manalyzer

A static analyzer for PE executables

#> whoami

- ACM&S International
  - @JusticeRage
  - Pentesting, reverse engineering, trainings…
- A few open-source projects on GitHub
  - ApkTrack
  - FreedomFighting
- Tor exit node operator
- Not CISSP

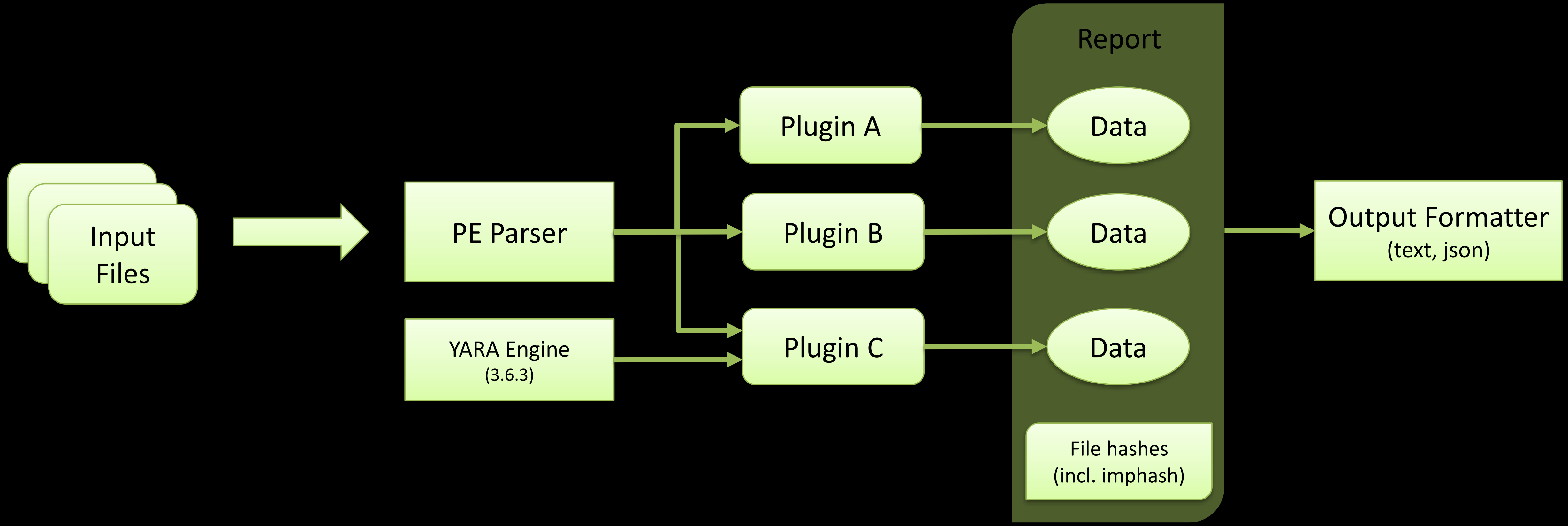
Project origins

- Started in Feb. 2014
- Annoyance at AV software’s opaque decisions
- Needed to automate repetitive tasks

Overview:

- A FOSS (GPLv3) tool written in C++ available on Windows / *nix
- Performs the initial assessment for unknown PE files
- Generates reports containing weak signals which hint at the file's behavior
- Developed for infosec professionals
- Static analysis only!

Architecture



PE Parser

- Design constraints: simple API, fast and robust
- PE files are complex
    - Microsoft’s documentation is cryptic at times
    - Windows’ loader is extremely lax
- Input files are untrusted and may try to fool the parser
    - See Reversing Labs’ “Undocumented PECOFF” talk from BH US 2011.

PE Parser – how robust is it, really?

