On the (Im)possibility of Obfuscating Programs

joint work with

Boaz Barak, Oded Goldreich
Russell Impagliazzo, Steven Rudich
Amit Sahai, Salil Vadhan

Ke Yang

February 21, 2002

Carnegie Mellon

Obfuscate

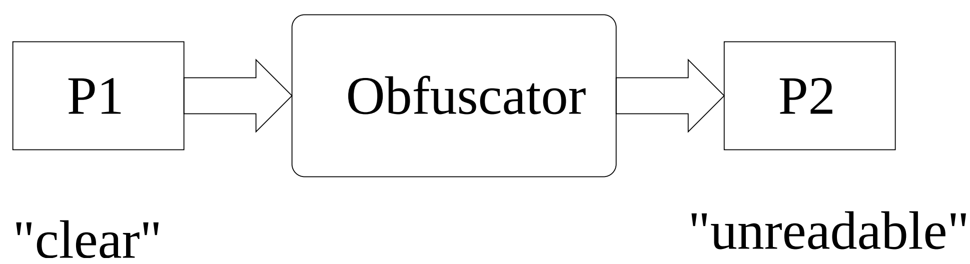
Obfuscate \ Ob*fus"cate \:

To make so confused or opaque as to be difficult to perceive or understand

(American Heritage Dictionary)

Obfuscating a Program

- Obfuscating a program: making the program impossibly difficult to understand (to both humans and computers)
- Obfuscator: a "compiler" that makes program "unreadable"

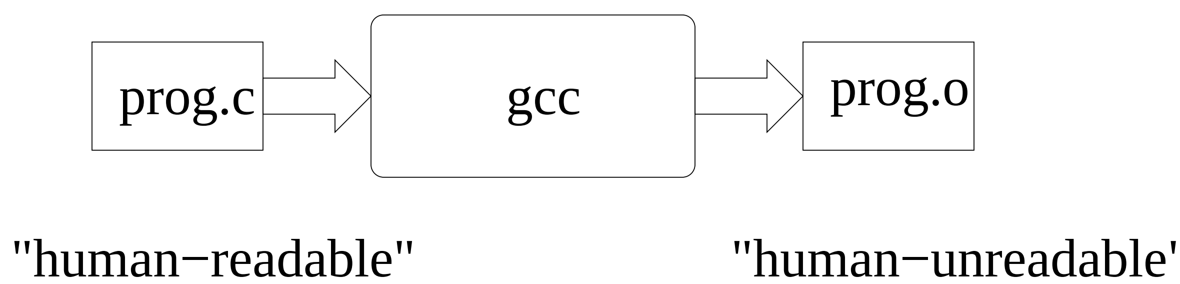


Carnegie Mellon

What do Obfuscators Look Like?

gcc

gcc converts any program into a “human-unreadable” form.



Carnegie Mellon

What do Obfuscators Look Like?, cont'd

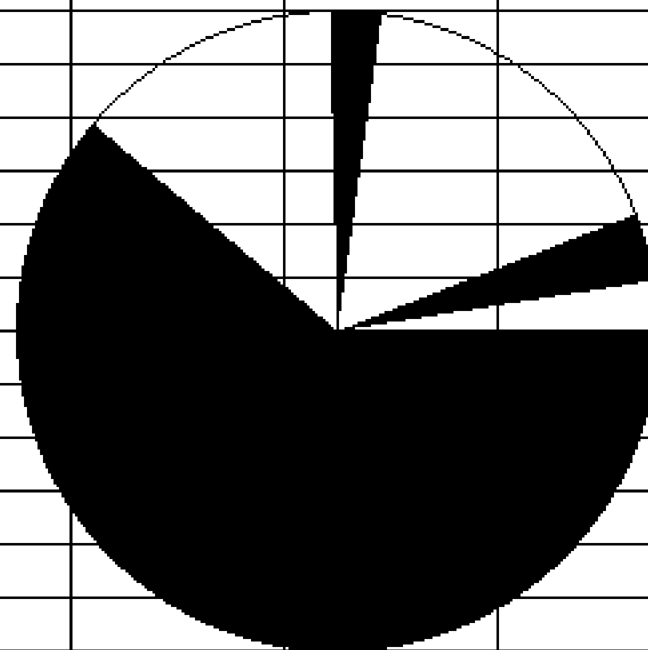
People who won the International Obfuscated C Code Contest

