Watermarking

Paolo Falcarin

Ca' Foscari University of Venice

Department of Enviromental Sciences, Informatics and Statistics

paolo.falcarin@unive.it

Credits: Christian Collberg

CM0626 – Software Security

1 April 2025

What is Watermarking?

Ca' Foscari University of Venice

- Embed a unique identifier into the executable of a program.
- A watermark is much like a copyright notice.
- It will not prevent an attacker from reverse engineering the program.

What is Watermarking?

Ca' Foscari University of Venice

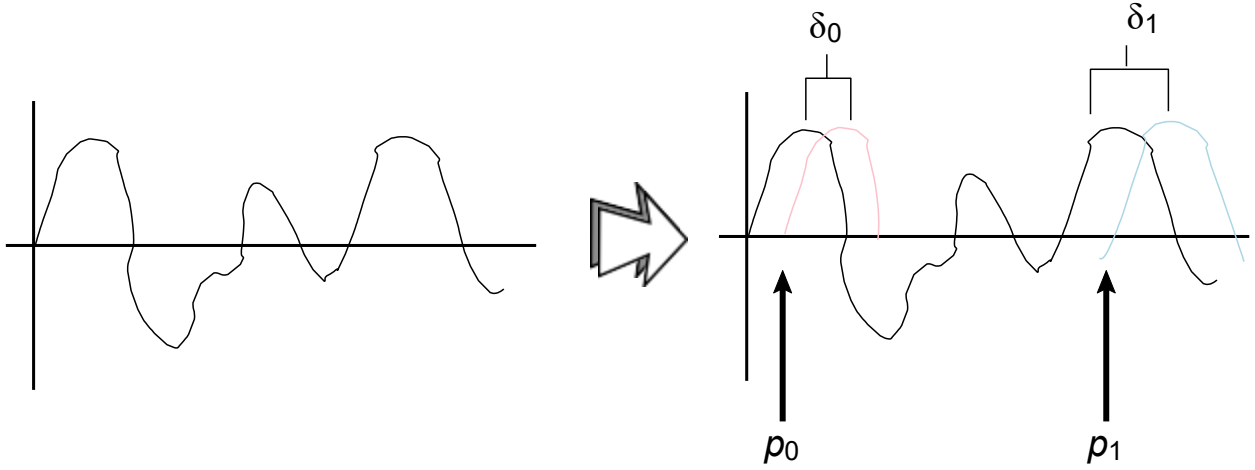
- Allows us to show that the program the attacker claims to be theirs, is actually ours.
- Software fingerprinting: every copy you sell will have a different unique mark in it
- Traitor Tracing: trace the copy back to the original owner, and take legal action.

Watermarking
History and Applications

Audio marking: Echo hiding

Ca' Foscari University of Venice

- Embed echoes that are short enough to be imperceptible to the human ear:



Audio: Least Significant Bit

Ca' Foscari University of Venice

- LSB of an audio sample is the one that contributes least to your perception
- Alter without adversely affecting quality!
- *Attack*: randomly replace the least significant bit of every sample!

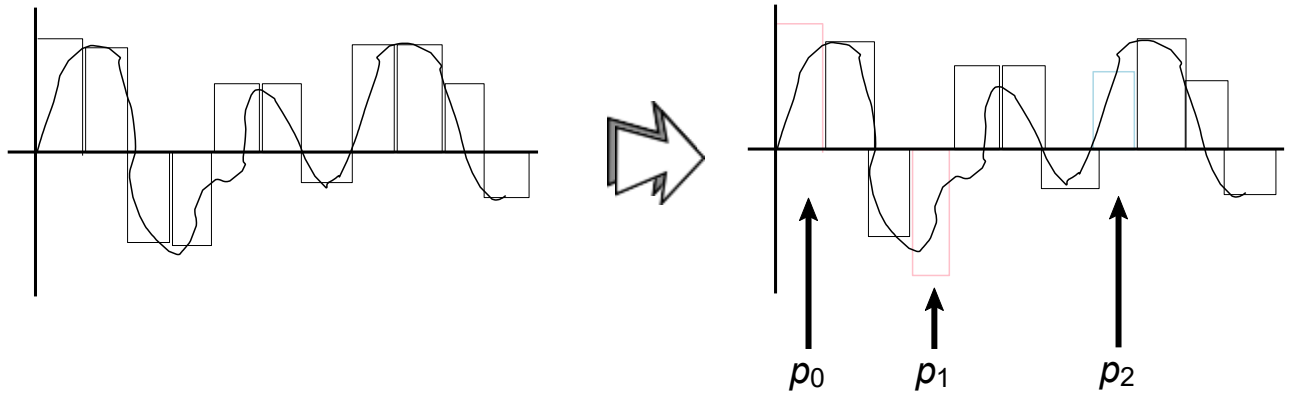
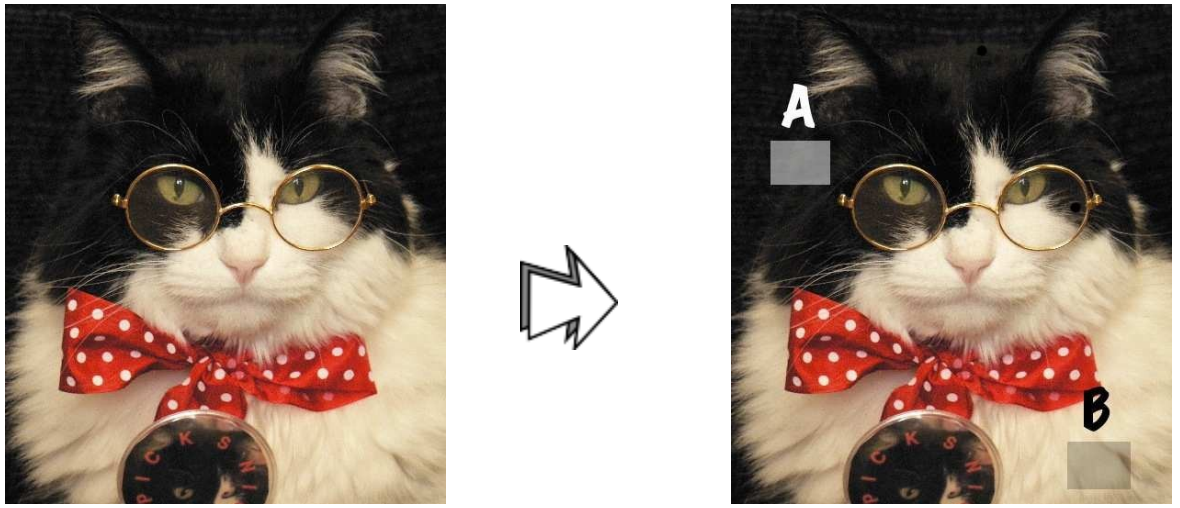


Image: Patchwork

Ca' Foscari University of Venice

- Embed a single bit by manipulating the brightness of pixels.
- Use a pseudo-random number sequence to trace out pairs (A, B) of pixels
- During embedding adjust the brightness of A up by a small amount, and B down by the same small amount:



Patchwork: Embedding And Recognition

Ca' Foscari University of Venice

- **EMBED**(P, key):

1. Init RND(key); $\delta \leftarrow 5$
2. $i \leftarrow$ RND(); $j \leftarrow$ RND()
3. Adjust the brightness of pixels a_i and b_i :

   $a_i \leftarrow a_i + \delta$; $b_j \leftarrow b_j - \delta$
4. repeat from 2 ≈ 10000 times

- **RECOGNIZE**(P, key):

1. Init RND(key); $S \leftarrow 0$
2. $i \leftarrow$ RND(); $j \leftarrow$ RND()
3. $S \leftarrow S + (a_i - b_j)$
4. repeat from 2 ≈ 10000 times
5. if $S \gg 0 \Rightarrow$ output "marked!"

