Purdue University

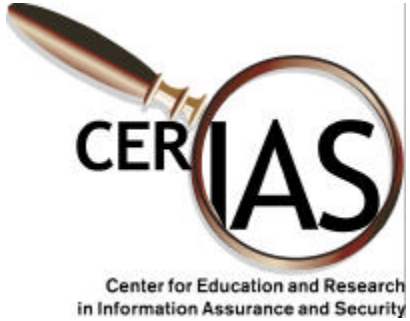
Center for Education and Research in Information Assurance and Security

Software Watermarking with Secret Keys

Jens Palsberg

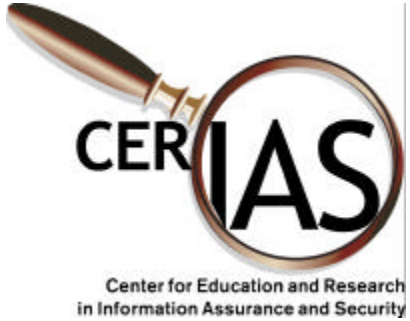
Purdue University

Secure Software Systems Group



Research Group

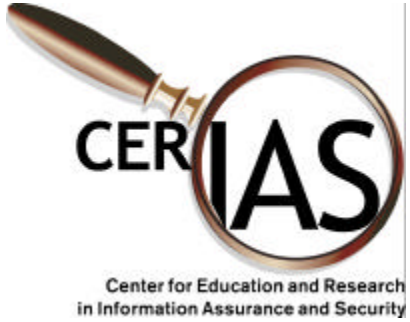
- Sowmya Krishnaswamy, Minseok Kwon, Di Ma, Quiyun Shao, Yi Zhang.
- Work supported by an NSF CAREER award and by Lilly Endowment Inc.



Secret Keys

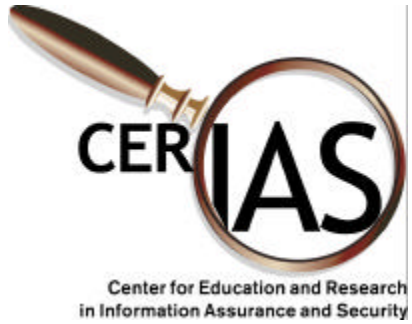
$encrypt_k(m)$

$watermark_k(P,W)$



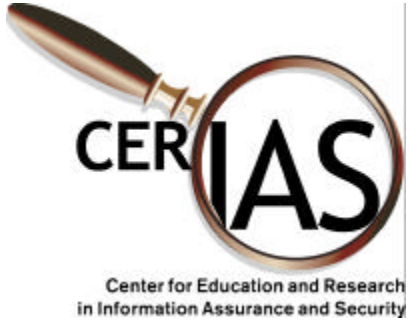
Overview

- ≥ 4 U.S. patents: software watermarking
- Collberg and Thomborsen (1999): embed a watermark in data structures.
- Our contribution: defense against ``open-source'' attacks.
- We have an implementation for Java: efficient, moderate overhead.



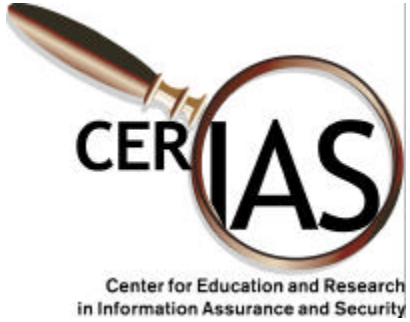
The Need to Prove Software Ownership

- Alice copyrights her software and sells it for profit.
- Bob manages to make a pirate copy.
- Bob wants the software because:
  - private use
  - industrial e-spionage
  - further selling for his own profit.



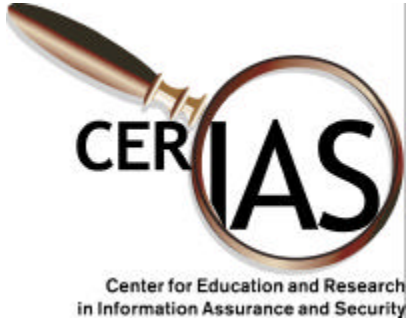
The Need to Prove Software Ownership

- Question: how does Alice protect her copyright?
- Answer: She must be able to prove ownership, possibly in a court of law, of a given copy of the software.



