",                "get disk information",
    "delay execution",             "initialize Winsock library", "check for time delay via GetTickCount",
    "delete registry value",       "create thread",              "get OS version",
    "set current directory",       "send data on socket",        "encode data using XOR",
    "set file attributes",         "get socket status",          "query or enumerate registry value"
]
```

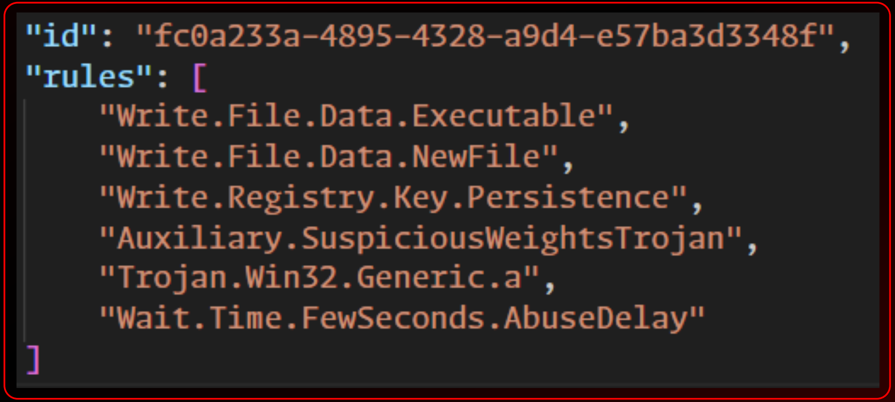
Dynamic techniques profile

Run the new file In PT Sandbox and collect all behavioral events generated by it.

Potentially interesting behavioral events:

- file system manipulation
- registry events
- process creation
- malware detects
- etc

Dynamic techniques profile example



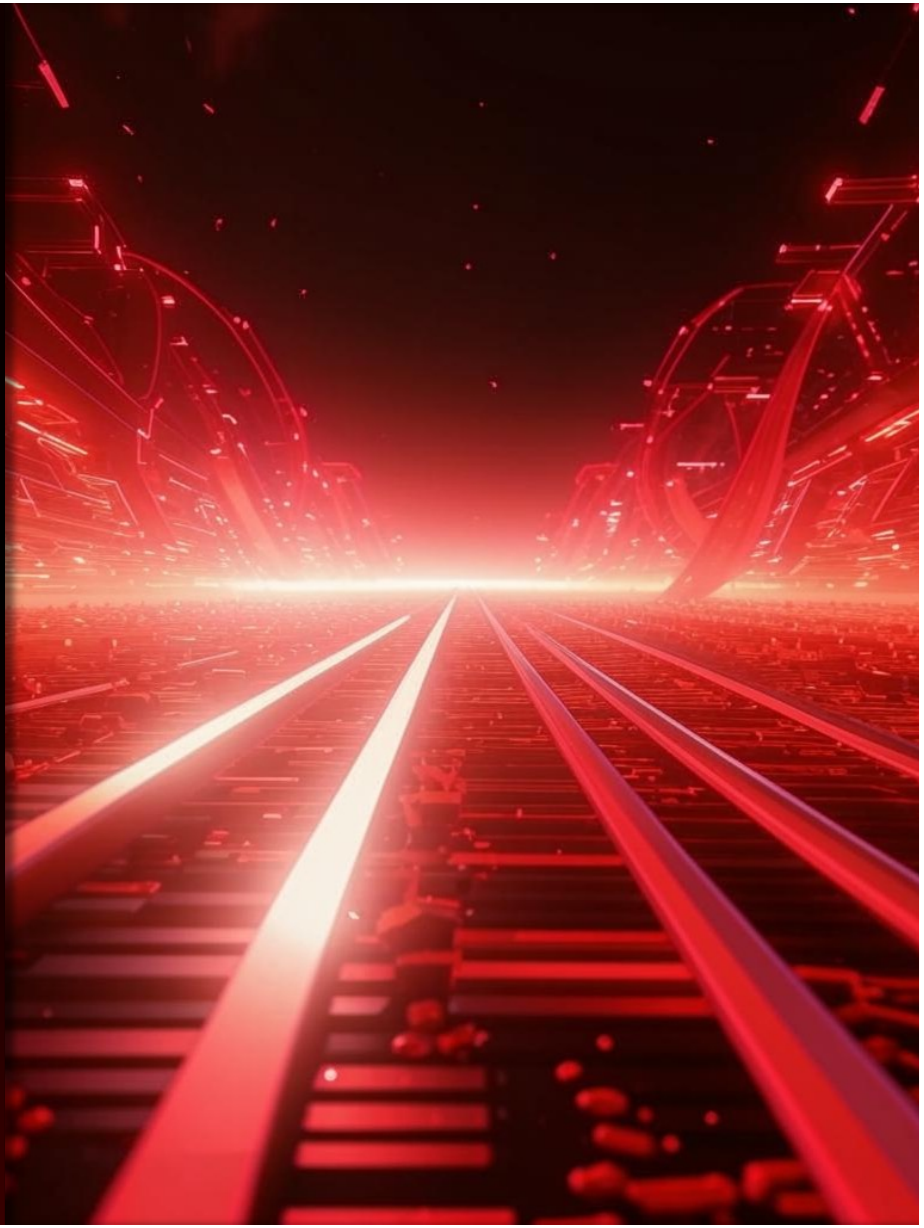
```
"id": "fc0a233a-4895-4328-a9d4-e57ba3d3348f",
"rules": [
    "Write.File.Data.Executable",
    "Write.File.Data.NewFile",
    "Write.Registry.Key.Persistence",
    "Auxiliary.SuspiciousWeightsTrojan",
    "Trojan.Win32.Generic.a",
    "Wait.Time.FewSeconds.AbuseDelay"
]
```

Traces profile

