Obfuscation

and other software White Boxes

A small primer on how secrets* might be concealed

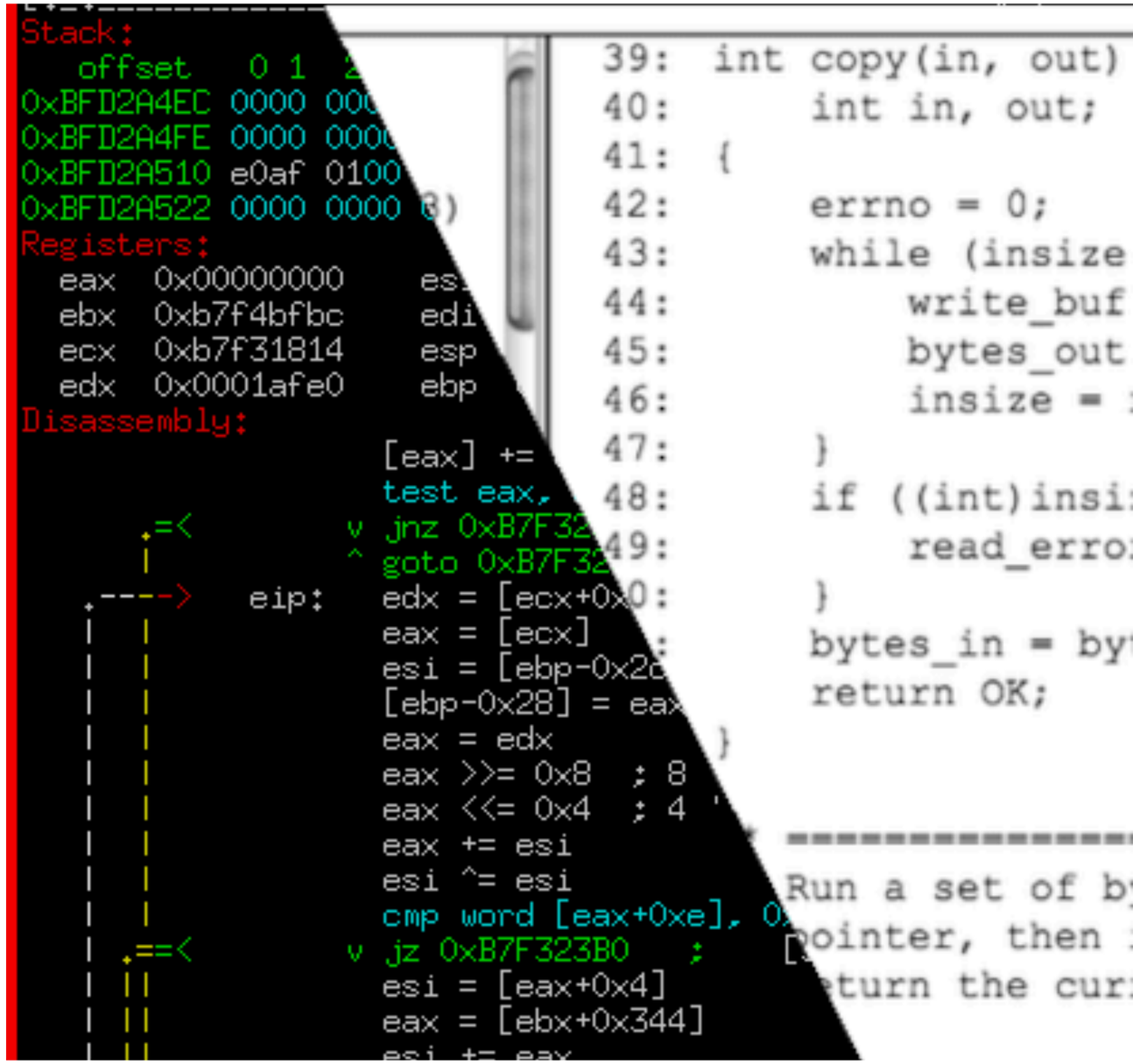
Mario Del Gaudio - RoadToNohat 2022-02-07

White Box

A system that can be freely analyzed

- Program w/source code
- HW with JTAG debugging
- Obfuscated code
  - Licensing DRM
  - Malware
  - Proprietary algorithms
- Traditional Media DRM

