

# Online Synthesis of Adaptive Side-Channel Attacks Based On Noisy Observations

Lucas Bang, Nicolás Rosner, Tevfik Bultan

Verification Lab (VLab)  
Department of Computer Science  
University of California Santa Barbara

# Adaptive Side-Channel Attacks

Secret Data

Program

```
1 private s = getBufferSize();
```

# Program

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

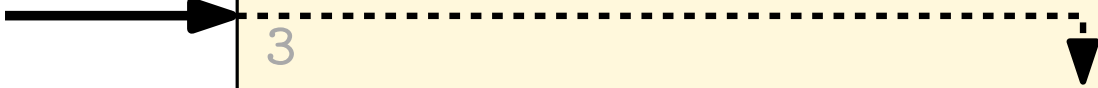
input, *i*



```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

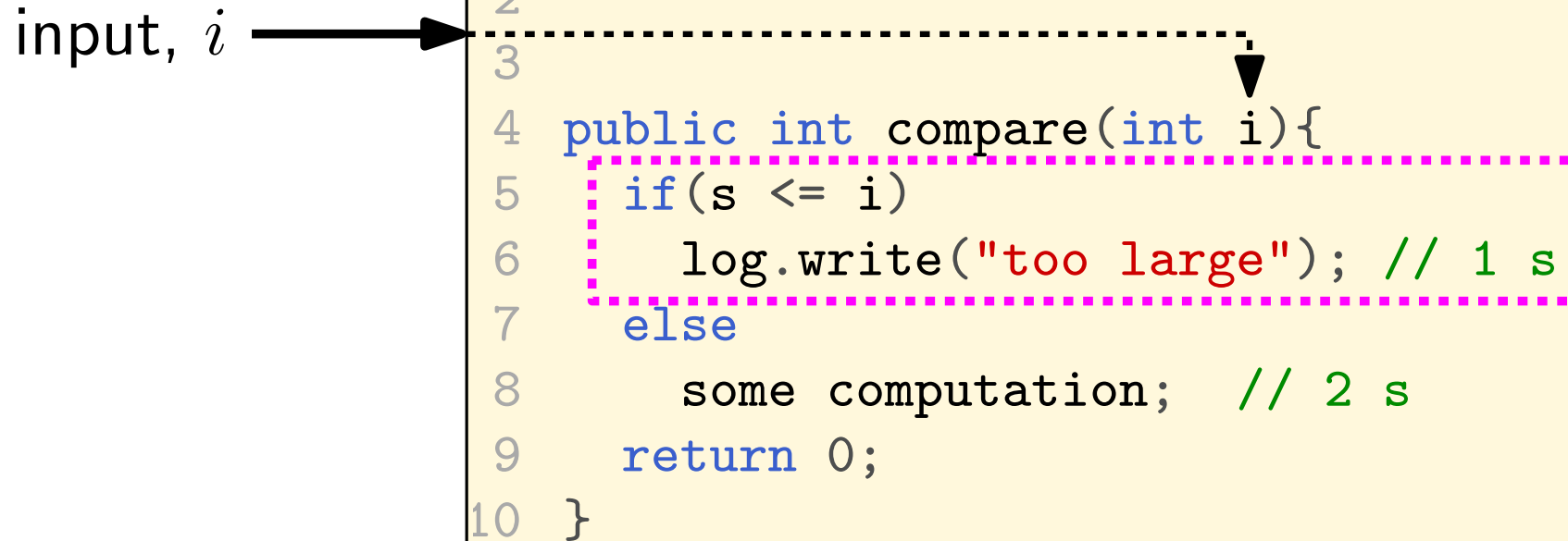
input, *i*

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

A diagram illustrating variable flow. A solid black arrow points from the text 'input, i' to the parameter 'i' in the function signature 'compare(int i)'. A dashed black arrow points from the parameter 'i' in the function signature to the variable 'i' in the condition 'if(s <= i)'. A solid black arrow points from the variable 'i' in the condition to the variable 'i' in the function signature.

input, *i*

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```





input,  $i$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

$$s \leq i \Rightarrow o = 1$$

input,  $i$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

$$s \leq i \Rightarrow o = 1$$

input,  $i$


```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

input,  $i$

$1 \leq s \leq 8$

  $s?$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

input,  $i$

$1 \leq s \leq 8$



$s?$

output,  $o$


```

1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }

```

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

input,  $i$   
 $1 \leq s \leq 8$   
  $s?$

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```


~~output, 0~~

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

input,  $i$  →

$1 \leq s \leq 8$

  $s?$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

input,  $i$   
 $1 \leq s \leq 8$



$s?$



```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$



```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

input,  $i$

$$1 \leq s \leq 8$$



$s?$



observe time

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

input,  $i$

$$1 \leq s \leq 8$$



$s?$



observe time

$$s \leq i \Rightarrow o = 1$$

$$s > i \Rightarrow o = 2$$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

input,  $i$

$$1 \leq s \leq 8$$



$s?$



observe time

$$o = 1 \Rightarrow s \leq i$$

$$o = 2 \Rightarrow s > i$$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

input, 4

$$1 \leq s \leq 8$$



s?