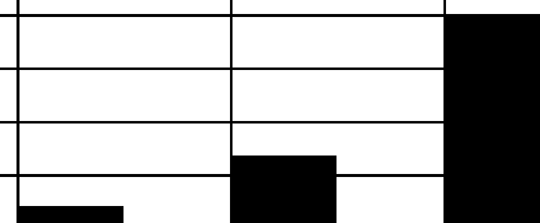
```
int q,P,W,Z,X,Y,r,u;char E[U][U][T+1],D[T];Window J;GC k;XEventw;Display*i; v(c,j ,K){
char*P=E[c][j],*X,g=0;double A=0,F=0,m[T];a;K<U&&*P;){m[g]=strtod(P,&X);a q=-1;++q <g;
F+=*P==64?e:0,A=*P==33?e>A? e:A :0);o isupper(*P)*isdigit(P[1])){o v(*P-65,atoi(P+1),K+1))
goto i;a m[g++]=atof(D);P++&&isdigit(*P););}else o P-X){g++;P=X;}else{i(=)i(+)i(-)i(*)
i(/)o *P-32)goto i;P++;}}o!--g)return!sprintf(D,"%10.2f",*m);i:a;A&&q--;XFillRectangle(
i,J,k,(n+q)*Q+S,s*S+S-K,Q/2,K))K=Q*e/A;a;q--&&F;A+=e){o q%2)t(White)XFillArc(i,J,k,I t(Black)
b(Arc)I}return!!strcpy(D,E[c][j]);}main(){read(q,E,z);i=XOpenDisplay(0);k=XCreateGC(i,
J=RootWindow(i,W),Z,0); XSelectInput(i,J=XCreateSimpleWindow(i,J,P,q,M,H,r,u,WhitePixel
(i,X)),ButtonPressMask|KeyPressMask|ExposureMask);a XMapWindow(i,J);;b(String)S,S,d,P=strlen
(d))){XNextEvent(i,&w);XLookupString(&w.xkey,D,1,&q,0);q&96&&q<128?d[P++]=q l C ? Y-- l L
? Y++ l V?X++ l _ ?d[--P]=0 l O&&X ? X--:P;o w.type==ButtonPress){X=r+(w.xbutton.x-S)/Q;
Y=u+w.xbutton.y/S-1;}X%=26;a X>r+p?r++ :X;X<r;r--);Y%=U;o Y<1)Y=1;a;q== R;exit(write(1 ,E,z)));
a Y>u+G ? u++ :u; Y<=u; u--); XClearWindow(i,J);a Z=u+1;sprintf(D,"%3d" ,Z)&&Z<=u+G;Z++)
{b(String)0,(s+2)*S-B,D,3);a%W=r;W<=r+p;b(String)n*Q+S+N,S*2-B,D,1),W++){%b(Rectangle)n*Q+S,
s*S+S,Q,S);v(W,Z,0);b(String)n*Q+S+N,(s+2)*S-B,D,strlen(D));*D=65+W;}}b(Rectangle)(X-r)*Q+S
+1,(Y-u) *S+S+1,Q-2,S-2);}}
```

Carnegie Mellon

Untitled

	A	B	C	D	E	F	G	H
1								
2	SALES 2000						D5 D6 D7 D8	D9 D10 @
3								
4	PRODUCT	SOLD	PRICE	REVENUE				
5	Product A	20400.00	230.00	4692000.00				
6	Product B	2291.00	449.00	1028659.00				
7	Product C	4.00	50000.00	200000.00				
8	Product D	4320.00	299.00	1291680.00				
9	Software	1750.00	150.00	262500.00				
10	Consulting	1611.00	120.00	193320.00				
11	TOTAL			7668159.00				
12								
13	COSTS (1998-2000)							
14								
15	ITEM	Year 1998	Year 1999	Year 2000				
16	Goods	2500230.00	3100200.00	3203200.00		PROFIT DEVEL	OPMENT	
17	Salaries	1430202.00	1536000.00	1636120.00		Year 1998	Year 1999	Year 2000
18	R & D	2000320.00	900000.00	950000.00				
19	Investments	560000.00	1103000.00	800000.00				
20	Taxes	500.00	700.00	12333.00				
21	TOTAL	6491252.00	6639900.00	6601653.00				
22	SALES	6602000.00	7002000.00	7668159.00				
23	PROFIT	110748.00	362100.00	1066506.00		B23 C23 D23	!	