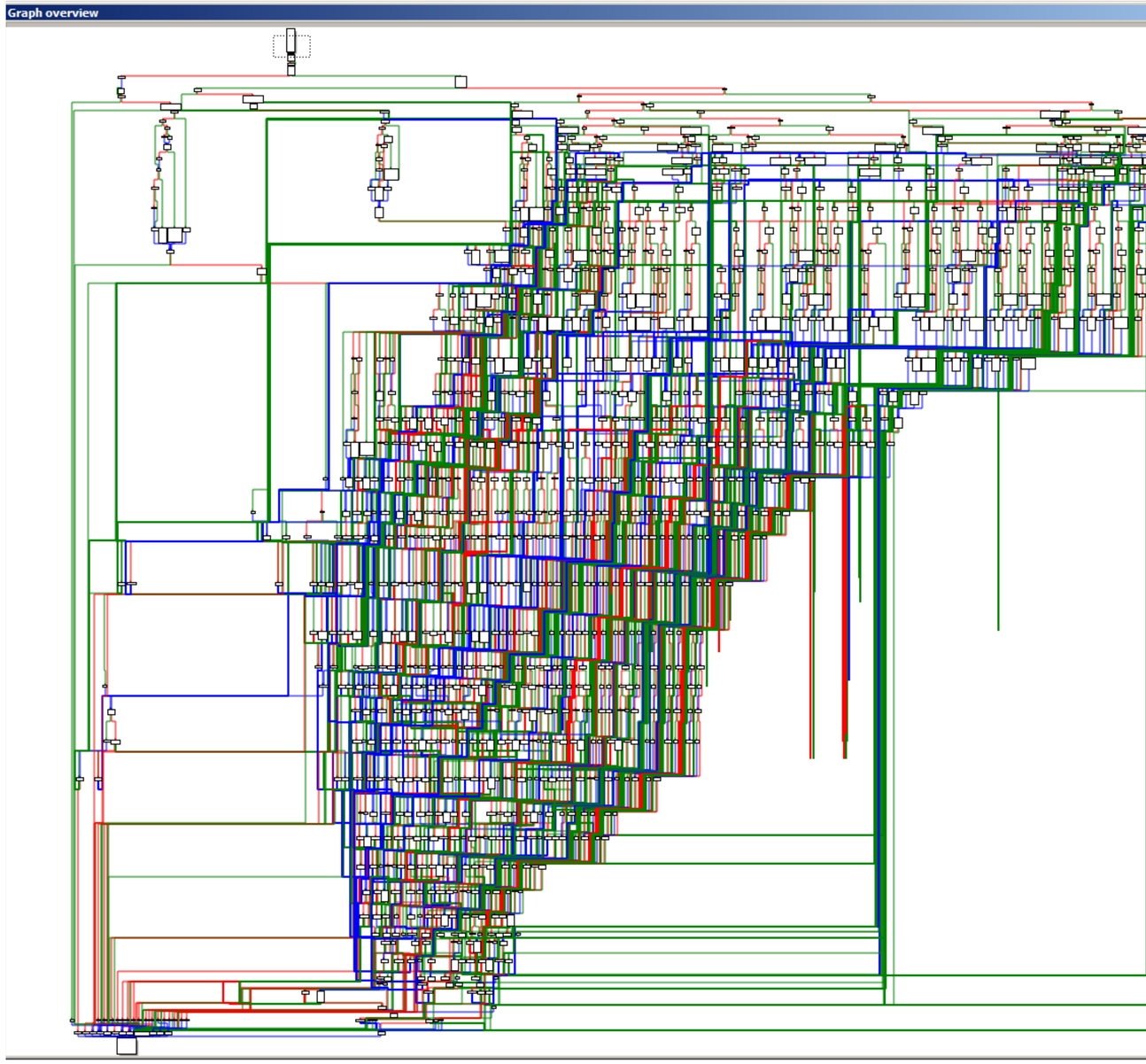
Basically everything your main CPU runs



Obfuscated machine code

Techniques

- Encoding, encryption, packing
- Function Level Encryption
- Virtual Machines
- Complexity
  - Opaque predicates
  - Dead code
  - Self modifying code