Blind vs Informed

Ca' Foscari University of Venice

- Watermarking recognizers are either blind or informed.
- To extract a blind mark you need the marked object and the secret key.
- To extract an informed mark you need extra information, such as original, unwatermarked, object.

StegHide on Kali

Ca' Foscari University of Venice

```
sudo apt-get install steghide

steghide embed -ef secret.txt -ef cat.jpg

steghide extract -sf cat.jpg
cat secret.txt
```

1. Install steghide
2. Embed the text in file secret.txt into the image cat.jpg (here below)
3. You will be prompted to create a passphrase
4. Share the image as you like
5. Extract the secret and insert the passphrase

(Extract it in a different folder to avoid overwriting the original secret.txt file)

StirMark

Ca' Foscari University of Venice

- StirMark is a generic free tool for simple robustness testing of image watermarking algorithms and other steganographic techniques.
- It can be applied to photographic digital images and it will distort the watermark of too simplistic marking techniques
  - => the embedded watermark or steganographic message cannot any more be detected and decoded from the result image.
- Stirmark can be used to destroy marks in embedded images!
  - https://www.cl.cam.ac.uk/~mgk25/stirmark.html

Watermarking Text

Ca' Foscari University of Venice

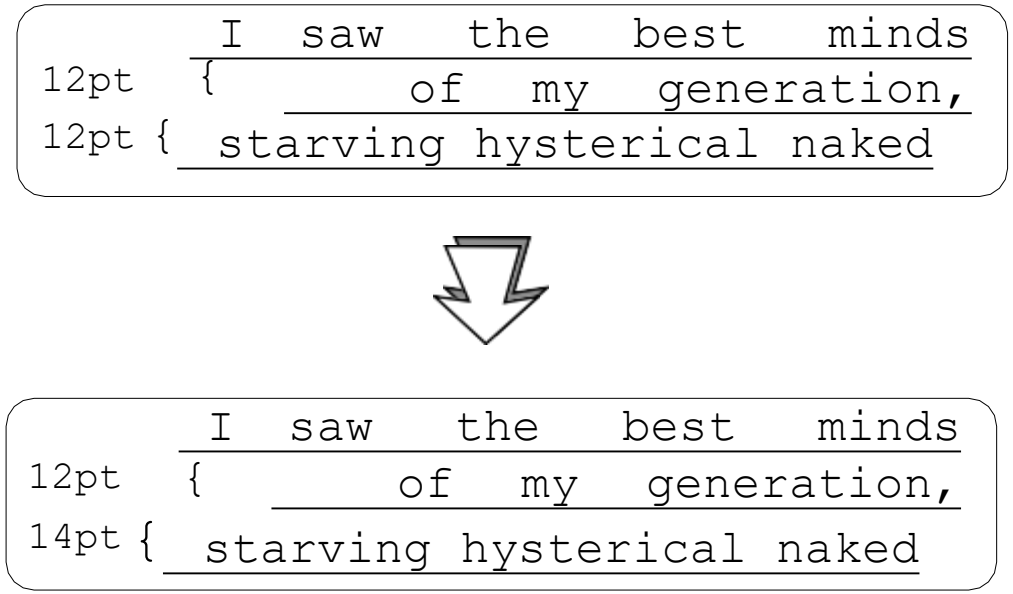
Cover object types:

- the text itself with formatting (ASCII text);
- or free-flowing text;
- an image of the text (PostScript or PDF).

Watermarking Text: PDF

Ca' Foscari University of Venice

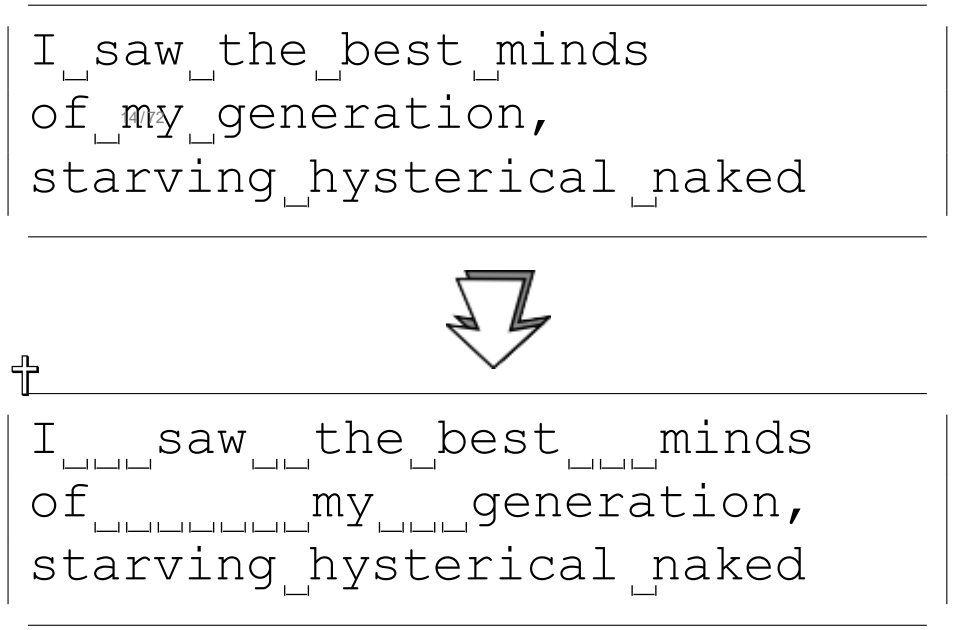
- Similar to marking images.
- Example: encode 0-bit or a 1-bit by hanging word/line spacing.