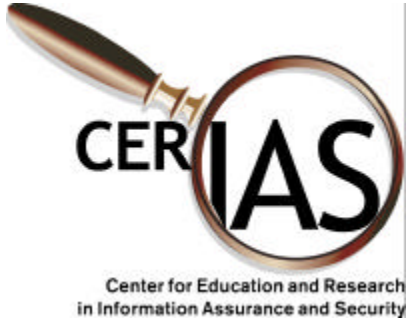
Approaches to Anti-piracy

- Keep a certified list of customers
- Link the software to the hardware.
- Link the software to a movable piece of hardware that cannot easily be copied
- Software watermarking: embed a secret into the software which can be retrieved on demand.



Attacks on Software Watermarks

- Locate, distort, remove the watermark.
- Transformation attacks: compilation, optimization, obfuscation, decompilation, dead-code removal.
- Defense: embed the watermark in data structures, not the structure of the program.



No, No, No!

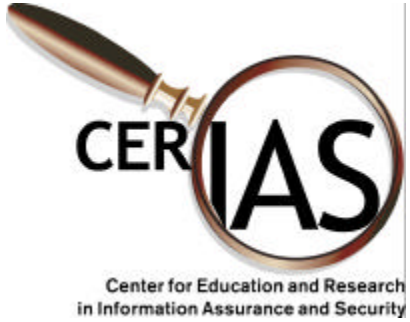
- A comment:

  /* My software, version 1.0 */

- A data string:

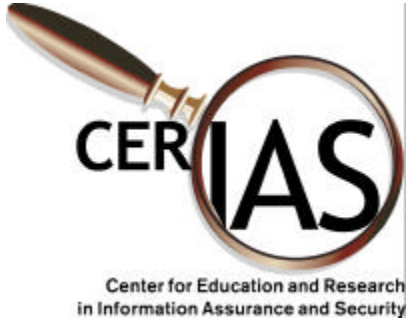
  printf("My software, version 1.0");

- A particular ordering of the instructions, e.g., the ordering of the branches of an n-branch switch-statement.



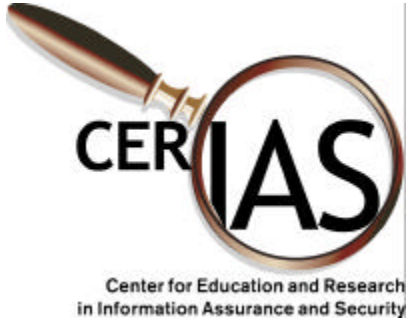
Attacks on Software Watermarks

- Subtractive attacks: Bob attempts to locate and remove the watermark.
- Additive attacks: Bob inserts his own watermark to make it plausible that his watermark came before Alice's.
- Collusion attacks: two attackers have two copies of the same watermarked program -- with different watermarks.



Open-Source Attack on Software Watermarks

- The attacker has access to the software for embedding and extracting watermarks.
- Brute-force: run embedding+extraction to learn how to locate a watermark.