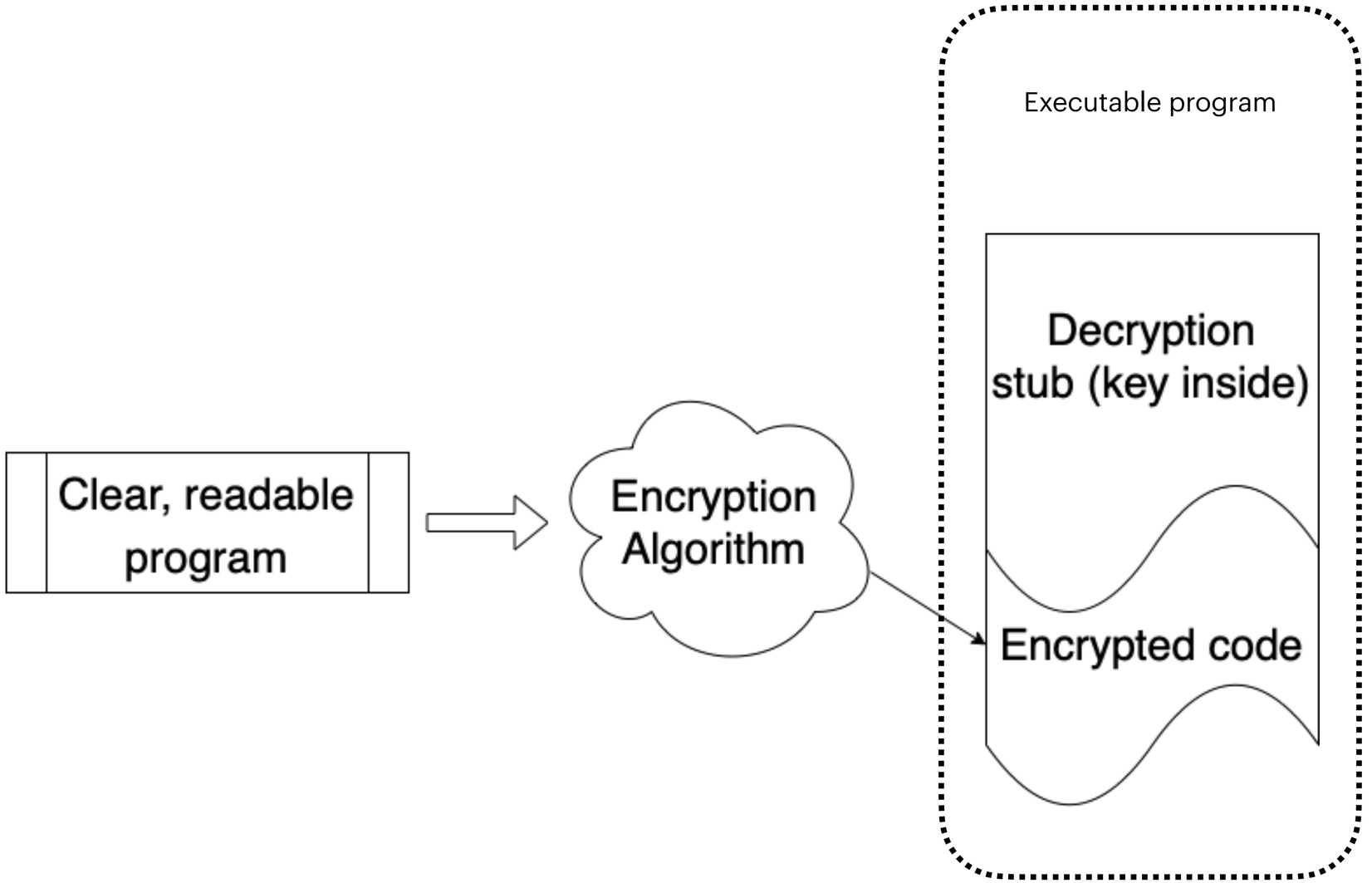
Code concealing

The whole program is transformed, so that static analysis isn't possible.