Watermarking Text: formatted ASCII

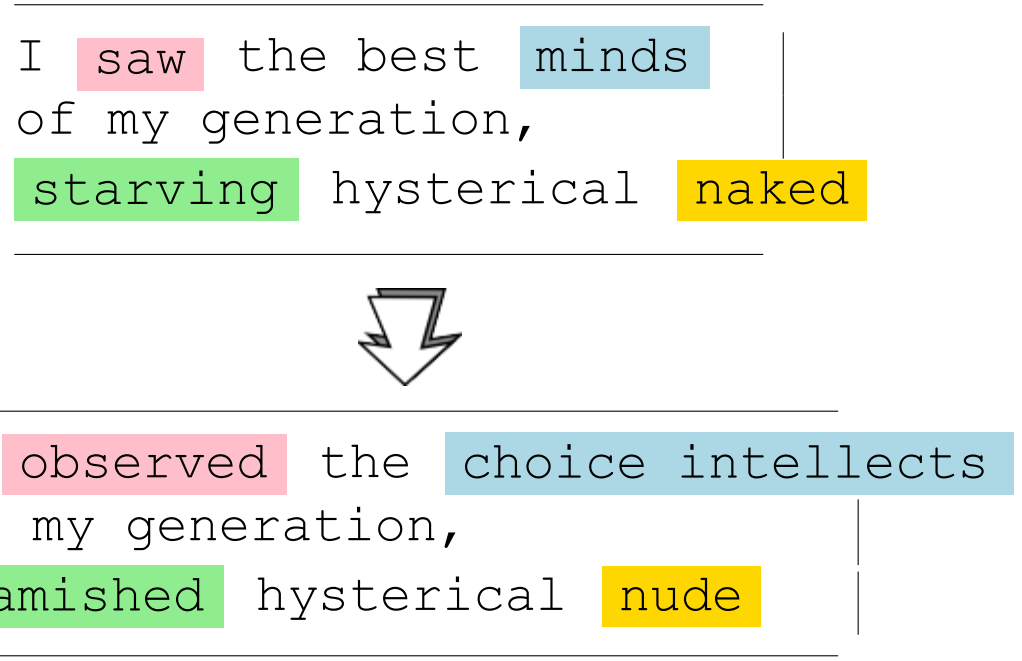
Ca' Foscari University of Venice

- Encode the mark in white-space: 1 space = 0-bit, 2 spaces = 1-bit:



Watermarking Text: Synonym replacement

- Replace words with synonyms.
- Insert spelling or punctuation errors.



Watermarking Text: Syntax

Ca' Foscari University of Venice

- Encode a mark in the syntactic structure of an English text:

1. Devise an extract function which computes a bit from a sentence
2. Modify the sentence until it embeds the right bit.

```
I saw the best minds
of my generation,
starving hysterical naked
```

```
It was the best minds
of my generation that I saw,
starving hysterical naked
```

Watermarking Text: Atallah et al.

Ca' Foscari University of Venice

- Chunk up the watermark, embed one piece per sentence.
- A function computes one bit per syntax tree node.
- Modify sentence until these bits embed a watermark chunk.
- A marker sentence precedes every watermark-bearing sentence.

```
I saw the best minds
of my generation,
starving hysterical naked
```

```
I saw the best minds of my
generation. They were starving
hysterical naked. None, baby,
none were smarter than them. Nor
more lacking in supply of essential
nutrients or in more need of
adequate clothing. Baby.
```

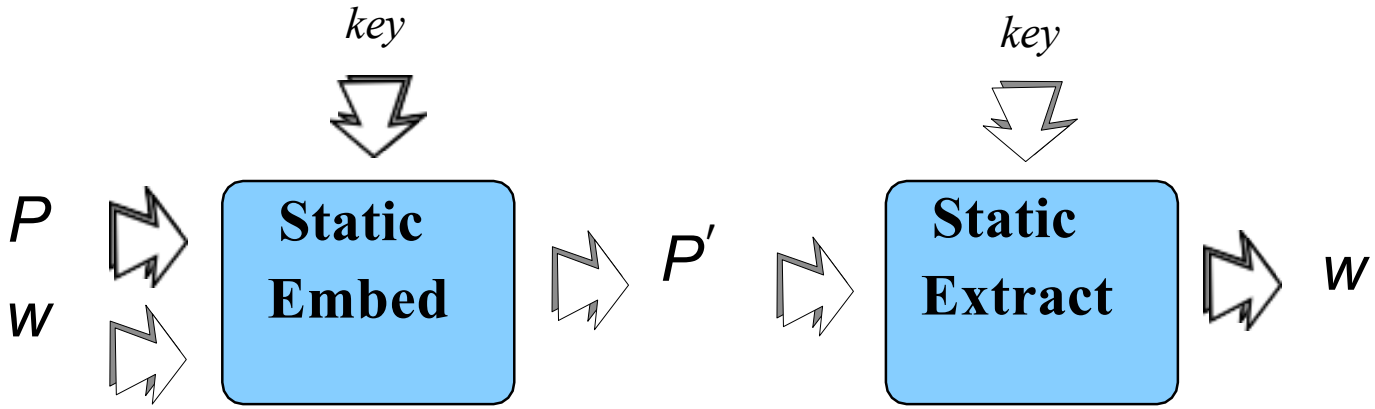
Watermarking Software

Static watermarks

Ca' Foscari University of Venice

You care about

- Encoding bitrate: percentage of bits used by the watermark
- Stealth: strength of the cover of the watermark
- Resilience to attack



Ideas for Software Watermark Algorithms

Ca' Foscari University of Venice

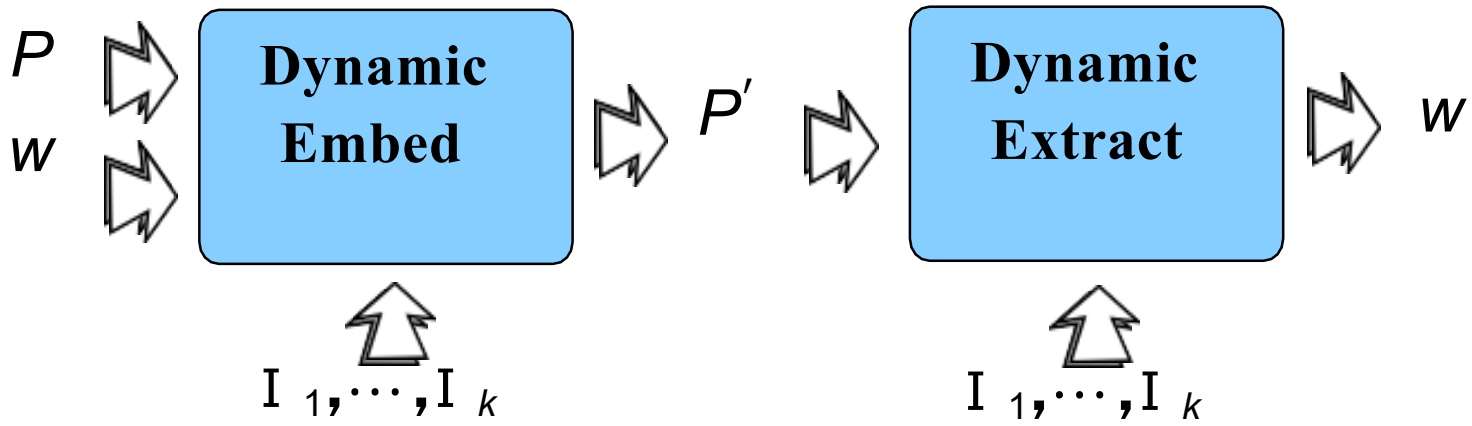
Encode the watermark

- in a permutation of a language structure
- in an embedded media object
- in a statistical property of the program
- as a solution to a static analysis problem
- in the topology of a CFG