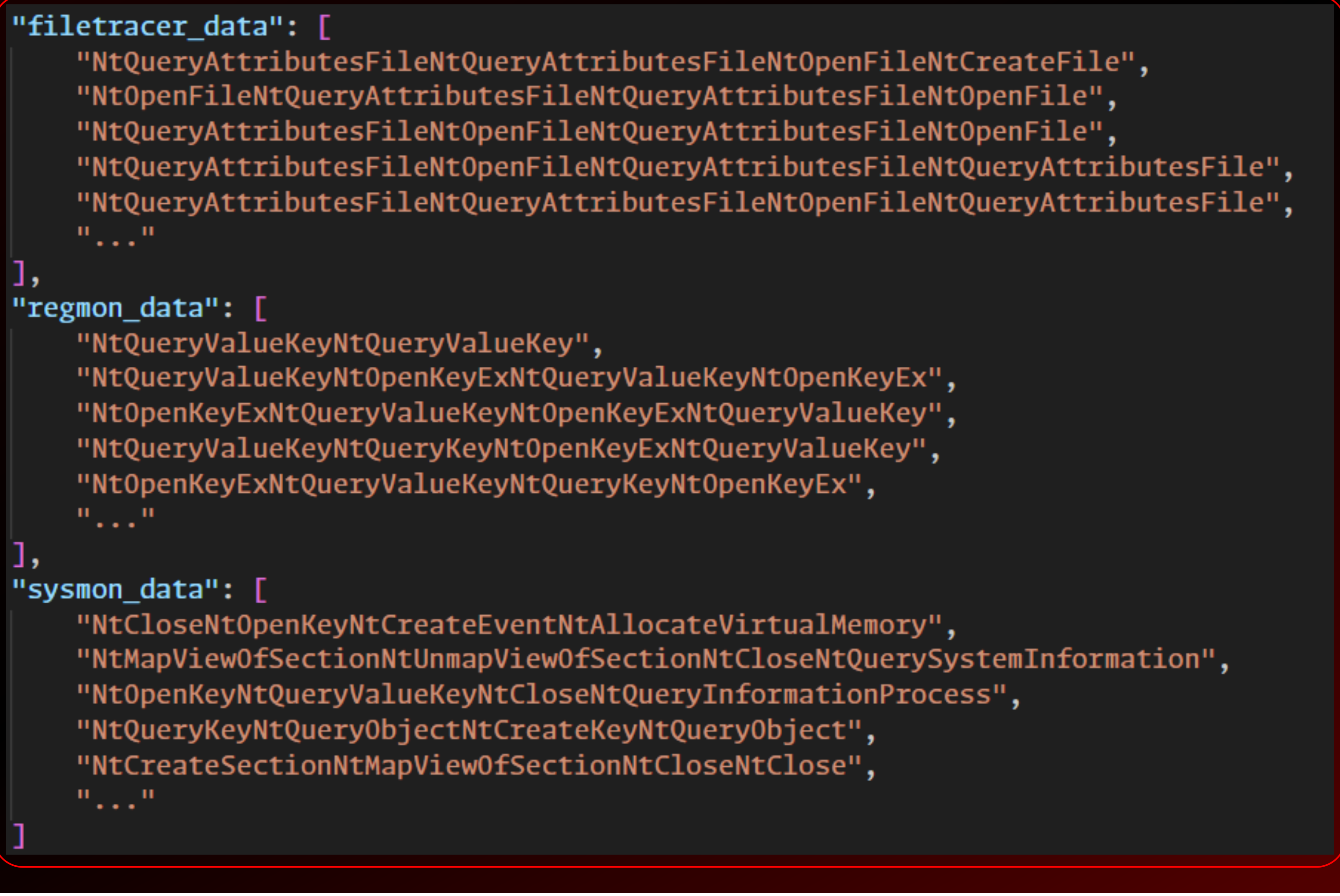
Capture the behavioral fingerprint, tracing every move during analysis in PT Sandbox

Use PT Sandbox execution traces and extract

- filesystem related API-calls
- registry related API-calls
- syscalls



Traces profile example

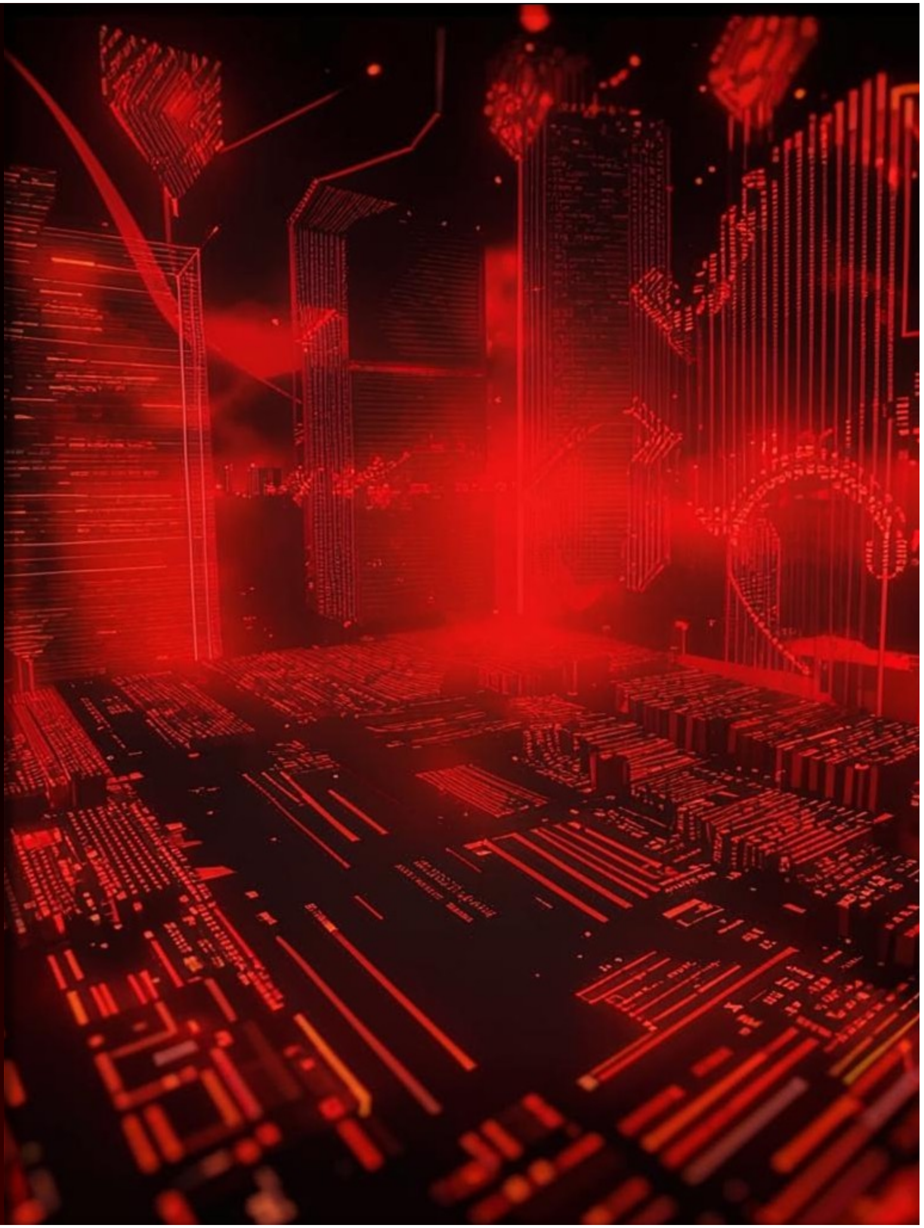


```
"filetracer_data": [
    "NtQueryAttributesFileNtQueryAttributesFileNtOpenFileNtCreateFile",
    "NtOpenFileNtQueryAttributesFileNtQueryAttributesFileNtOpenFile",
    "NtQueryAttributesFileNtOpenFileNtQueryAttributesFileNtOpenFile",
    "NtQueryAttributesFileNtOpenFileNtQueryAttributesFileNtQueryAttributesFile",
    "NtQueryAttributesFileNtQueryAttributesFileNtOpenFileNtQueryAttributesFile",
    "..."
],
"regmon_data": [
    "NtQueryValueKeyNtQueryValueKey",
    "NtQueryValueKeyNtOpenKeyExNtQueryValueKeyNtOpenKeyEx",
    "NtOpenKeyExNtQueryValueKeyNtOpenKeyExNtQueryValueKey",
    "NtQueryValueKeyNtQueryKeyNtOpenKeyExNtQueryValueKey",
    "NtOpenKeyExNtQueryValueKeyNtQueryKeyNtOpenKeyEx",
    "..."
],
"sysmon_data": [
    "NtCloseNtOpenKeyNtCreateEventNtAllocateVirtualMemory",