Carnegie Mellon

Why do People do Obfuscation?

- FUN
- Security through obscurity

  "If you don't understand it, you can't mess around with it."
- Intellectual Property Protection

  "If you don't understand it, you can't steal it."

Who Makes Obfuscators?

- cloakware.com
  "Tamper Resistant Software"
- Microsoft
  Protecting parts of its OS from reverse engineering.

OBFUSCATED

Carnegie Mellon

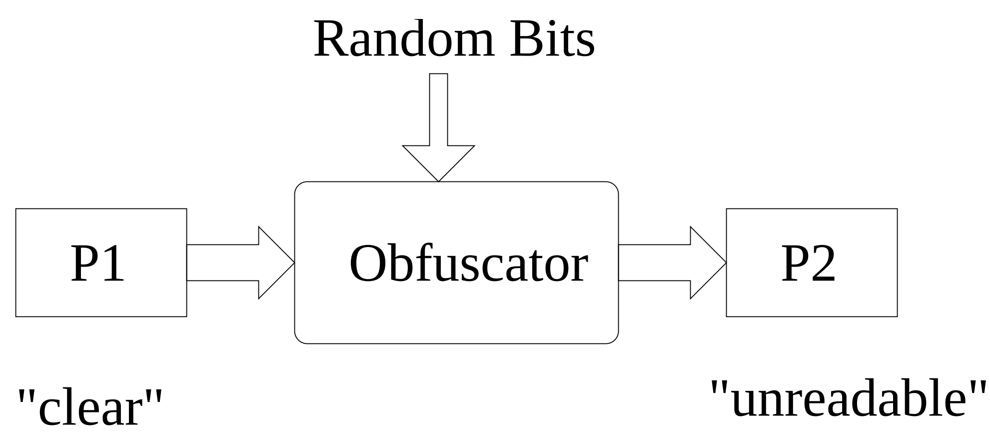
This Talk

We investigate the theoretical notion of program obfuscation.

- What are obfuscators?
- Why do we want them so badly?
- Why are they too good to be true?

What is an Obfuscator? — Intuitions

- An Obfuscator is an efficient, randomized compiler.



- $P1$ and $P2$ compute the same function.
- $P2$ is **unreadable**.

Unreadable Programs?

A program is always executable.

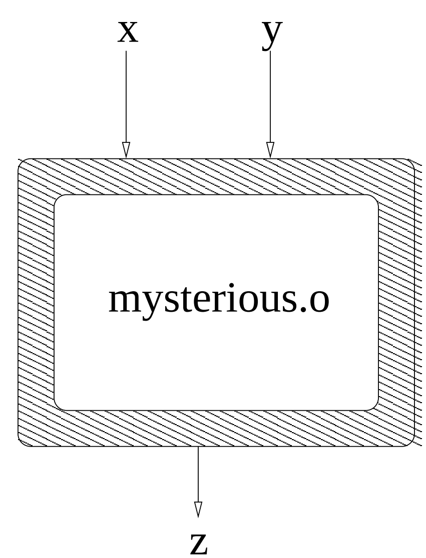
But what does it mean to say it is unreadable?

We need to make a distinction between Black Boxes and source codes ...

What can you do with a Black Box?

mysterious.h

```
/* This function does weird
   operations to input x and
   y and outputs a mysterious
   number
*/
int mysterious(int x, int y);
```



Very limited information...

- Input-output behavior
- Running time

Carnegie Mellon

What can you do with the source code?

mysterious.c

```
int mysterious(int x, int y)
{
  int z;
  z = x + y;
  return z;
}
```

Carnegie Mellon

A Source Code Analyzer Can do More...

Static Analysis: basic blocks, variable usage ...

Dynamic Analysis: stacks, program flow ...

Efficiency Analysis: statistics, hot-spots ...

Mutational Analysis: change fragments of the program.