Dynamic watermarks

Ca' Foscari University of Venice

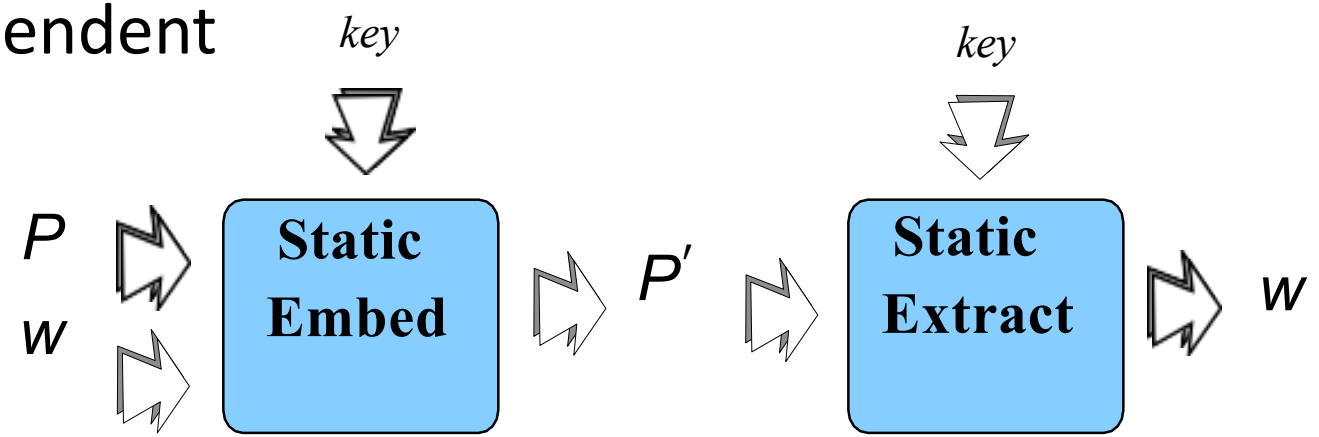
- Encode the watermark in the runtime state of the program
- Dynamic marks appear more robust, but are more complex to use



Attacks against software watermarks

Ca' Foscari University of Venice

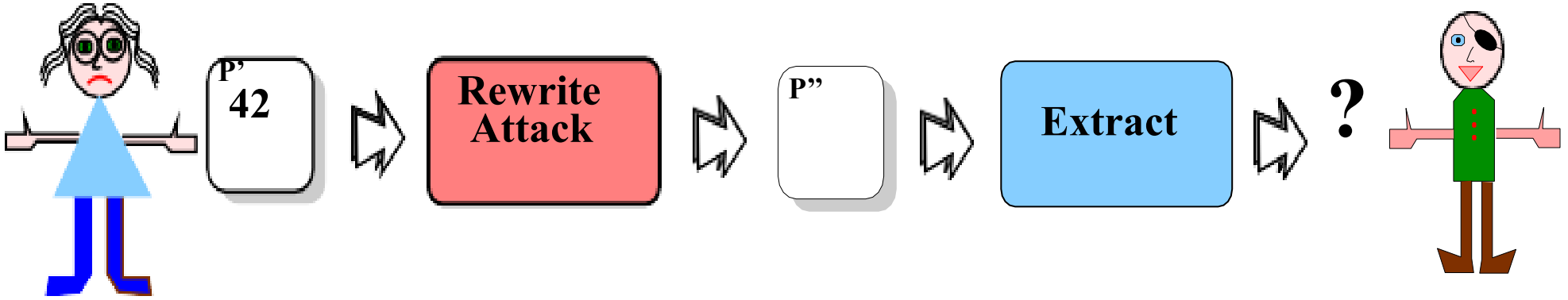
- The adversary knows the algorithm
- The adversary has complete access to the program
- The adversary doesn't know the key
- The adversary doesn't know the embedding location
  - it's key dependent



Attacks — Rewrite attack

Ca' Foscari University of Venice

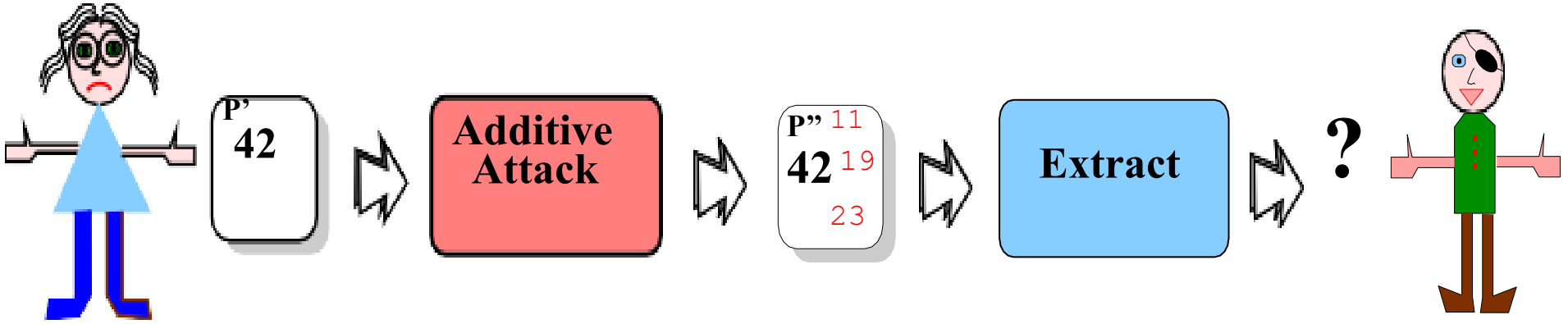
- Alice has to assume that Bob will try to destroy her marks before trying to resell the program!
- One attack will always succeed. . .
- Ideally, this is the only effective attack.



Attacks — Additive attack

Ca' Foscari University of Venice

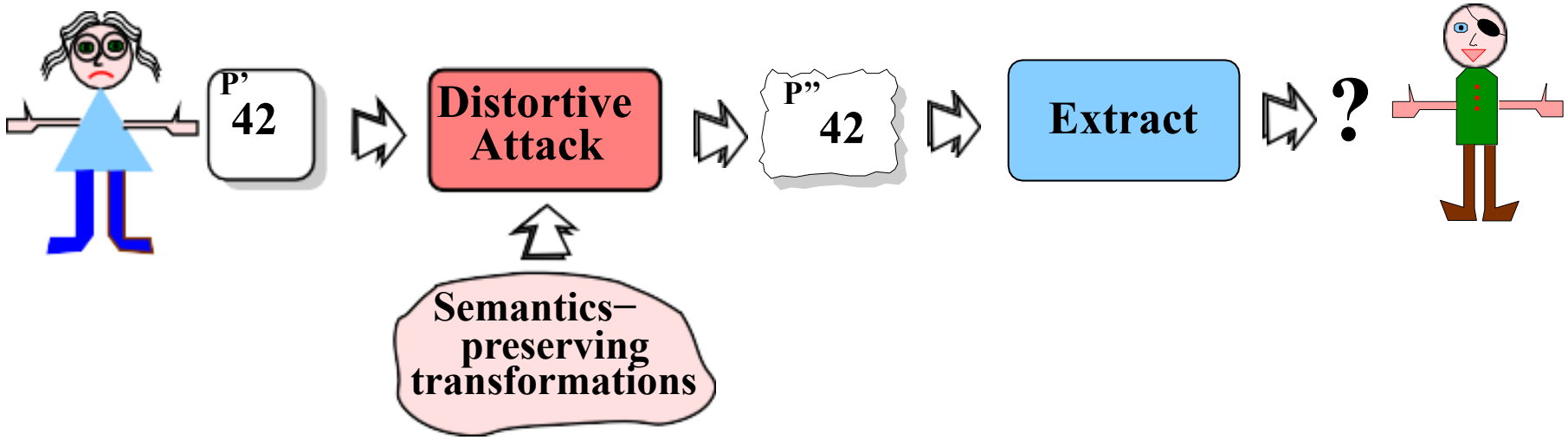
- Bob can also add his own watermarks to the program
- An additive attack can help Bob to cast doubt in court as to whose watermark is the original one.