    "NtMapViewOfSectionNtUnmapViewOfSectionNtCloseNtQuerySystemInformation",
    "NtOpenKeyNtQueryValueKeyNtCloseNtQueryInformationProcess",
    "NtQueryKeyNtQueryObjectNtCreateKeyNtQueryObject",
    "NtCreateSectionNtMapViewOfSectionNtCloseNtClose",
    "..."
]
```

Genotypes

Genotype – byte sequence that represent certain code fragment that implement some suspicious or malicious functionality

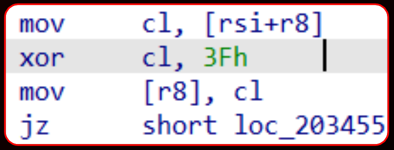
- Scan file using heuristic engine to find Points-of-interest
- Scopes of searching are basic block and function
- Gather basic block byte sequence as is and use it as plain genotype
- Convert plain genotype to template to make it more fuzzy



Genotypes

Plain

Instructions bytes as is

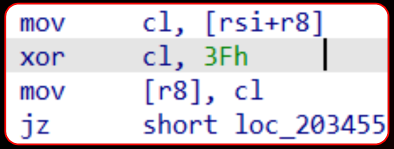


```
mov     cl, [rsi+r8]
xor     cl, 3Fh
mov     [r8], cl
jz      short loc_203455
```

42 8A 0C 06
80 F1 3F
41 88 08
74 0F

YARG

Mod R/M, SIB parametrization

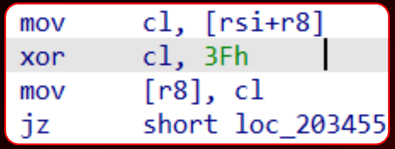


```
mov     cl, [rsi+r8]
xor     cl, 3Fh
mov     [r8], cl
jz      short loc_203455
```

4?8A(?4|?C)(0?|1?|2?|3?)