The main goal is to make code unreadable until it is executed, and this is accomplished using non-trivial encryption algorithms with keys embedded in the binary, or using static encoding or compression algorithms

Encryption

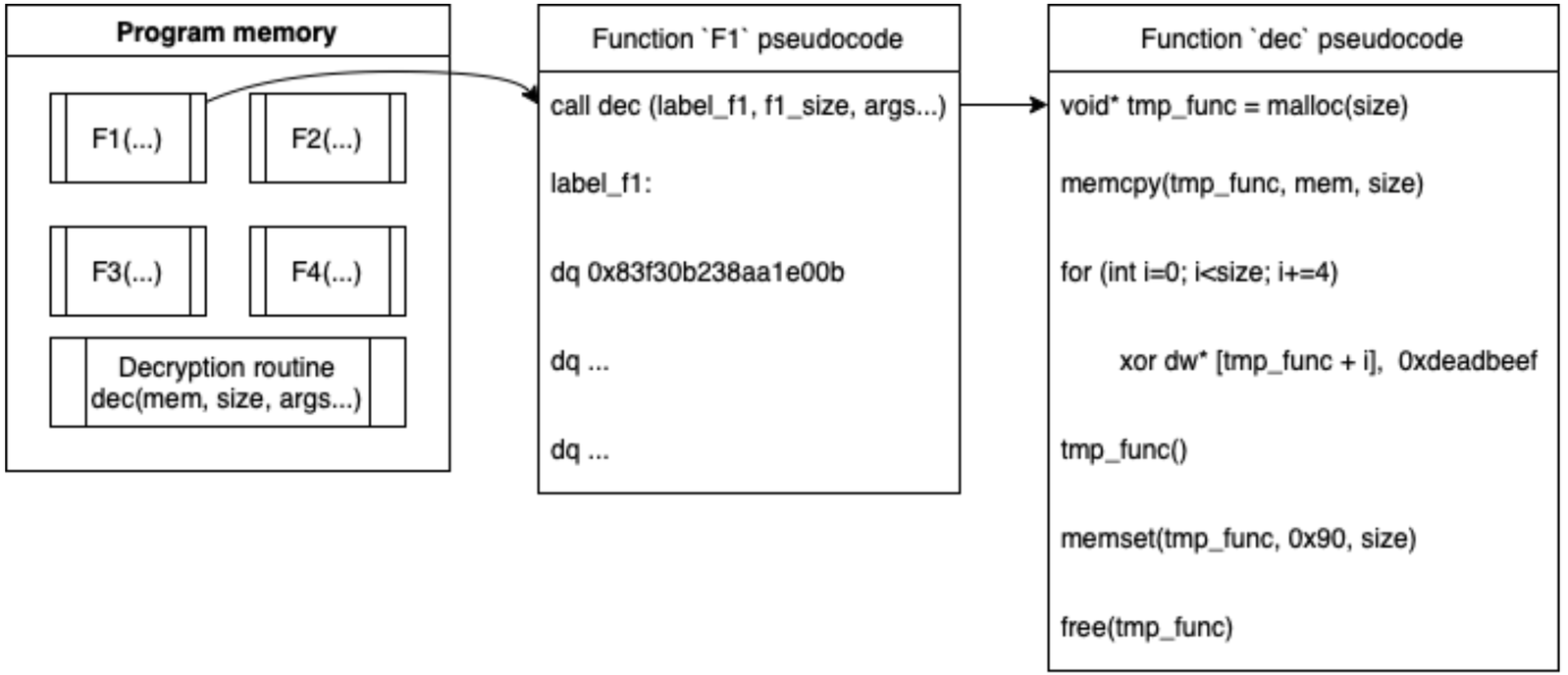
But this diagram is valid for encoding and packing as well