Attacks — Distortive attack

Ca' Foscari University of Venice

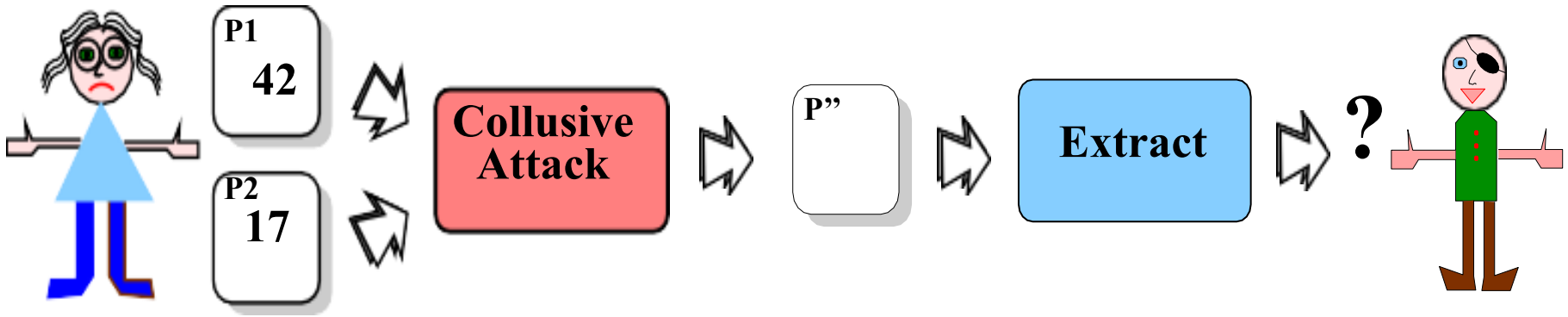
- A distortive attack applies semantics-preserving transformations to try to disturb Alice's recognizer
- Transformations: code optimizations, obfuscations, …



Attacks — Collusive attack

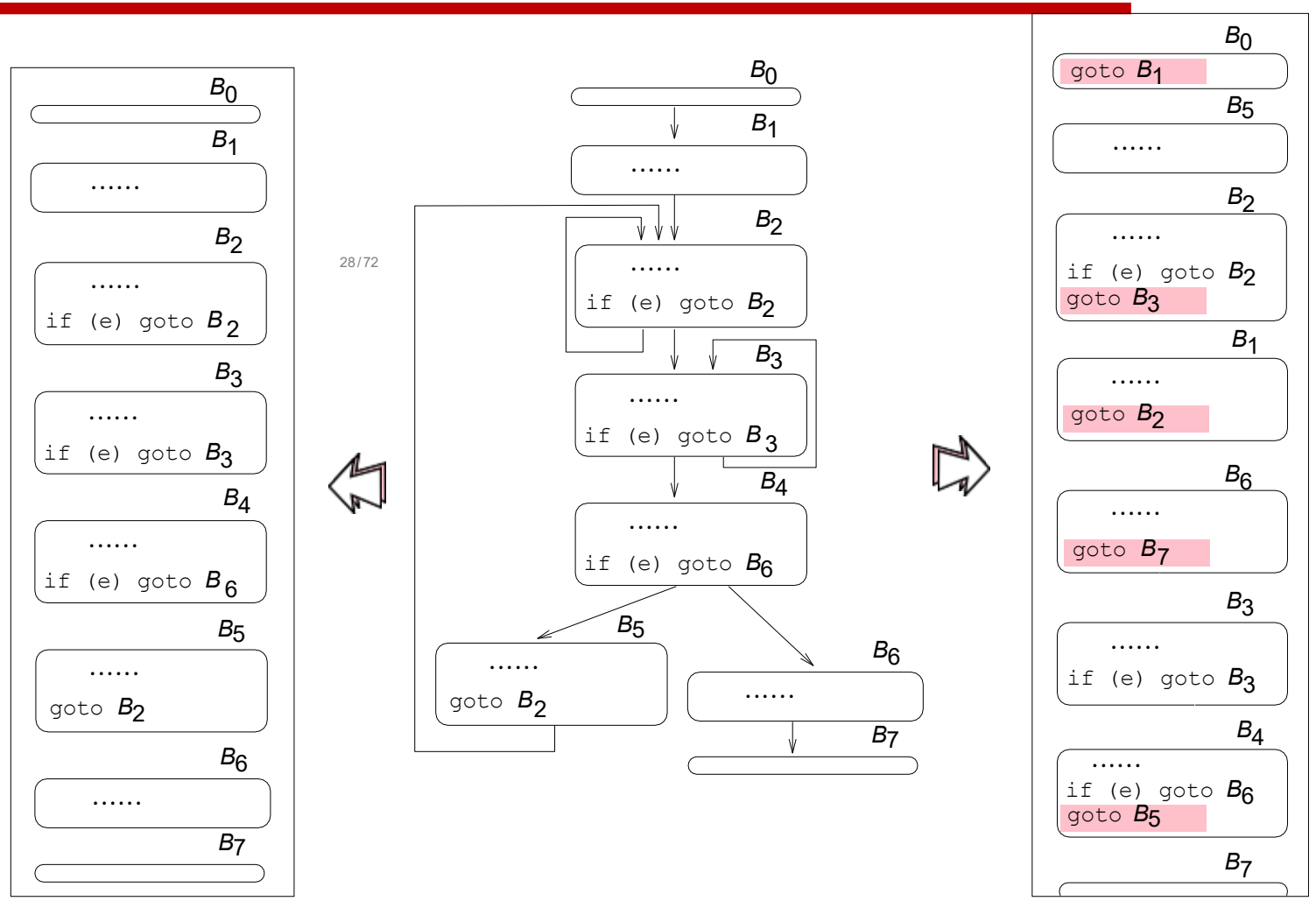
Ca' Foscari University of Venice

- Bob buys two differently marked copies and comparing them to discover the location of the fingerprint
- Alice should apply a different set of obfuscations to each distributed copy, so that comparing two copies of the same program will yield little information.



Watermarking by Permutation

Algorithm wmDM: Reordering Basic Blocks



Algorithm wmDM: Reordering Basic Blocks

Ca' Foscari University of Venice

- Performance overhead of 0-11% for three standard high-performance computing benchmarks.
- Negligible slowdown for a set of Java benchmarks.
- If you have m items to reorder you can encode $\log_2(m!) \approx \log_2(\sqrt{2\pi m}(m/e)^m) = O(m \log m)$ watermarking bits.
- What about stealth?

Algorithm wmVVS: Watermarks in CFGs

Ca' Foscari University of Venice

- Basic idea:

1. Embed the watermark in the CFG of a function.
2. Tie the CFG tightly to the rest of the program.

- Issues

1. How do you encode a number in a CFG?
2. How do you find the watermark CFG?
3. How do you attach the watermark CFG to the rest of the program?