80F?3F
4?88(0?|1?|2?|3?)
74

XED

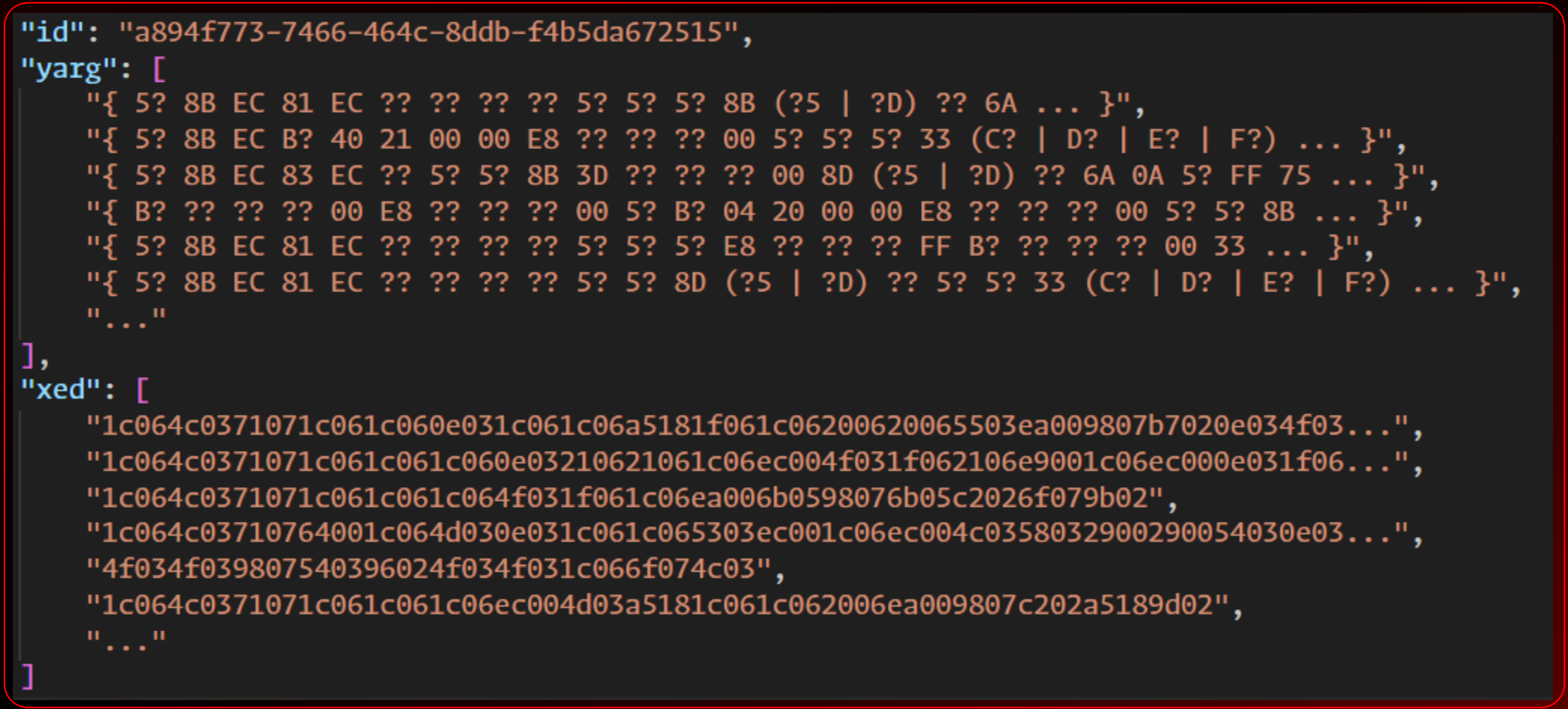
Instructions forms



```
mov     cl, [rsi+r8]
xor     cl, 3Fh
mov     [r8], cl
jz      short loc_203455
```

MOV_GPR8_MEMb
XOR_GPR8_IMMb_80r6
MOV_MEMb_GPR8
JZ_RELBRb

Genotypes example



```
"id": "a894f773-7466-464c-8ddb-f4b5da672515",
"yarg": [
    "{ 5? 8B EC 81 EC ?? ?? ?? ?? 5? 5? 5? 8B (?5 | ?D) ?? 6A ... }",
    "{ 5? 8B EC B? 40 21 00 00 E8 ?? ?? ?? 00 5? 5? 5? 33 (C? | D? | E? | F?) ... }",
    "{ 5? 8B EC 83 EC ?? 5? 5? 8B 3D ?? ?? ?? 00 8D (?5 | ?D) ?? 6A 0A 5? FF 75 ... }",
    "{ B? ?? ?? ?? 00 E8 ?? ?? ?? 00 5? B? 04 20 00 00 E8 ?? ?? ?? 00 5? 5? 8B ... }",
    "{ 5? 8B EC 81 EC ?? ?? ?? ?? 5? 5? 5? E8 ?? ?? ?? FF B? ?? ?? ?? 00 33 ... }",
    "{ 5? 8B EC 81 EC ?? ?? ?? ?? 5? 5? 8D (?5 | ?D) ?? 5? 5? 33 (C? | D? | E? | F?) ... }",
    "..."