Ana and BAna

We are interested in 2 types of polynomial-time analyzers:

- Ana is a source-code analyzer that can read the program.

$$\text{Ana}(P)$$

- BAna is a black-box analyzer that only queries the program as an oracle.

$$\text{BAna}^{P}(\texttt{time}(P))$$

Carnegie Mellon

Ana vs. BAna

Ana seems more powerful than BAna...

- Ana can simulate BAna.
- Furthermore, Ana can obtain information that BAna cannot get, like the prgram flow.

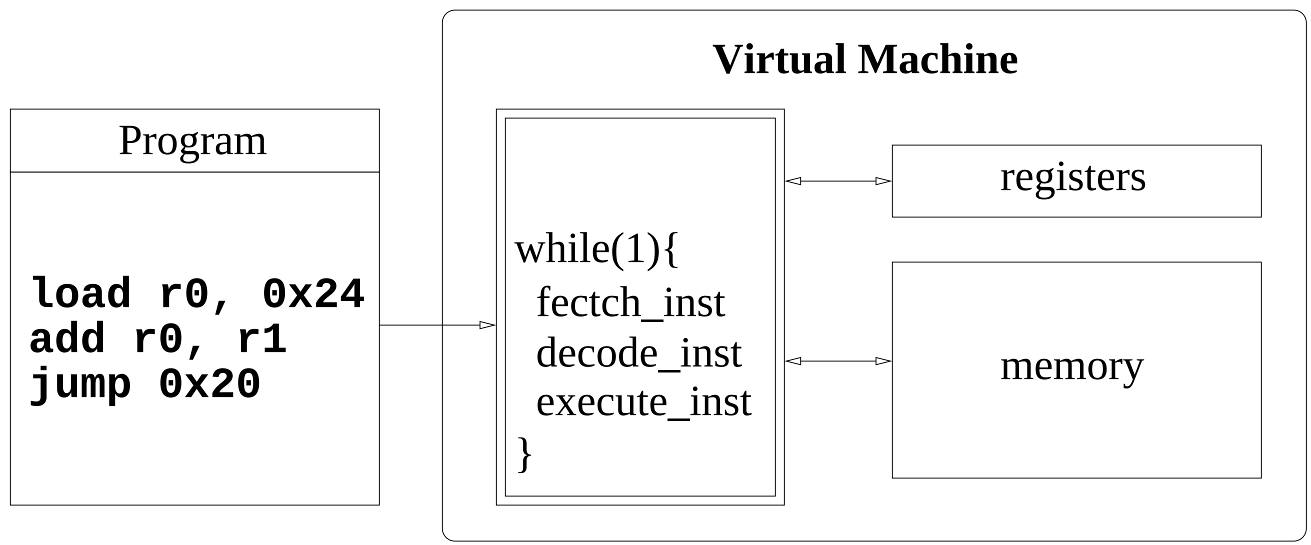
Is it true that Ana is always strictly more powerful than BAna, even for programs that are deliberately rendered "unreadable"?

Carnegie Mellon

Case Study: How to Make Instruction Trace Useless

Instruction Trace: The sequence of the instructions executed.

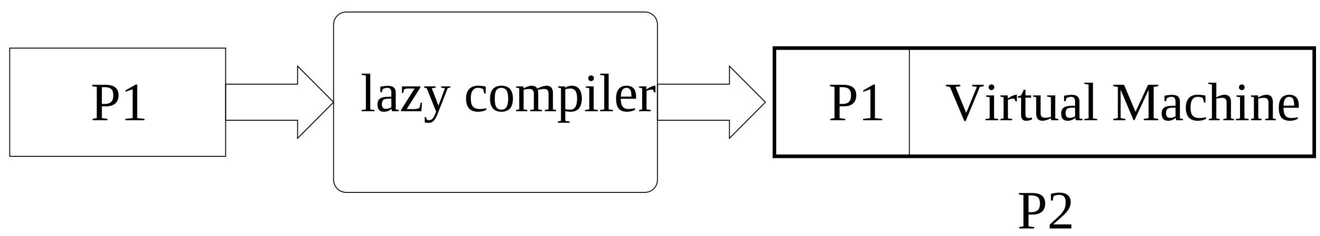
Virtual Machine: An “interpreter” that executes the input program.