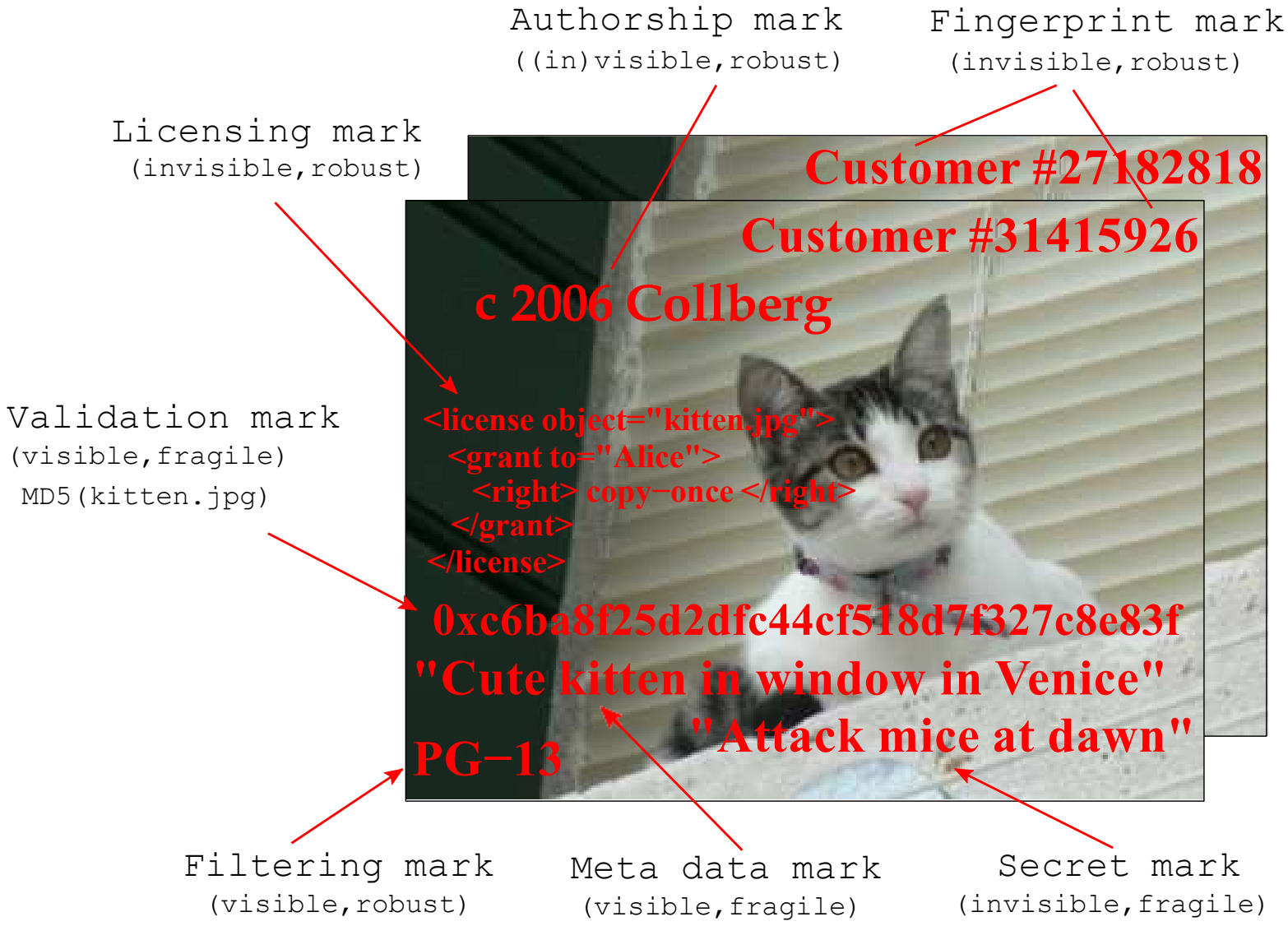
Watermark Embeddings

Ca' Foscari University of Venice

- Watermarks are
  - short identifiers
  - difficult to locate
  - hard to destroy
- The adversary
  - knows that the object is marked
  - knows the algorithm used
  - doesn't know the key
  - is active
- You care about
  - data-rate
  - stealth
  - resilience

Steganographic Embeddings

Ca' Foscari University of Venice

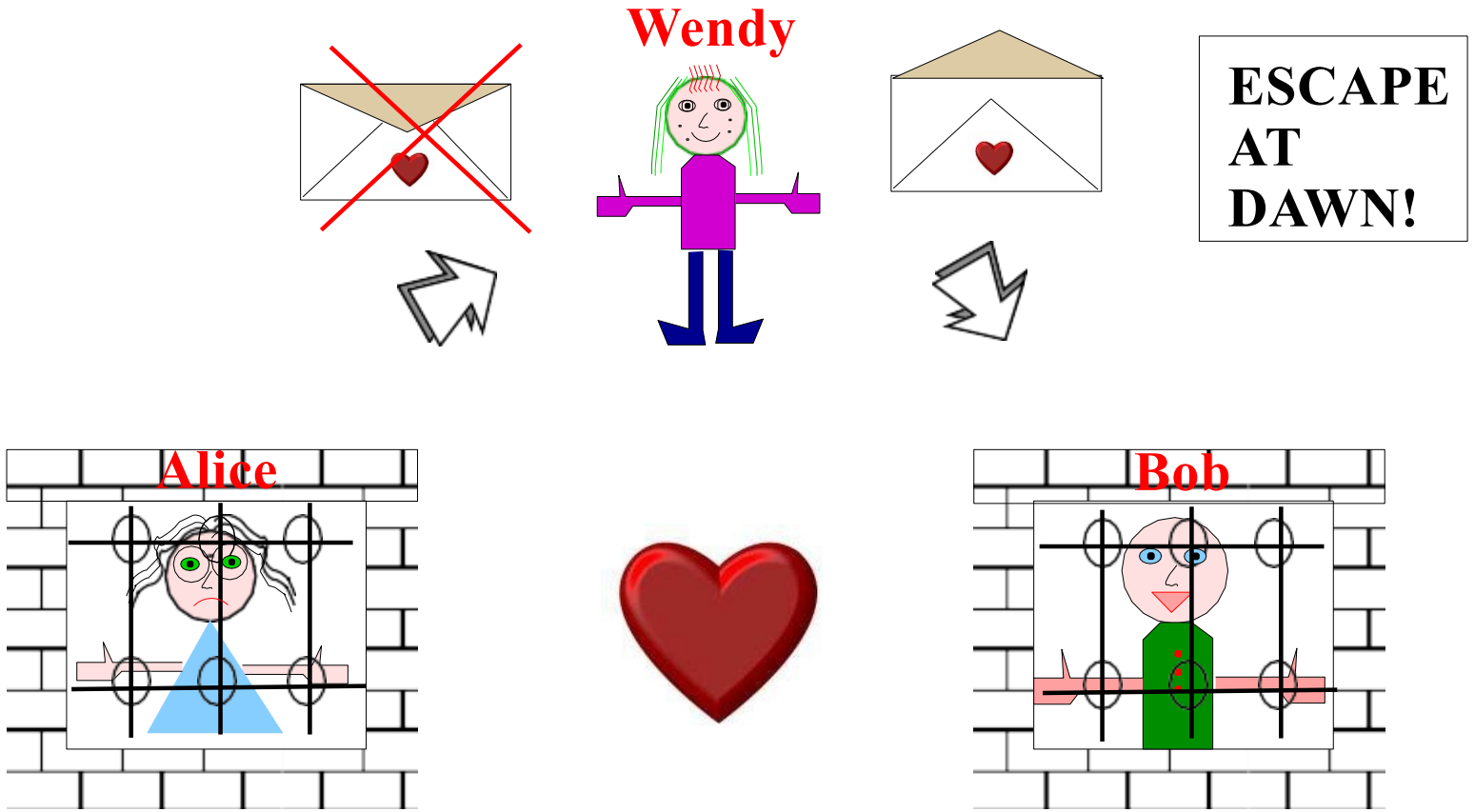


Steganographic Embeddings

Ca' Foscari University of Venice

- Stegomarks are
  - long identifiers
  - difficult to locate
- The adversary
  - wants to know if the object is marked
  - knows the algorithm used
  - doesn't know the key
  - is passive
- You care about
  - data-rate
  - stealth