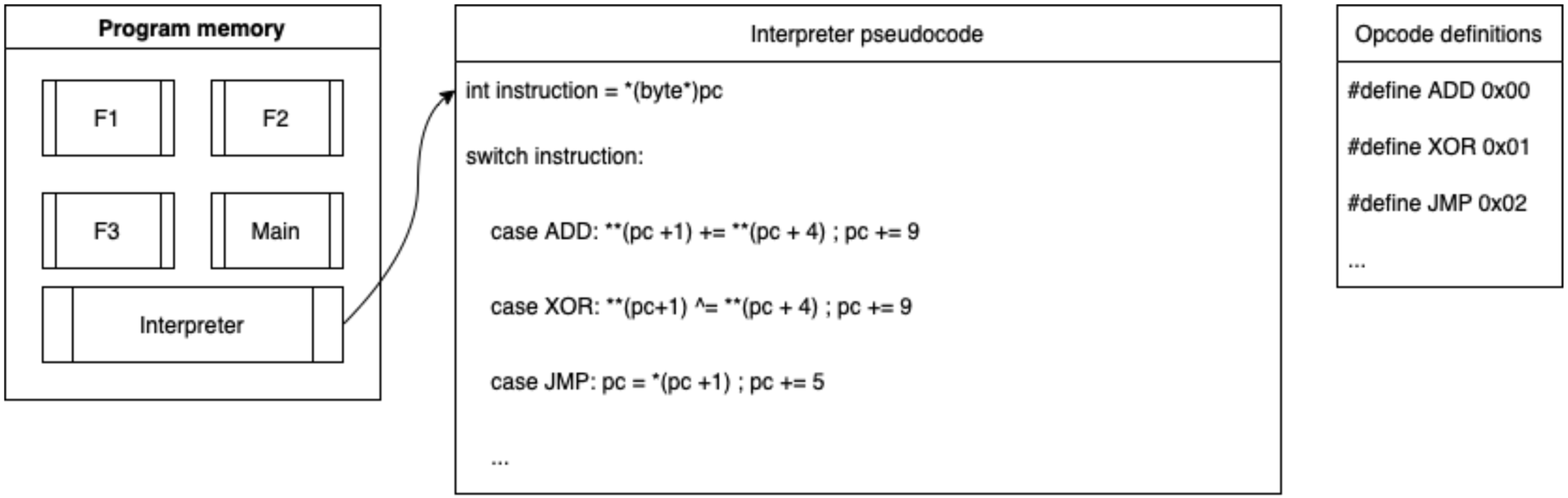
This can be easily defeated by dumping program memory while running to get the cleartext code

Function-level encryption

The goal of this is to never have the whole program unencrypted in memory



This is trickier: to be able to do static analysis you need to write an unpacker which decodes each function.



This diagram shows an interpreted VM: native machine code is never visible. The bytecode can be analyzed by understanding every single custom opcode used in it.

JIT compiled VMs on the other hand are similar to function-level encryption, because code is directly converted to machine language before it is executed, thus cleaner automated unpacking is possible.

Virtual machines

Virtual machines are used to add an obfuscation layer by defining an alternate machine language for all instructions, then JIT compiling or interpreting them.

Virtual machines are the most difficult of the bunch when it comes to reverse engineering

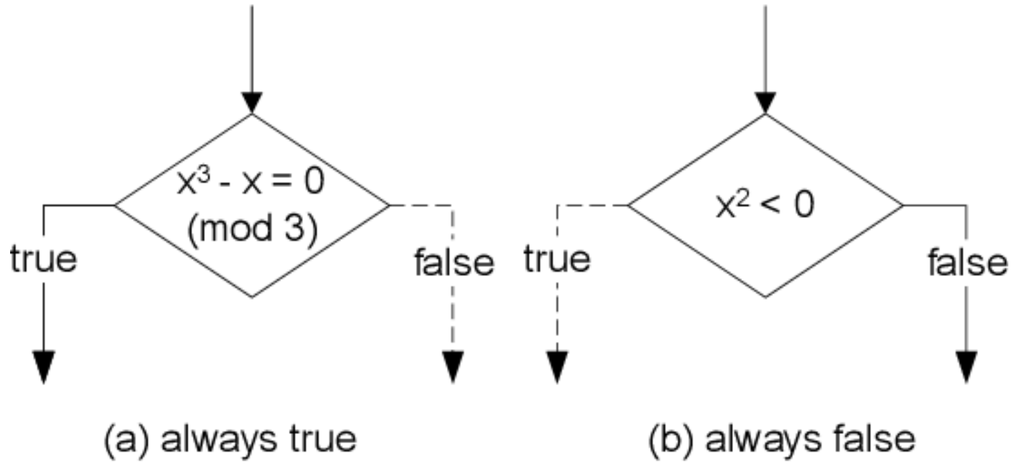
Complexity

Adding complexity to the code is just as effective in wasting reverse engineers' time.

This is accomplished by using less known and more complex machine instructions, adding unused code, branches and calculations.