],
"xed": [
    "1c064c0371071c061c060e031c061c06a5181f061c06200620065503ea009807b7020e034f03...",
    "1c064c0371071c061c061c060e03210621061c06ec004f031f062106e9001c06ec000e031f06...",
    "1c064c0371071c061c061c064f031f061c06ea006b0598076b05c2026f079b02",
    "1c064c03710764001c064d030e031c061c065303ec001c06ec004c035803290029005403 0e03...",
    "4f034f039807540396024f034f031c066f074c03",
    "1c064c0371071c061c061c06ec004d03a5181c061c062006ea009807c202a5189d02",
    "..."
]
```

Control flow graph profile

Analyze function’s CFG to extract graph-based features vector

- nodes types and quantity
- edges types and quantity
- xrefs types and quantity
- graph signature
- function’s instructions categories and forms
- imported functions used
- unique constants

Control flow graph profile

Nodes vector
Number of nodes of different types

Edges vector
Number of edges of different types

Xrefs vector
Number of xrefs (from, to)

Dominator tree signature

Instructions categories vector
Number of instructions XED categories

Filtering

Filter out incomparable pairs of functions

Decisive

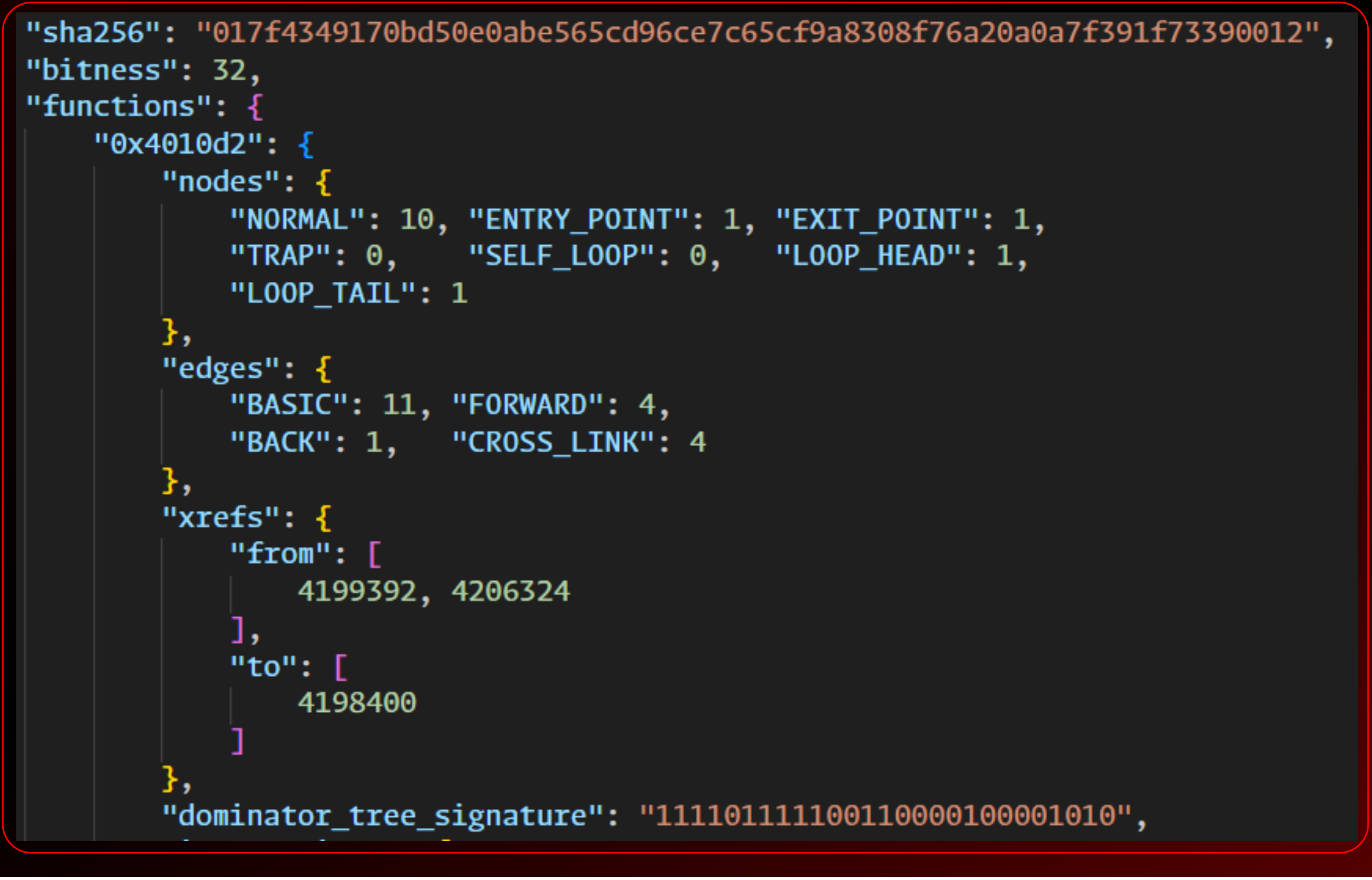
Features that decide pair of functions similarity degree