Steganography — Prisoners' Problem



Steganography — Null cipher

Ca' Foscari University of Venice

Easter is soon, dear! So many flowers! Can you smell them? Are you cold at night? Prison food stinks! Eat well, still! Are you lonely? The prison cat is cute! Don't worry! All is well! Wendy is nice! Need you!):

wmASB: Hidden Messages in x86 Binaries

Ca' Foscari University of Venice

- Basic idea: Play Compiler
  - whenever the compiler has a choice in which code to generate, or the order in which to generate it, pick the choice that embeds the next bits from the message W.
- Four sources of ambiguity:
  - code layout (ordering of chains of basic blocks)
  - instruction scheduling (instruction order within basic blocks)
  - register allocation
  - instruction selection

wmASB: Embedding

Ca' Foscari University of Venice

1. Construct:
   1. codebook B of equivalent instruction sequences
   2. statistical model M of real code
2. Encrypt W with key
3. Canonicalize P:
   1. Sort block chains, procedures, modules
   2. Order instructions in each block in standard order

```
mul ri,x,5
shl ri,x,2
add ri,ri,x
add ri,x,x
add ri,ri,ri
add ri,ri,x
```

WMASB: Embedding

Ca' Foscari University of Venice

4. Code layout: Embed bits from *W* by reordering code segments within the executable.
5. Instruction scheduling:
   1. Build dependency graph
   2. Generate all valid instruction schedules
   3. Embed bits from W by picking a schedule

   Use M to avoid picking unusual schedules.

6. Instruction selection:
   - Use B to embed bits from *W* by replacing instructions.
   - Use M to avoid unusual instruction sequences.

WMASB: Stealth

Ca' Foscari University of Venice

- Instruction selection:
  - There are 3078 different encodings of three instructions for EAX=(EAX/2)!
  - Most don't occur in real code. . .
- Instruction scheduling:
  - Avoid bad schedules: no compiler would generate it!
  - Avoid generating different schedules for two blocks with the same dependency graph!
- Code layout:
  - Compilers lay out code for locality: don't deviate too much from that!

WMASB: Stealth

Ca' Foscari University of Venice

- Encoding rate
  - Unstealthy code 1/27
  - Stealthy code 1/89
- Encoding space:
  - 58% from code layout
  - 25% from instruction scheduling
  - 17% from instruction selection
- Real code doesn't use unusual instruction sequences.
- Real code contains many schedules for the same dependency graph

How to design a watermarking algorithm

Ca' Foscari University of Venice

- Find a language into which to encode the mark (CFGs, threads, dynamic control flow. . .)
- Construct an encoder/decoder (number $\leftrightarrow$ CFG,. . .)
- Construct a tracer/locator to find locations for the mark (using key, every function, . . .)
- Construct an embedder/extractor to tie the mark to surrounding code
- Decide on an attack model.

Dynamic Watermarking

Static watermarking?

Ca' Foscari University of Venice

- Embedding ideas: Encode the watermark
  1. as a permutation of the original code, or
  2. in new but non-functional code
- Recognition: Extract the mark by analysing the code itself.
- Attack ideas: Disrupt the recognizer by permuting the original code, or embedding your own watermark, . . .

Dynamic watermarking!

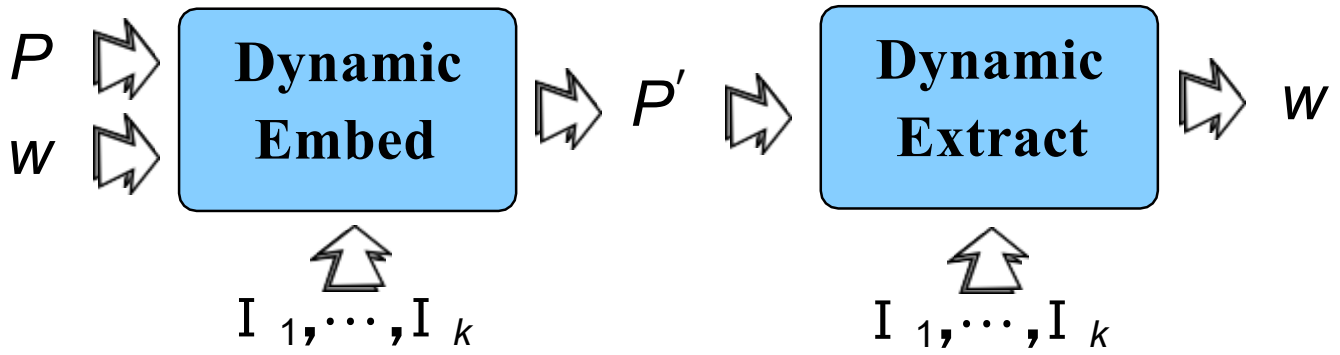
Ca' Foscari University of Venice

- Embedding idea: Embed the mark by adding code that changes the program's behaviour.
- Recognition idea: Extract the mark by *running* the program and analysing its behaviour.
- The program produces the watermark into its state.

Dynamic Watermarking API

Ca' Foscari University of Venice

- Secret key: a special input sequence $I_1, \cdots, I_k$



Example

- Simple semantics-preserving transformations are less likely to affect a dynamic watermark.
- Secret watermarking input sequence: "hello","world".
- The mark is stored in a global variable watermark
- Recognition Procedure:
  - Run the watermark program with the secret input,
  - Examine the state for the watermark.

```
int watermark=0;
...
if (read() == "hello")
if (read() == "world")  watermark=42;
```

Example

- An attacker might apply various static code transformations to destroy the mark
- Static recognizers — get confused!
- Dynamic recognizers —run the code, look at the value of *watermark*, read out the mark.

```
int watermark=0; int x = 3;
x = x*2;
...
if (read() == "hello")
if (read() == "world")  watermark=x*6+6;
```