Carnegie Mellon

Case Study, cont'd

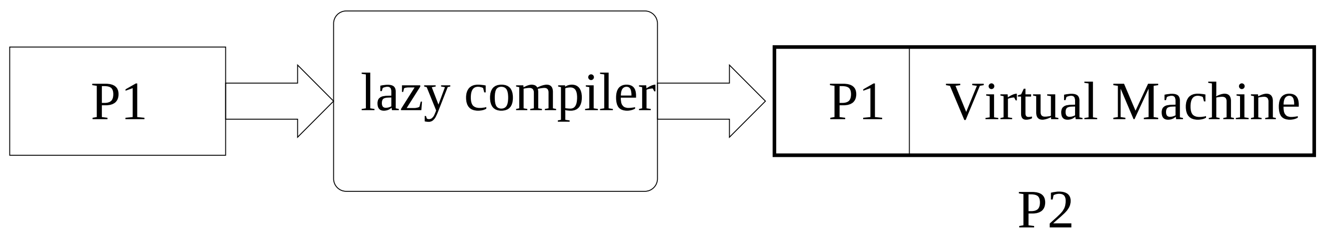
Consider the following lazy compiler...



The output of the compiler is the Virtual Machine with P1 hardwired.

The instruction trace of P2 is the trace of the Virtual Machine, which is the same for different P1's.

Useless Instruction Trace



The instruction trace of $P2$ is always the same for different $P1$'s of the same running time.

So BAna can generate the trace without even knowing the program!

More Generally...

A compiler can hide a lot of features from the source code...

- Create dummy variables, so the number of variables is the same for all programs of the same running time.

- Add dummy code, so each piece of the program uses the same amount of time.

...

Is Ana Always More Powerful?

It seems that you can always change your program to “hide” information from Ana.

Intuition: Ana isn’t necessarily more powerful than BAna for unreadable programs.

Virtual Black-Box

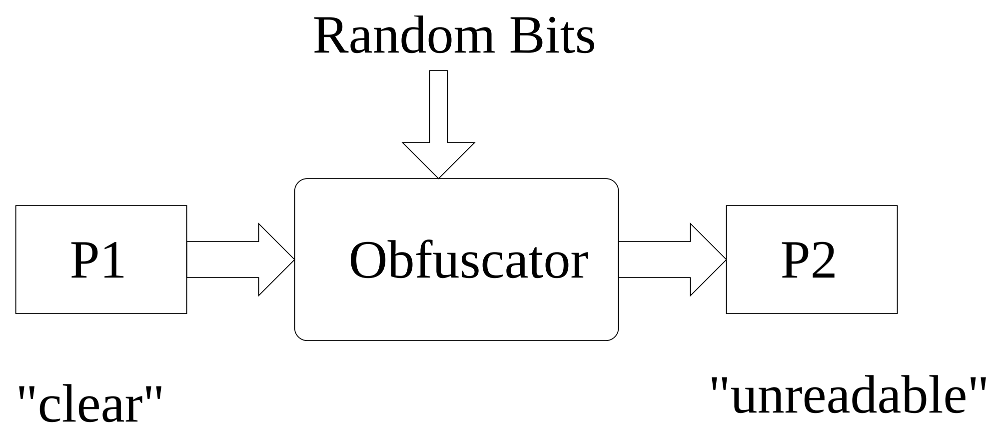
An unreadable program is like a virtual black-box.

Anything Ana can do, BAna can do as well...

and an obfuscator converts any program into a virtual black-box!

What's an obfuscator? — a Semi-formal Definition

- An Obfuscator is an efficient, randomized compiler.



- $P1$ and $P2$ compute the same function.
- For every Ana, there is a BAna, such that

$$\text{Ana}(P2) \approx \text{BAna}^{P2}(\texttt{time}(P2))$$

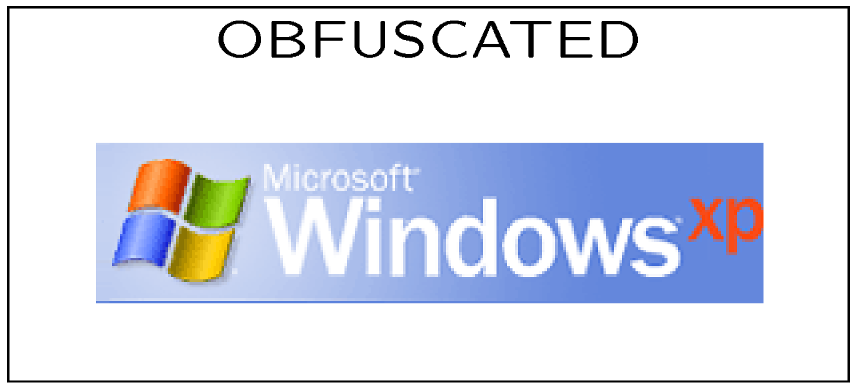
Carnegie Mellon

Why Do We Want Obfuscators That Much?