- Fuzzed for ≈2 months with AFL - no crashes
    - Input files: Ange Albertini's handcrafted PEs
- Manalyze's bug bounty
    - Send me a sample which triggers a crash, get 100€
    - Not paying for crashes in third-party libraries
    - More details at https://manalyzer.org/bounty
    - Tip: fuzz with -fsanitize=address
- Speed: went through a VirusShare release (≈68 Go) in 10 minutes
    - Caveat: many non-PE files were rejected early on
    - Caveat: all plugins were turned off

YARA

- A pattern searching tool written by Victor M. Alvarez (VirusTotal)
- Slightly modified version included in Manalyze
  - Code stripped down to a library
  - Added C++ wrappers
  - Replaced the PE-format awareness plugin

ClamAV Plugin

- ClamAV signatures without ClamAV
- Signature files are converted to YARA rules
- Signatures are NOT distributed with Manalyze
    - A Python script is provided to download and translate them
- Caveat: .hdb and .mdb databases are not imported

Resource analysis plugin

- Analyzes files contained in the PE
  - High entropy resources may be compressed/encrypted.
  - PE is 75% resources and/or contains another PE? Possibly a dropper.
- Resources can be extracted by the parser for further inspection

“PEiD” plugins

- Apply PEiD signatures
- Public PEiD signatures translated to YARA rules and spread over two plugins:
  - Compiler detection
  - Packer detection
- PEiD is not maintained anymore :(

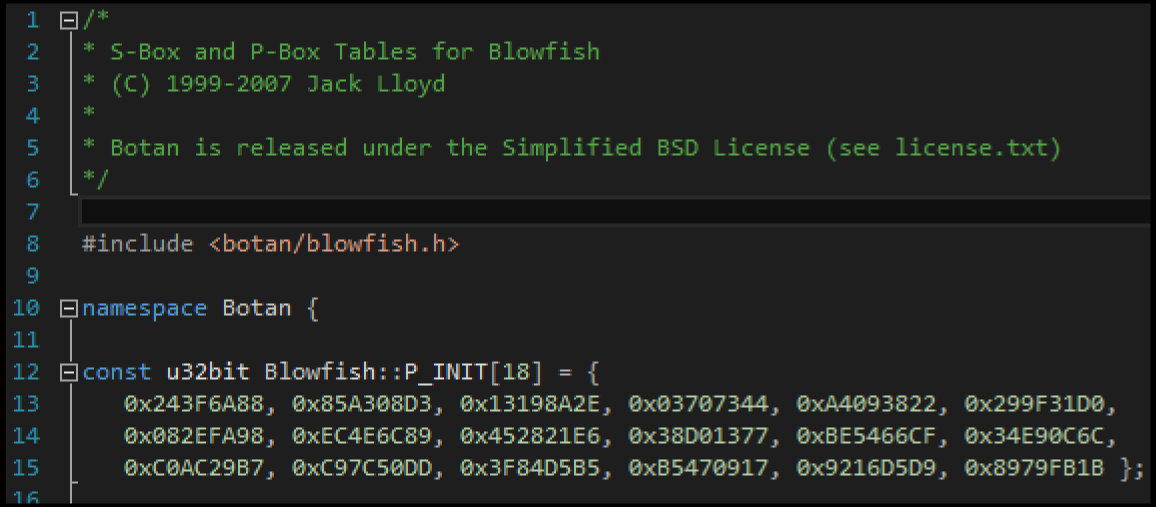
Strings plugin

- Looks for suspicious strings in the file
  - References to system tools (i.e. regedit.exe, taskmgr.exe, etc.)
  - References to specific registry keys and the WMI
  - Debugger and/or AV process names
  - VM detection techniques
  - etc.

Cryptography detection plugin

- Look for cryptographic constants used by well-known ciphers

- Methodology: download a cryptographic library and look for things like this:



```
/*
* S-Box and P-Box Tables for Blowfish
* (C) 1999-2007 Jack Lloyd
*
* Botan is released under the Simplified BSD License (see license.txt)
*/

#include <botan/blowfish.h>

namespace Botan {

const u32bit Blowfish::P_INIT[18] = {
   0x243F6A88, 0x85A308D3, 0x13198A2E, 0x03707344, 0xA4093822, 0x299F31D0,
   0x082EFA98, 0xEC4E6C89, 0x452821E6, 0x38D01377, 0xBE5466CF, 0x34E90C6C,
   0xC0AC29B7, 0xC97C50DD, 0x3F84D5B5, 0xB5470917, 0x9216D5D9, 0x8979FB1B };
```

- Detected: MD5, SHA(1|256|512), AES, DES, RC(5|6), Blowfish, Twofish, Whirlpool, Tiger, Camellia