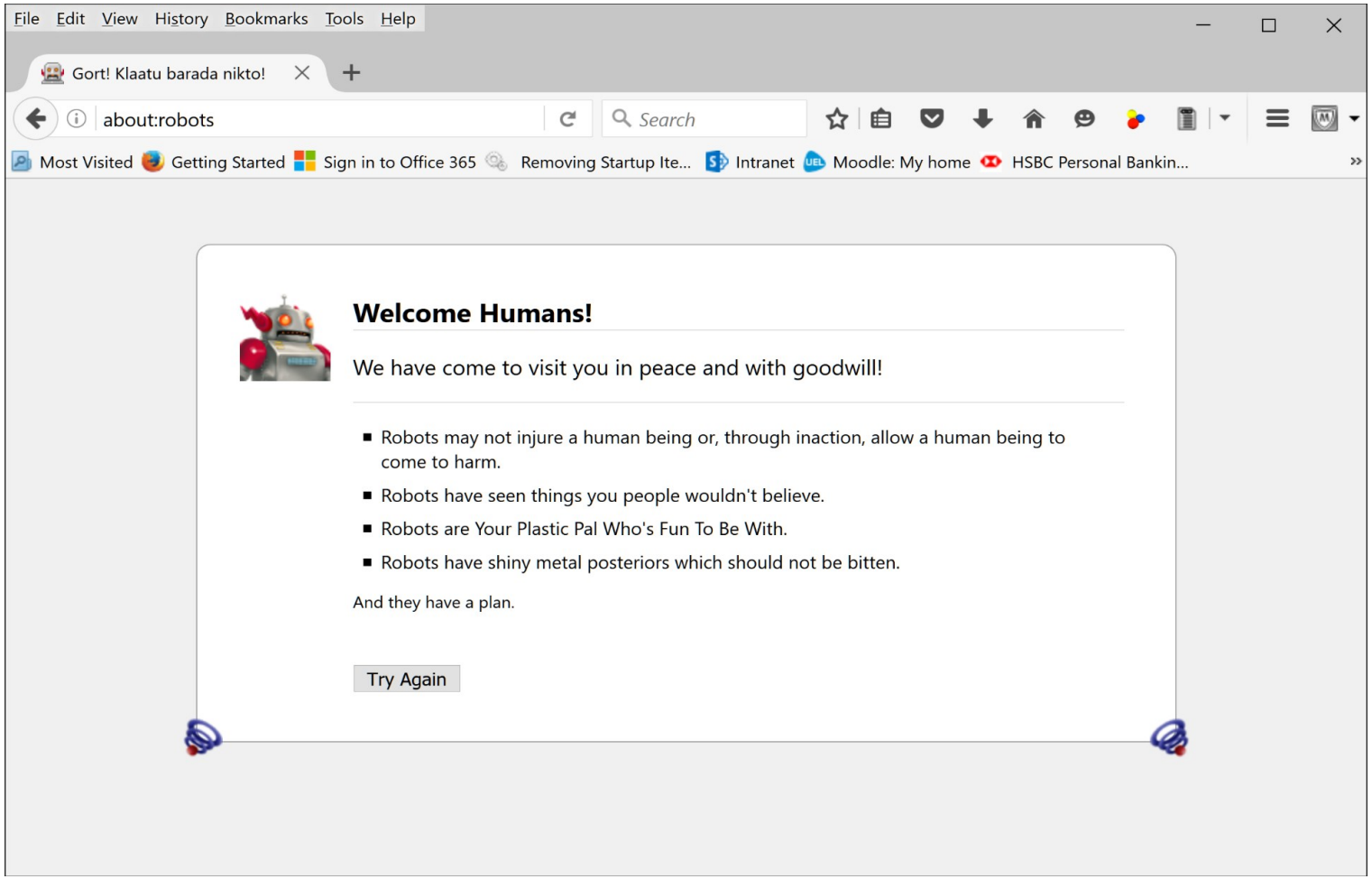
Dynamic Watermarks

Ca' Foscari University of Venice

- A virtual Easter egg is an example of dynamic watermark, an intentional hidden message or feature in an object such as a movie, book, DVD,
- Also in computer programs.
  - Try "about:mozilla" in Firefox
  - Try "about:robots" in Firefox

Mozilla Firefox example

Ca' Foscari University of Venice



Attack against dynamic state

Ca' Foscari University of Venice

- The attacker could split the watermark variable!
- Careful in choosing what state in which you store the mark!
  1. It should be difficult for the adversary to modify the state.
  2. Modifying the state would make the program too slow or too large to be useful.
  3. It should be easy to tamperproof the state.

Disadvantages of dynamic watermarks

Ca' Foscari University of Venice

- Dynamic algorithms can't protect parts of programs:
  - to recognize the mark the program needs to be executed!
- Dynamic watermarks have to be executed to be recognized.
- Non-determinism in the program can affect the recognizer.

Algorithm wmCT: Exploiting aliasing

Ca' Foscari University of Venice

- Basic insight: Pointer analysis is hard!
- Basic embedding idea:
  1. Let n be the watermark number
  2. Convert n to a graph G that encodes the number
  3. Convert G to code C that builds the graph
  4. Add C to your program so that it's executed for the special input
- Basic recognition idea:
  1. Run the program for the special input
  2. Dump all objects on the heap
  3. Reconstruct the graph G
  4. Convert G to the watermark number n

Algorithm wmCT: Example

Ca' Foscari University of Venice

```
public class M {
public void main (String args[]) {
 for (int i=1; i<args.length; i++) {
     if (args[0].equals(args[i])) {
          System.out.println("YES");
          return;
     }
 }
 System.out.println("NO");
}
```

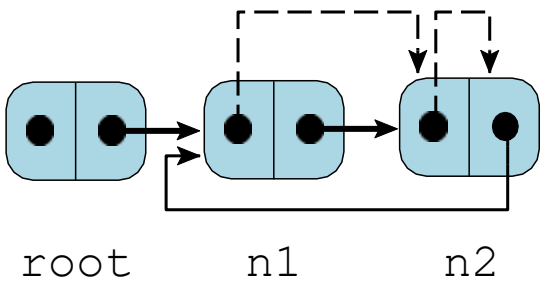
```
> java M 2 3 4 5 2
YES

> java M 2 3 4 5 3
NO
```

Algorithm wmCT: Example

Ca' Foscari University of Venice

- Here we're using a radix encoding.
- Let n = 2, the base-2 expansion is $1 \cdot 21 + 0 \cdot 20$:
- This graph is created by this code




```
WMNode n2 = new WMNode();
n2.digit = n2;
Node n1 = new WMNode();
n2.spine = n1;  n1.spine = n2;
n1.digit = n2;
```

Algorithm WMCT: Example

Ca' Foscari University of Venice

```
public class M {
   public static WMNode root;
        class WMNode {
                public Node spine, digit;
        }
  public void main (String args[]) {
        if (args[0].equals("2")) {
                Build the graph here!
        }
        for (int i=1; i<args.length; i++) {
                if (args[0].equals(args[i])) {
                        System.out.println("YES"); return;
                }       }
        System.out.println("NO");
}
```

Increasing bitrate

Ca' Foscari University of Venice

- How do we increase the bitrate?
- Easy — just build a bigger graph!
- But — bigger graph, less stealth!
- Ideas:
  1. split the graph in smaller pieces!
  2. choose an efficient graph encoding!

Algorithm WMCT: Summary

Ca' Foscari University of Venice

- Size overhead: embed a 20-bit watermark only requires 141 bytes of Java bytecode. (Tamperproofing code adds more).
- Performance overhead: the graph-building code is only executed for the special input.
- Chief advantage: tamperproofing is easy.
- Attacks: Scan the code for allocations and pointer manipulations.
- Stealth: good in OO programs.

wmNT — Thread-Based Watermark

Ca' Foscari University of Venice

- Embed mark in which threads execute which basic blocks.
- Can have huge performance degradation.
- Why? Parallelism-analysis is hard.