We have seen a definition of obfuscators

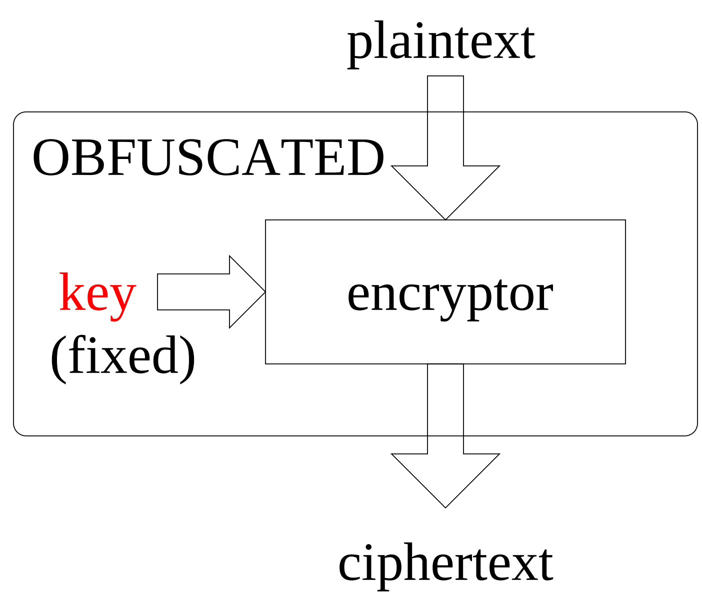
What can we do with them?

Obfuscator for Code Protection



A strong guarantee that nobody can understand your program.

Converting Any Private-Key System to a Public-Key System



- The public key is the obfuscated encryptor with a fixed key
- The private key is the key

Carnegie Mellon

cloakware.com

They used their Temper-Resistant Technology on obfuscating a DES encryptor.

"... the task of de-cloaking a Data Encryption Standard application is shown to be computationally infeasible."

Why People Want Obfuscators so Badly?

Obfuscators will imply a lot of cryptographical applications that no one knows how to do now.

- Private-key to Public-key convertion
- Homomorphic Encryption
- Removal of the Random Oracle

Why are Obfuscators Too Good to be True?

We have see what one can do obfuscators

However, they are too good to be true...

A Formal Definition for Obfuscators

An Obfuscator ($\mathcal{O}$) is a polynomial-time, randomized algorithm, which takes input $P1$ as the encoding of a Turing Machine, and outputs the encoding of an equivalent Turing Machine $P2$:

$$\mathcal{O}(P1) = P2$$

- Polynomial Slowdown: There exists a polynomial $p(\cdot)$, s.t. $\texttt{time}(P2) \leq p(\texttt{time}(P1))$.
- Virtual Black-Box Condition: For any Ana, there exists a BAna, such that for any Turing Machine $P1$ and its obfuscated version, $P2 = \mathcal{O}(P1)$:

$$|\Pr[\,\text{Ana}(P2) = 1] - \Pr[\,\text{BAna}^{P2}(1^{\texttt{time}(P2)}) = 1]\,| \leq 1/poly$$

Carnegie Mellon

“Straw-man Definition?”

Am I cheating by presenting a definition that’s too strong?

- There are many possible definitions we considered
- Empirically, this one is the “minimal” definition:
  - All other definitions imply this one.
  - An even weaker definition probably won’t have provable cryptographical applications.