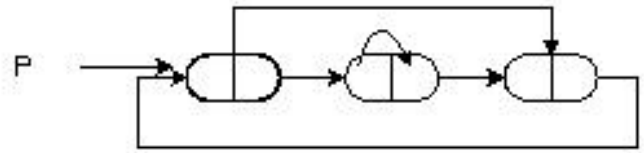


Secret Keys

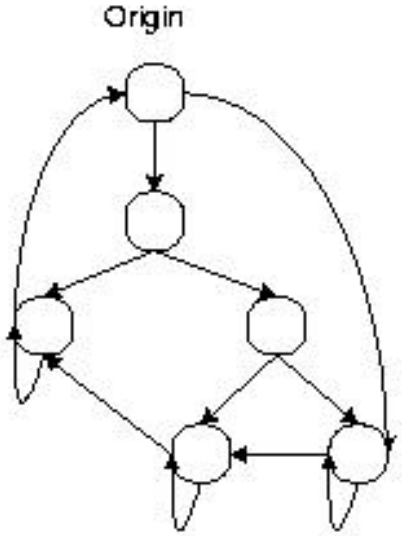
$encrypt_k(m)$

$watermark_k(P,W)$

Watermarks as Graphs

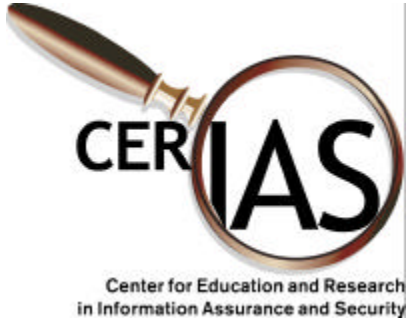


Radix-k



PPCT

Each graph encodes a number (the watermark,) which can contain the serial number, customer number, date, etc.



The Design of our Watermarking System

original data structures

built by

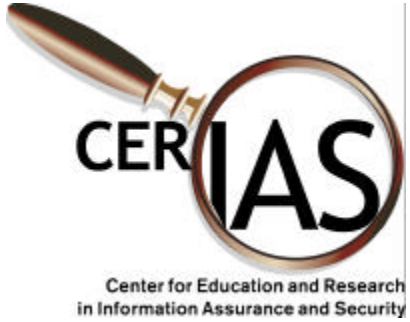
original code

obfuscated by

a watermark PPCT

built by

code that builds wm PPCT

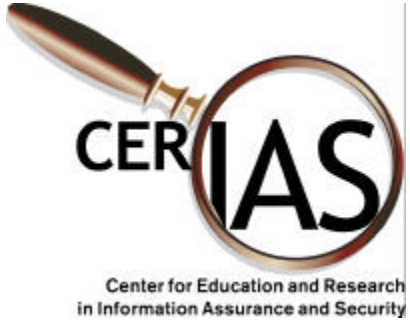


The Design of our Watermarking System

- W: the watermark
- P: the Java 1.2 program
- Three steps to watermark(P,W):
  1. generate(W)
  2. merge(P,generate(W))
  3. obfuscate the outcome of (2).

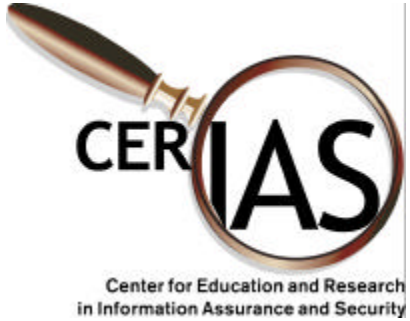
Benchmark Programs

Program	Description	Test input
javac	a compiler for Java	the JavaCup source code
javadoc	a Java API documentation generator	the JavaCup source code
JavaCup	an LALR parser-generator for Java	the CORBA grammar
JTB	a frontend for Sun's JavaCC	the Java 1.2 grammar
JavaWiz	our watermarking system	the JTB source code
JAX	a Java packaging tool from IBM	Hanoi demo shipped with JAX
BLOAT	a Java bytecode optimization tool	the JavaWiz source code



Experimental Results

Program	Lines of Code		Time to Watermark	Running Time		Needed Heap Space	
	before	after		before	after	before	after
javac	6,053	6,946	3.4s	2.8s	3.2s	1.6 MB	1.9 MB
javadoc	7,812	8,407	3.7s	11.8s	13.3s	4.6 MB	6.0 MB
JavaCup	11,029	11,799	2.8s	5.3s	7.4s	2.3 MB	3.5 MB
JTB	13,125	14,059	3.0s	5.6s	7.9s	1.6 MB	2.5 MB
JavaWiz	22,397	23,152	3.4s	3.0s	5.3s	2.0 MB	2.6 MB
JAX	22,705	23,537	2.7s	12.3s	13.6s	7.2 MB	7.8 MB
BLOAT	55,518	56,052	4.1s	3.6s	3.9s	0.5 MB	1.0 MB



Conclusion

- Our watermarking system is efficient
- The watermarked programs use only moderately more time and space
- The watermarked programs are resilient to attacks.
- Coming soon: measurements of the extraction process.