Opaque predicates

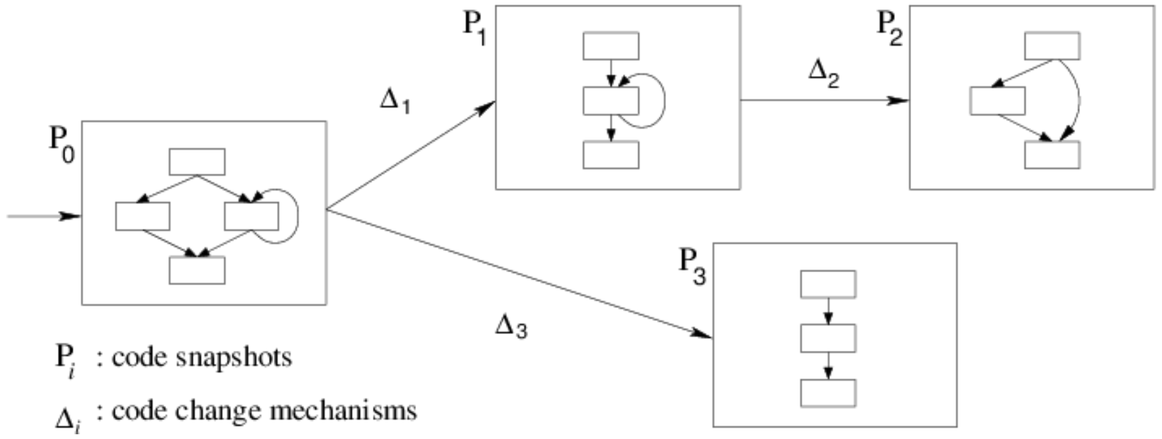
An expression that always evaluates the same way (known to the creator)



Opaque predicates are closely related to dead code; dead code is code that never gets executed.

Self modifying code

To further complicate static analysis, code can be modified while running, thus rendering invalid a traditional Control Flow Graph drawn from disassembled code



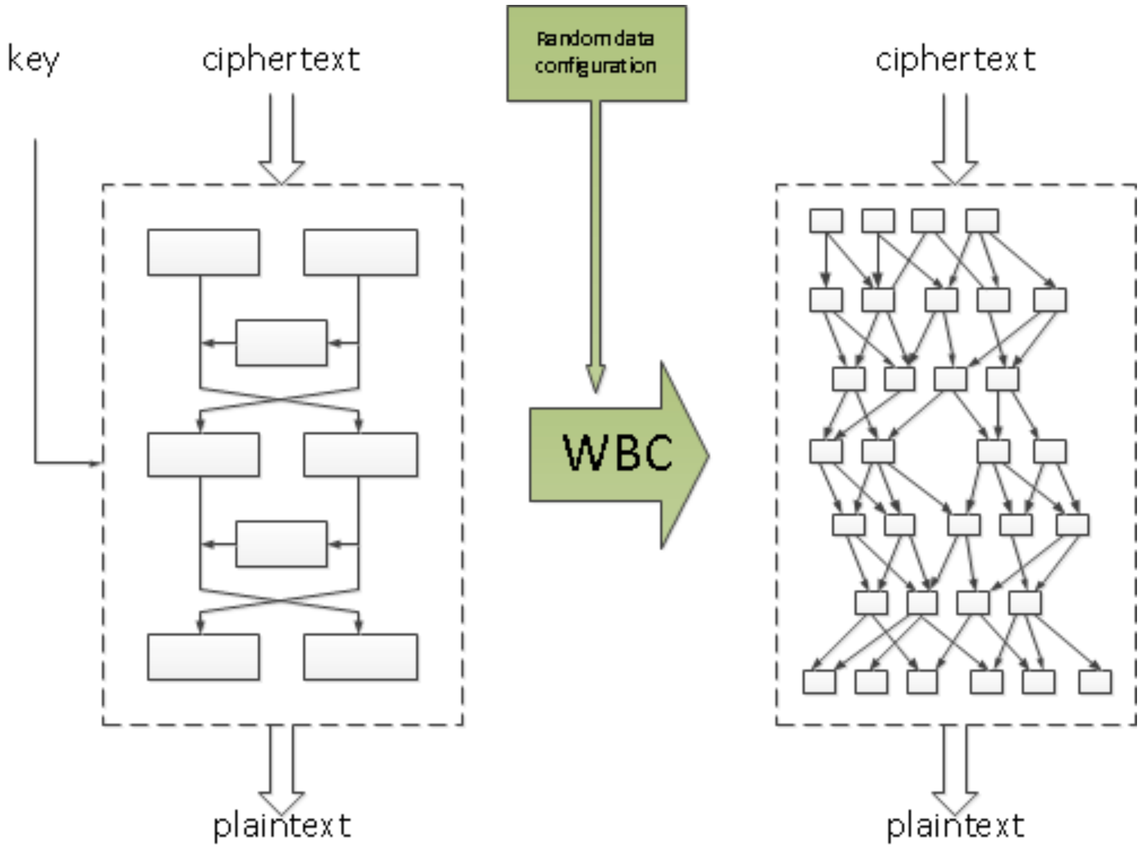
In this diagram, code which modifies itself based on the result of a branch, rather than executing different (static) code on each side of the branch itself