Secret-Leaking Functions

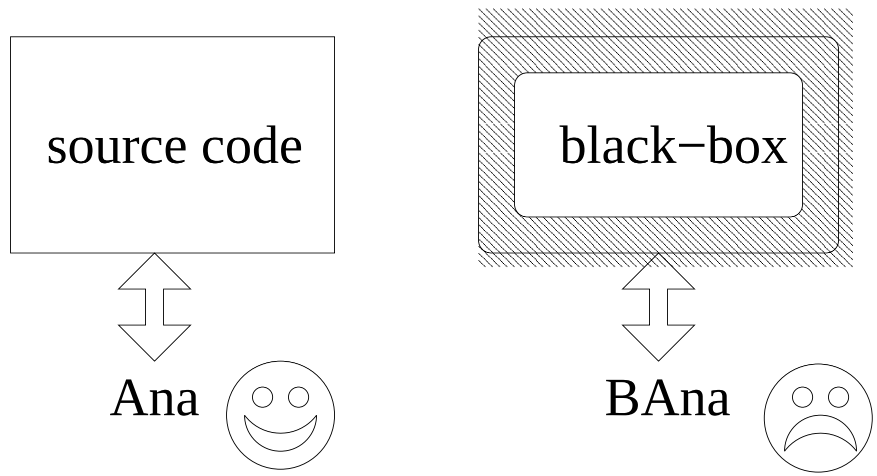
There exists efficient functions $\{f_s\}$ such that

- Each f_s contains a "secret" s
- No BAna using f_s as an oracle can obtain the secret with high probability
- But **any** program that computes f_s will "leak" the secret !

The existence of secret-leaking functions will imply that obfuscators don't exist!

Carnegie Mellon

How to Leak Your Secret?

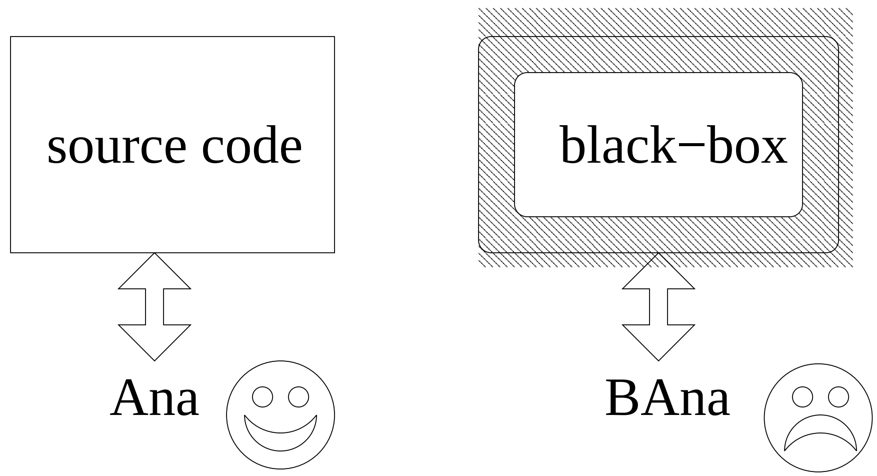


The secret-leaking function cannot leak the secret to BAna...

but **any** source code will leak the secret to Ana!

Carnegie Mellon

Mission Impossible?



Simple approaches don't work

- Encode the secret as comments in source code.

  Doesn't work for every source code.

- The function outputs the secret if you give the correct input.

  How do you make sure that Ana knows the correct input, and BAna doesn't?

Carnegie Mellon

Correct Input?

We need the correct input to be...

- Obtainable from **any** source code.
- But not obtainable via black-box access.

What's the Correct Input?

How about making **the source code itself** the correct input?

Cannibalistic Function (intuition)

"Feed me somebody that behaves like me, and I'll leak my secret!"

```
FUNC CANNIBAL (Prog)
IF (Prog behaves like me)
THEN OUTPUT secret;
ELSE OUTPUT ``0'';
```

- Without the source code, BAna cannot produce a Prog that behaves like CANNIBAL.
- But Ana can since she has the source code for CANNIBAL, which behaves **exactly** like CANNIBAL!

Carnegie Mellon

Formal Definition

The function CANNIBAL consists 2 parts: ID and Leaker.

$$\text{ID}_{\alpha,\beta}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Leaker}_{\alpha,\beta,s}(P) = \begin{cases} s & \text{if } P(\alpha) = \beta \\ 0 & \text{otherwise} \end{cases}$$

ID has the correct "behavior" of CANNIBAL.