Constants
Set of unique or rare constants

Imports
List of imported API functions

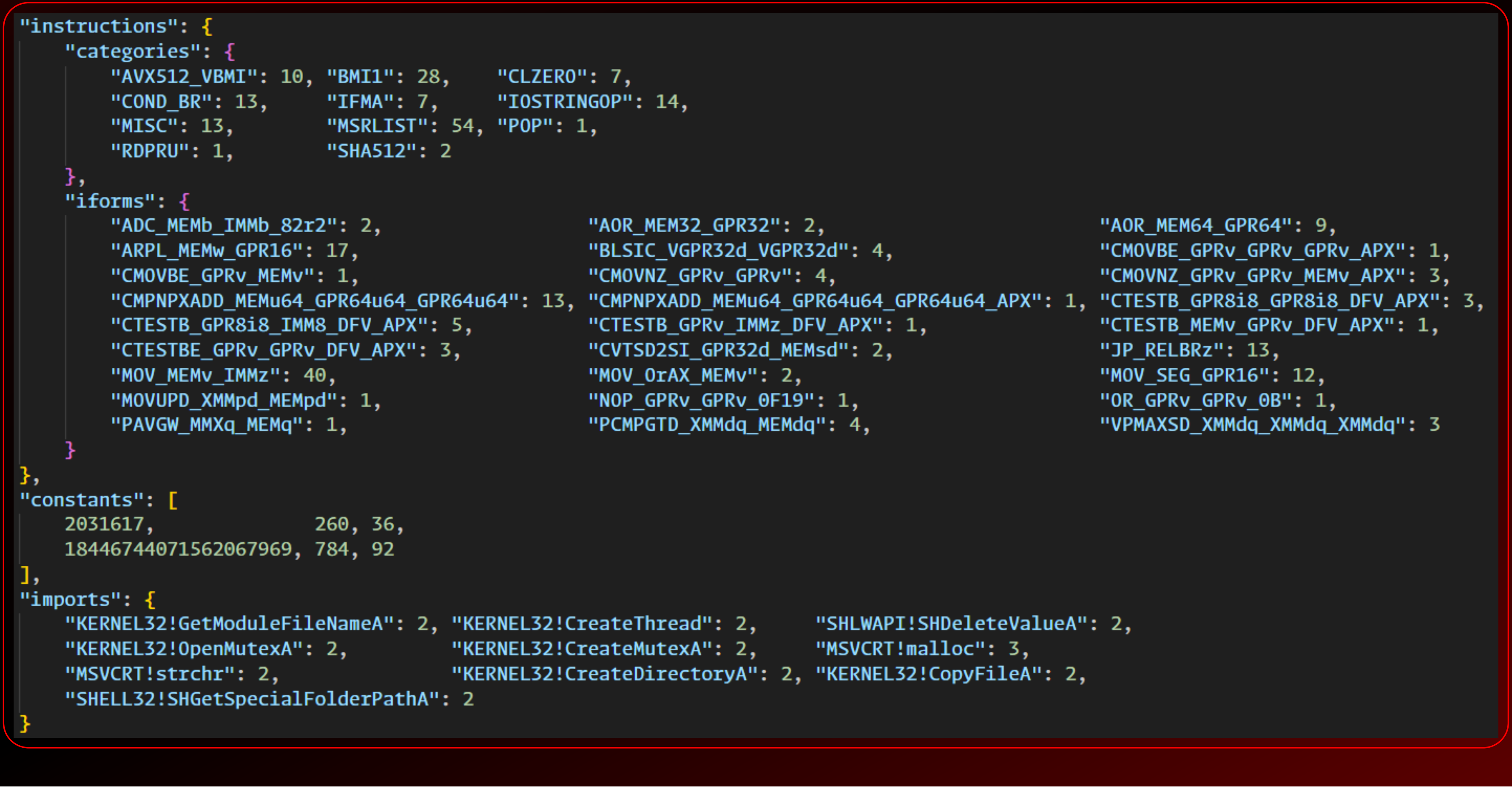
Instructions forms vector
Number of instructions XED forms

Control flow graph profile example 1



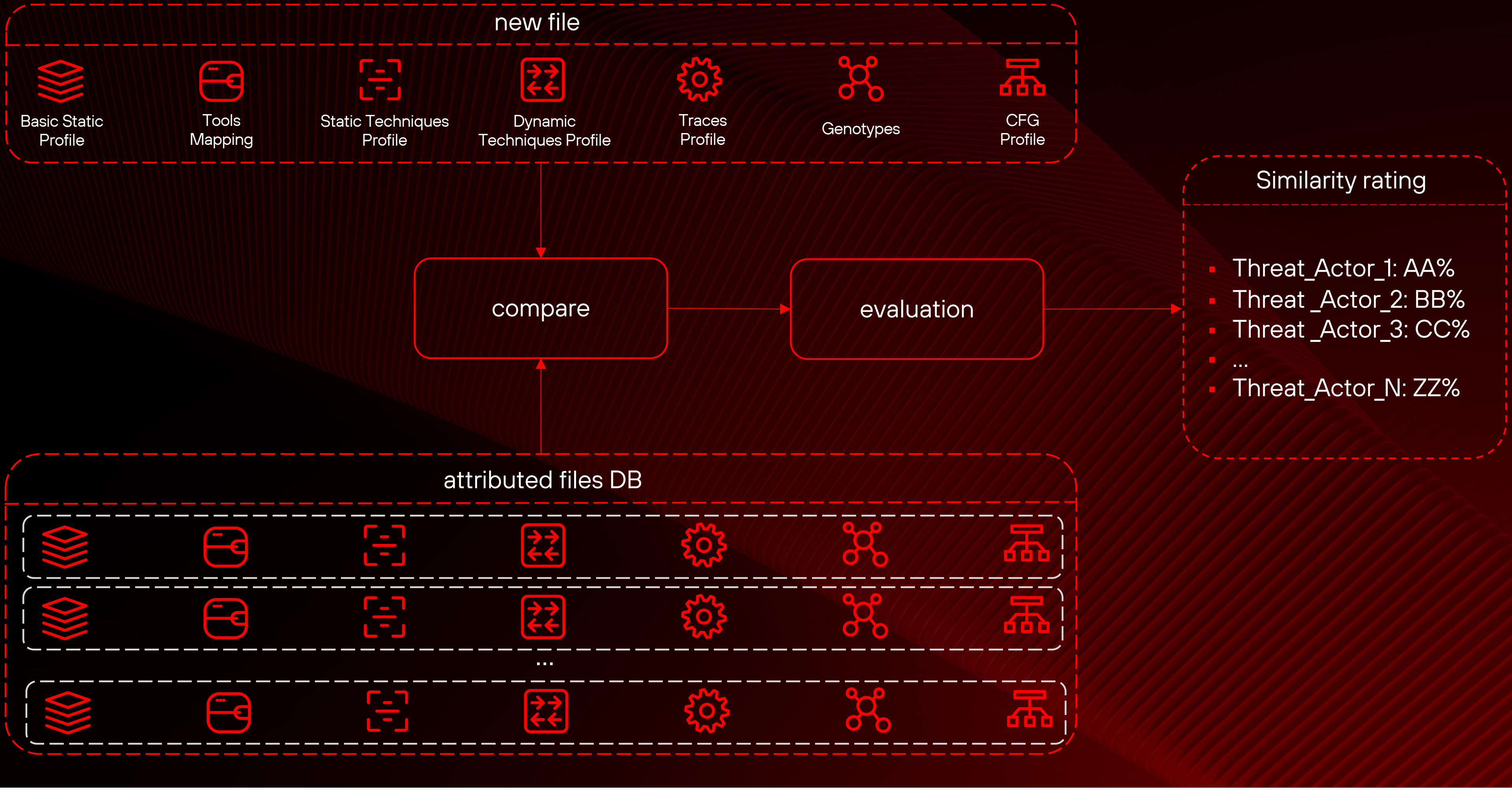
```
"sha256": "017f4349170bd50e0abe565cd96ce7c65cf9a8308f76a20a0a7f391f73390012",
"bitness": 32,
"functions": {
    "0x4010d2": {
        "nodes": {
            "NORMAL": 10, "ENTRY_POINT": 1, "EXIT_POINT": 1,
            "TRAP": 0,     "SELF_LOOP": 0,    "LOOP_HEAD": 1,
            "LOOP_TAIL": 1
        },
        "edges": {
            "BASIC": 11, "FORWARD": 4,
            "BACK": 1,    "CROSS_LINK": 4
        },
        "xrefs": {
            "from": [
                4199392, 4206324
            ],
            "to": [
                4198400
            ]
        },
        "dominator_tree_signature": "1111011111001100001000001010",
```

Control flow graph profile example 2



```
"instructions": {
    "categories": {
        "AVX512_VBMI": 10, "BMI1": 28,    "CLZERO": 7,
        "COND_BR": 13,     "IFMA": 7,     "IOSTRINGOP": 14,
        "MISC": 13,        "MSRLIST": 54, "POP": 1,
        "RDPRU": 1,        "SHA512": 2