Media DRM

White box cryptography

The goal of white box cryptography is to provide a way to decipher data without revealing the key.

This is accomplished by embedding the key itself in code that gets executed, rather than using it as an input to a function



White Box Cryptography

Why ship a *hidden* key? To avoid easy replication of content (e.g. piracy)

Can't an attacker easily retrieve the key? No

Is the key still in the attacker's control? Yes

How is code lifting avoided?

How is algorithm manipulation avoided?

Can this really work on its own?

White Box Cryptography

Code lifting

Code lifting consists in basically taking the code that represents the cryptographic function and running it outside the original application, thus decrypting arbitrary data.

White Box Cryptography

Code lifting

This can be avoided by implementing additional encoding techniques to the decrypted content, such that only the original application can use it/play it back.

This really isn't that secure, as the code is still running on a white box controlled by the attacker. Thus given enough time those operations can be reverse engineered and copied

White Box Cryptography

Algorithm manipulation

The white box cryptographic algorithm's code can be manipulated by the attacker to better understand it or to try and retrieve the key.

An example is zeroing out xor tables or other affine operations to get closer to the beginning of a decryption round (the key is thus easier to retrieve)

White Box Cryptography

Algorithm manipulation

To make the attacker's life more difficult, substitution tables that use machine code as entries can be made, so that if the code gets modified, the transformations do too, and the key gets lost.

This also is provably not secure against reverse engineering and memory cloning attacks.

White Box Cryptography

At last, why even bother?

At some point in time, the decrypted content is in attacker-controlled memory, and attacks on white box cryptographic algorithms can be used to retrieve the symmetric key. It doesn't seem like a good idea to use this to protect content.

The reality is that, alongside obfuscation techniques, WBC buys companies enough time before it's cracked, that it's more profitable using it as a DRM rather than nothing.