Leaker will output the secret s only when the input program P has the correct behavior.

$$\text{Leaker}_{\alpha,\beta,s}(\text{ID}_{\alpha,\beta}) = s$$

Carnegie Mellon

Putting 2 functions together

We combine 2 functions into one single function.

$$\text{CANNIBAL}_{\alpha,\beta,s}(y,b) = \begin{cases} \text{ID}_{\alpha,\beta}(y) & \text{if } b = 0 \\ \text{Leaker}_{\alpha,\beta,s}(y) & \text{if } b = 1 \end{cases}$$

Carnegie Mellon

How Ana can obtain the secret

STEP 1: generate the source code for ID.

```
cannibal(char* y, int b){
  int my_variable;
  ...
}
```

→

```
string ID = "
  ID(char *y){\
    const int b = 0;\
    int my_variable;\
    ...
  }\
";
```

STEP 2: run cannibal on ID to get the secret.

```
secret = cannibal(ID, 1);
```

Carnegie Mellon

BAna cannot learn much from CANNIBAL

$$\mathrm{ID}_{\alpha,\beta}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ 0 & \text{otherwise} \end{cases}$$

$$\mathrm{Leaker}_{\alpha,\beta,s}(P) = \begin{cases} s & \text{if } P(\alpha) = \beta \\ 0 & \text{otherwise} \end{cases}$$

BAna only makes polynomially many queries.

If α, β, and s are all chosen randomly, the probability to find them is exponentially small.

Carnegie Mellon

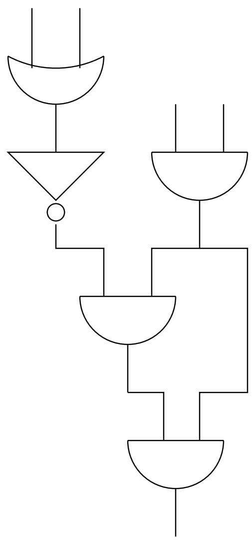
Putting Everything Together...

$$\text{CANNIBAL}_{\alpha,\beta,s}(y,b) = \begin{cases} \text{ID}_{\alpha,\beta}(y) & \text{if } b = 0 \\ \text{Leaker}_{\alpha,\beta,s}(y) & \text{if } b = 1 \end{cases}$$

- There exists an efficient Ana that always learns s.
- No polynomial-time BAna can learn s with high probability
- $\{\text{CANNIBAL}_{\alpha,\beta,s}\}$ is a family of **secret-leaking functions**!
- No obfuscators exist for CANNIBAL.

Carnegie Mellon

Impossibility Results for the Circuit Model



- We just proved the impossibility results for the Turing Machine model
- The result holds for the Circuit model as well, though the proof is trickier.
- Since a circuit cannot eat itself, you have to chop it into pieces and feed them to the circuit piece-by-piece.

Carnegie Mellon

What did we Just Prove?

It is impossible to design a general-purpose obfuscator for any function.

How about special-purpose obfuscators for some natural classes of cryptographical functions?

For example: private-key encryption functions.

Carnegie Mellon

Secret Leaking Private Key Systems

"Feed me somebody that behaves like me, and I'll leak my secret key!"

CANNIBAL_ENCRYPTOR (X)

```
IF (X behaves like me)
THEN OUTPUT secret_key;
ELSE OUTPUT encrypt(X);
```

- CANNIBAL_ENCRYPTOR is a secure private-key system if used as a black-box.
- Any source-code implementation of CANNIBAL_ENCRYPTOR is insecure.

Carnegie Mellon

More Impossibility Results on Obfuscation

There don't exist obfuscators for:

- Encryption schemes
- Digital Signature schemes
- Pseudorandom Functions
- Message Authentication Codes (MAC).

Conclusions

- Definitions of Obfuscators (virtual black-box property)
- Applications for Obfuscators
- General-purpose obfuscators don't exist.
- The impossibility results hold for obfuscating natural cryptographical functions

Any questions?

Paper available at

http://www.cs.cmu.edu/~yangke/papers/obfuscator.ps

Slides available at

http://www.cs.cmu.edu/~yangke/papers/obf-talk.ps

Ke Yang (412-268-7571)

yangke@cmu.edu

Carnegie Mellon