    },
    "iforms": {
        "ADC_MEMb_IMMb_82r2": 2,                   "AOR_MEM32_GPR32": 2,                          "AOR_MEM64_GPR64": 9,
        "ARPL_MEMw_GPR16": 17,                     "BLSIC_VGPR32d_VGPR32d": 4,                    "CMOVBE_GPRv_GPRv_GPRv_APX": 1,
        "CMOVBE_GPRv_MEMv": 1,                     "CMOVNZ_GPRv_GPRv": 4,                         "CMOVNZ_GPRv_GPRv_MEMv_APX": 3,
        "CMPNPXADD_MEMu64_GPR64u64_GPR64u64": 13, "CMPNPXADD_MEMu64_GPR64u64_GPR64u64_APX": 1, "CTESTB_GPR8i8_GPR8i8_DFV_APX": 3,
        "CTESTB_GPR8i8_IMM8_DFV_APX": 5,           "CTESTB_GPRv_IMMz_DFV_APX": 1,                 "CTESTB_MEMv_GPRv_DFV_APX": 1,
        "CTESTBE_GPRv_GPRv_DFV_APX": 3,            "CVTSD2SI_GPR32d_MEMsd": 2,                    "JP_RELBRz": 13,
        "MOV_MEMv_IMMz": 40,                       "MOV_OrAX_MEMv": 2,                            "MOV_SEG_GPR16": 12,
        "MOVUPD_XMMpd_MEMpd": 1,                   "NOP_GPRv_GPRv_0F19": 1,                       "OR_GPRv_GPRv_0B": 1,
        "PAVGW_MMXq_MEMq": 1,                      "PCMPGTD_XMMdq_MEMdq": 4,                      "VPMAXSD_XMMdq_XMMdq_XMMdq": 3
    }
},
"constants": [