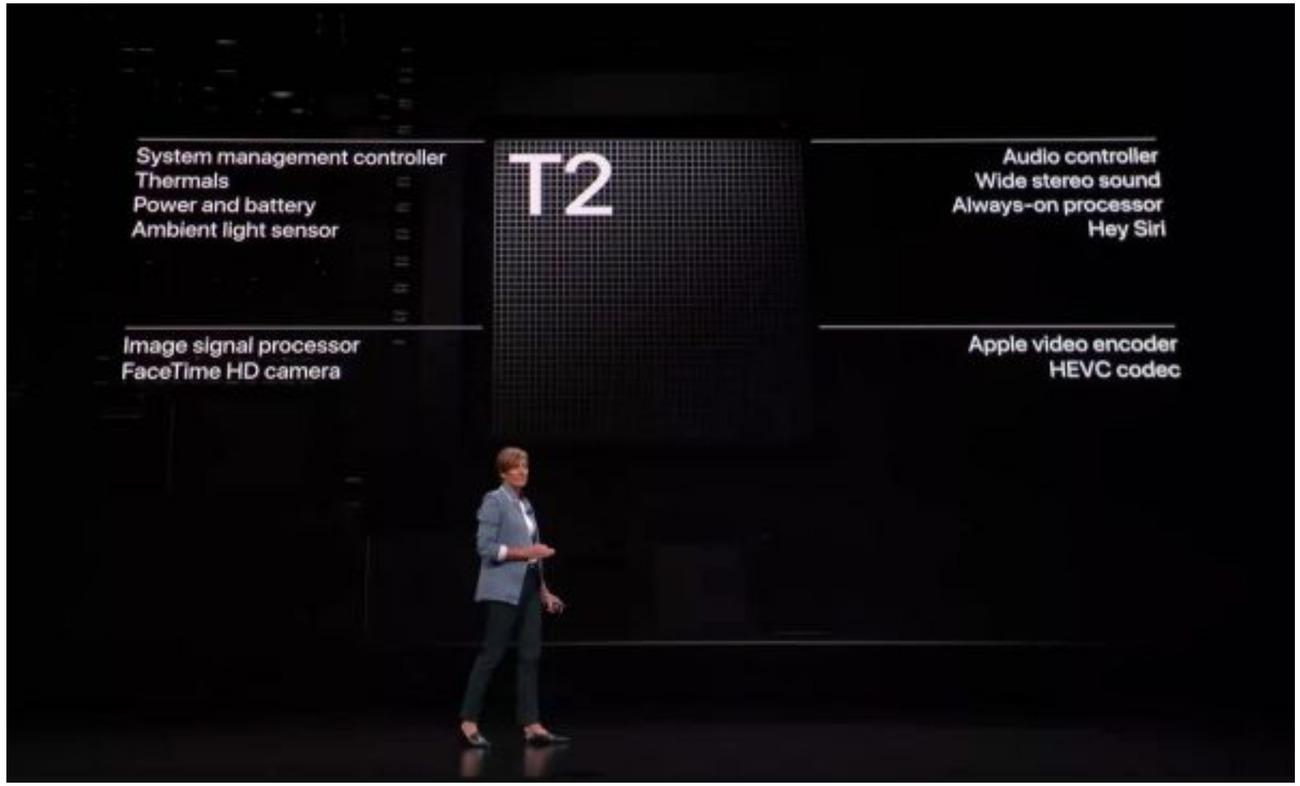
Black Box

Tech Corps' wet dreams

- Secure Enclave processors
- Intel Management Engine (etc.)
- Much more

Used to protect:

- Advanced Media DRM
- Cryptographic secrets
- Proprietary algorithms

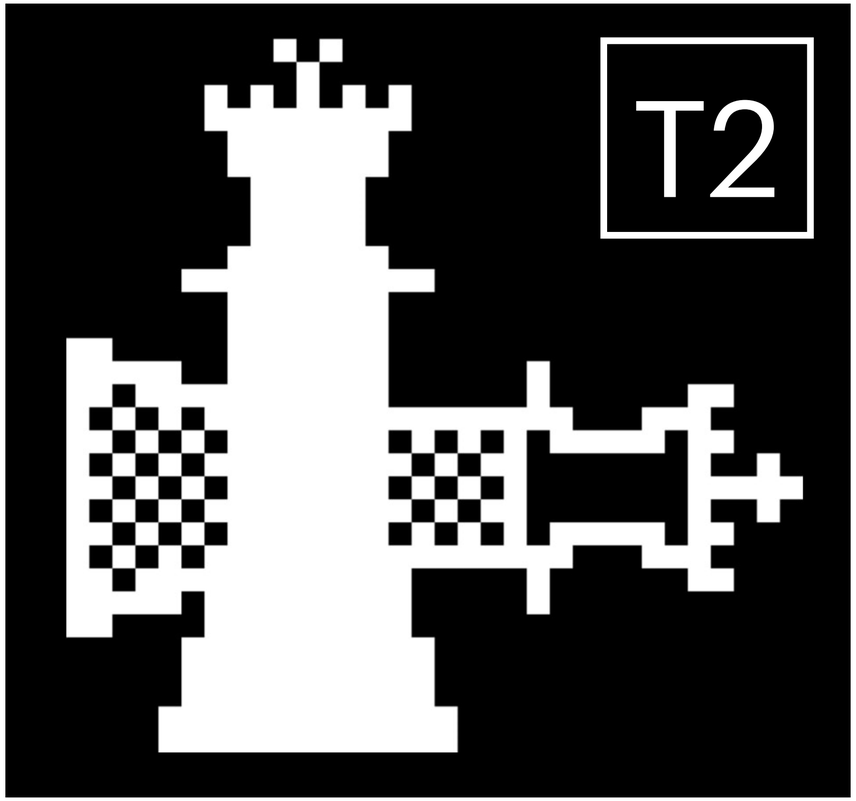


The Apple T2 security coprocessor handles everything ranging from basic hardware control to advanced audio/video processing, in order to protect the proprietary algorithms used.

The T2 chip is required to use Apple's FairPlay DRM, which protects video streaming services such as Netflix

Pwning Black Boxes?

Maybe for another talk



Ask me anything

Twitter: @NotHdesk