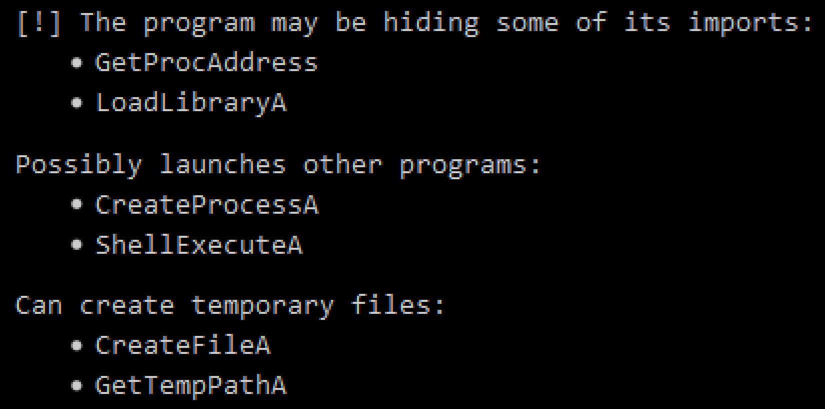
Packer detection plugin

- Heuristics to detect packed executables
    - Contains a white-list of section names
    - Checks that the number of imports is reasonable
    - Looks for high entropy / WX sections
    - Miscellaneous inconsistencies caused by some packers

Import analysis plugin

- Tries to infer the program's behavior based on imported functions
  - VirtualAlloc + WriteProcessMemory + CreateRemoteThread = BAD
  - Networking functions
  - Process, Service and Registry manipulation APIs
  - Functions which can be used for anti-debugging purposes
  - Code injection: Process Hollowing, PowerLoader, Atom Bombing…

- Guess what this one does:



```
[!] The program may be hiding some of its imports:
    • GetProcAddress
    • LoadLibraryA

Possibly launches other programs:
    • CreateProcessA
    • ShellExecuteA

Can create temporary files:
    • CreateFileA
    • GetTempPathA
```

Bitcoin plugin

- Looks for Bitcoin addresses contained in the binary.
- The address' structure is validated to eliminate false positives.

```
LegalCopyright:     © Microsoft Corporation. All rights reserved.
OriginalFilename:   diskpart.exe
ProductName:        Microsoft® Windows® Operating System
ProductVersion:     6.1.7601.17514

[ MALICIOUS ] This program may be a ransomware.
    Contains a valid Bitcoin address:
        115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn
        12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
        13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94
```

Authenticode plugin

- Verifies the digital signature of the PE (if any)
- Unsigned binary claiming to come from Adobe / Oracle / Google?
  - Raise an alert.
- Only available on Windows
- *nix version in the works
  - Relies on OpenSSL
  - Can display the certificate’s issuer, but doesn’t check the signature yet.
  - Problem: do I need to bundle Windows’ trusted certs with Manalyze?
  - Problem: Microsoft executables are known by hash in the security catalog.

VirusTotal plugin

- Get AV detection results from VirusTotal
  - Only the file hash is submitted.
  - File was never seen by VT ? Suspicious.
- Caveat: registration on virustotal.com is required to obtain an API key.

manalyzer.org

- Web portal created to use the tool online
- Submit a file (or link to a file) to have it analyzed
- Access existing reports from the command-line
  - `curl https://manalyzer.org/json/539f8f30c06967919b5d508198b70fbe`
- Samples are not shared with anyone
- You don’t have to trust me: build manalyze and run it locally

Usability & reusability

- No headaches while building the tool:
  - `apt-get install [dependencies]`
  - `git clone https://github.com/JusticeRage/Manalyze.git`
  - `cmake . && make`
- You don't have to read the source code to write plugins
- Need a PE parser for another project? Just copy some .cpp files and reuse this one.
  - Find instructions at https://docs.manalyzer.org
  - Blog post: using the PE parser to implement process hollowing at https://blog.Kwiatkowski.fr/?q=en/process_hollowing

Future works

- Authenticode plugin on *nix
- Icon recognition plugin
- Resolve dynamic imports with Capstone Engine
- Integrate a search engine (Solr / ElasticSearch) on manalyzer.org
- Python bindings

Demonstration

?