    2031617,              260, 36,
    18446744071562067969, 784, 92
],
"imports": {
    "KERNEL32!GetModuleFileNameA": 2, "KERNEL32!CreateThread": 2,    "SHLWAPI!SHDeleteValueA": 2,
    "KERNEL32!OpenMutexA": 2,         "KERNEL32!CreateMutexA": 2,    "MSVCRT!malloc": 3,
    "MSVCRT!strchr": 2,               "KERNEL32!CreateDirectoryA": 2, "KERNEL32!CopyFileA": 2,
    "SHELL32!SHGetSpecialFolderPathA": 2
}
```

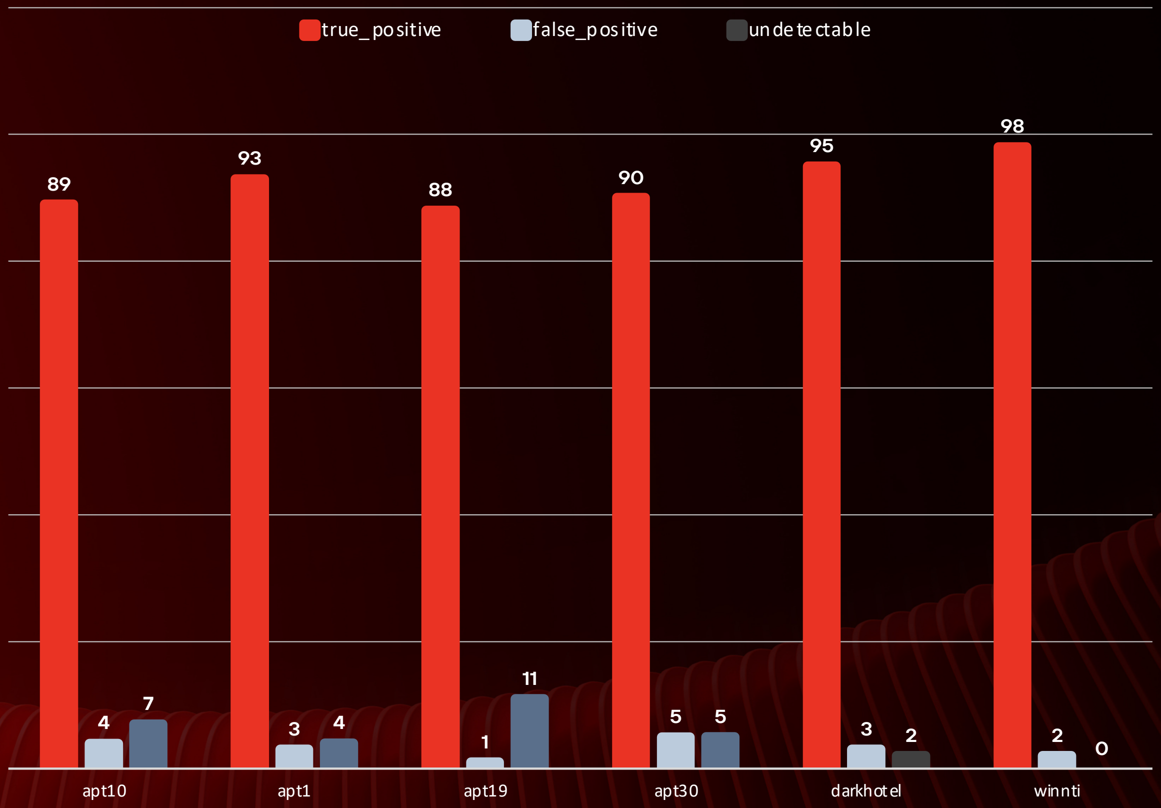
Comparison

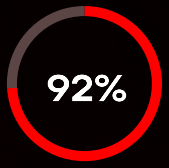


Similarity rating

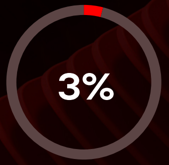
- Threat_Actor_1: AA%
- Threat _Actor_2: BB%
- Threat _Actor_3: CC%
- ...
- Threat_Actor_N: ZZ%

Test results

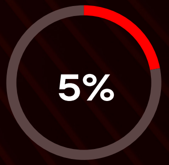




True Positive



False Positive



Undetectable

Conclusion

IoC's enrichment

- Understanding attack motivations and scope
- Anticipating and preparing for future attacks
- Prioritizing response efforts and resource allocation
- Developing targeted defense

Rating, not a prediction

- Nuanced and realistic approach
- Informed decision-making
- More effective IR

Modular architecture

- Maintenance
- Updates
- Substitutions
- Prioritization
- Scalability

Thank you!