observe time

$$o = 1 \Rightarrow s \leq i$$

$$o = 2 \Rightarrow s > i$$

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

input, 4

$$1 \leq s \leq 8$$



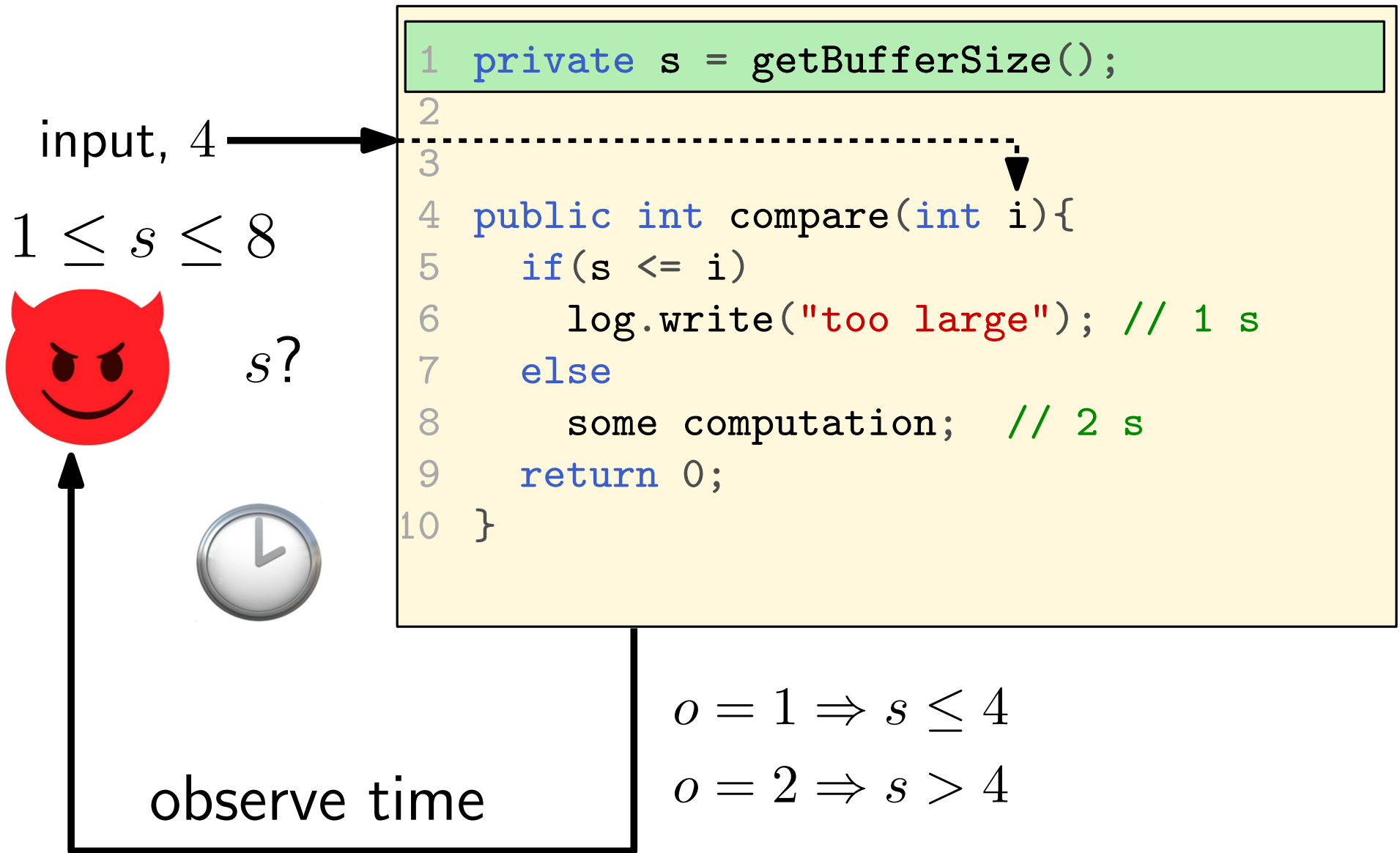
s?



observe time

$$o = 1 \Rightarrow s \leq 4$$

$$o = 2 \Rightarrow s > 4$$



Attacker can binary search on  $s$  using  $i$  and  $o$ .

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

Hardware + OS



Network

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

Hardware + OS



s?

Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

Hardware + OS



$s?$

input,  $i$



Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

Hardware + OS



$s?$

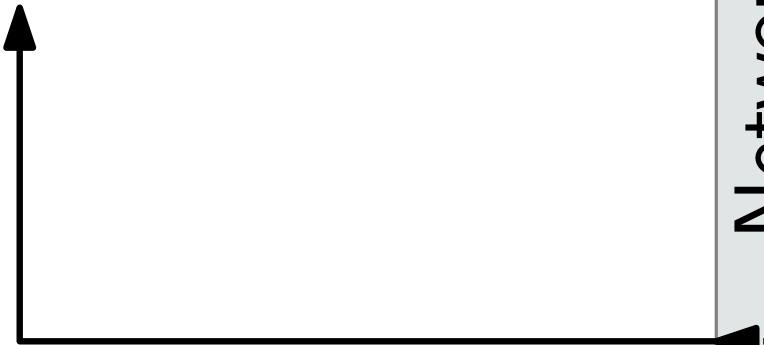
input,  $i$



Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

Hardware + OS





$s?$

input,  $i$



Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

Hardware + OS





$s?$

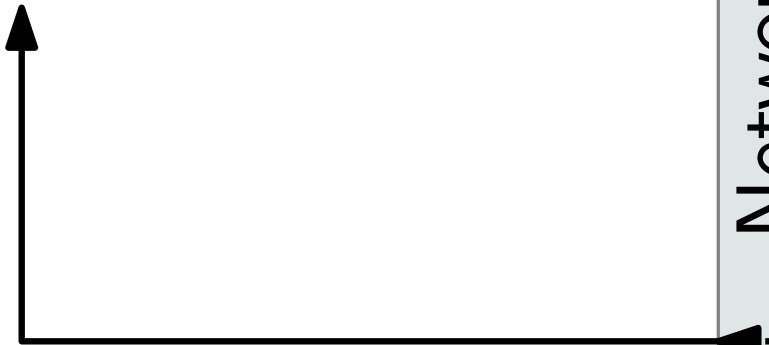
input,  $i$



Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

Hardware + OS





$s?$

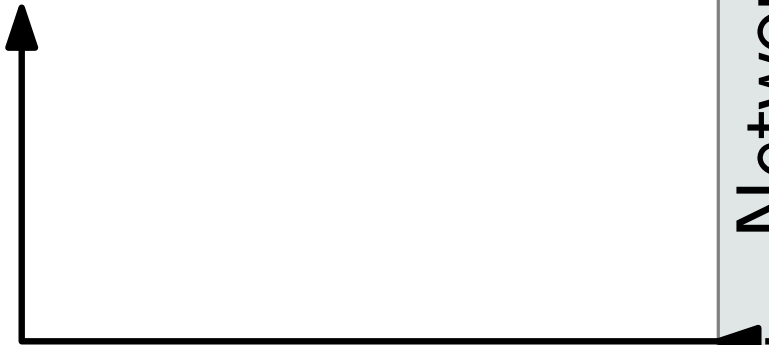
input,  $i$



Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

Hardware + OS



$$s \leq i \Rightarrow o = 1$$



$s?$

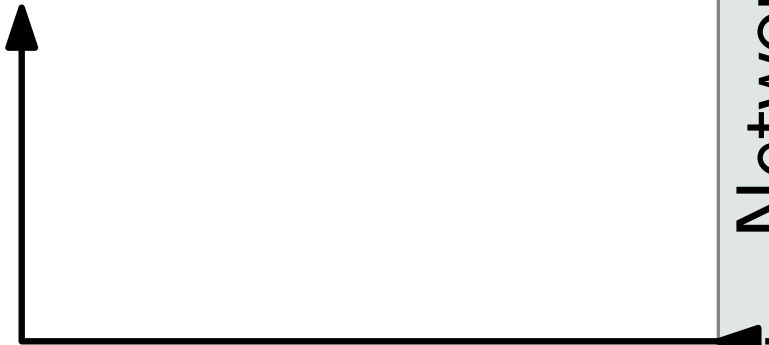
input,  $i$



Network

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

Hardware + OS



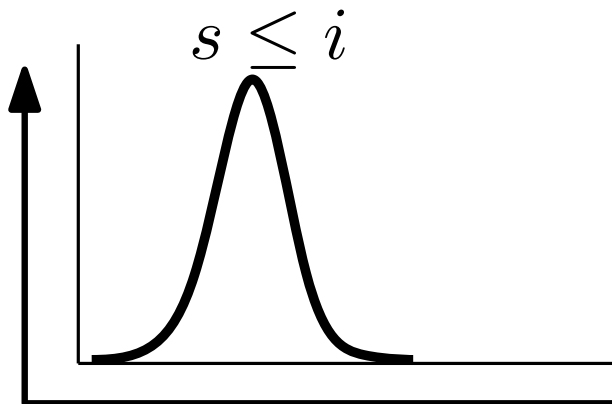
~~$s < i \rightarrow 0 = 1$~~





$s?$

input,  $i$



Network

```
1 private s = getBufferSize();
2
3
4 public int compare(int i){
5     if(s <= i)
6         log.write("too large"); // 1 s
7     else
8         some computation; // 2 s
9     return 0;
10 }
```

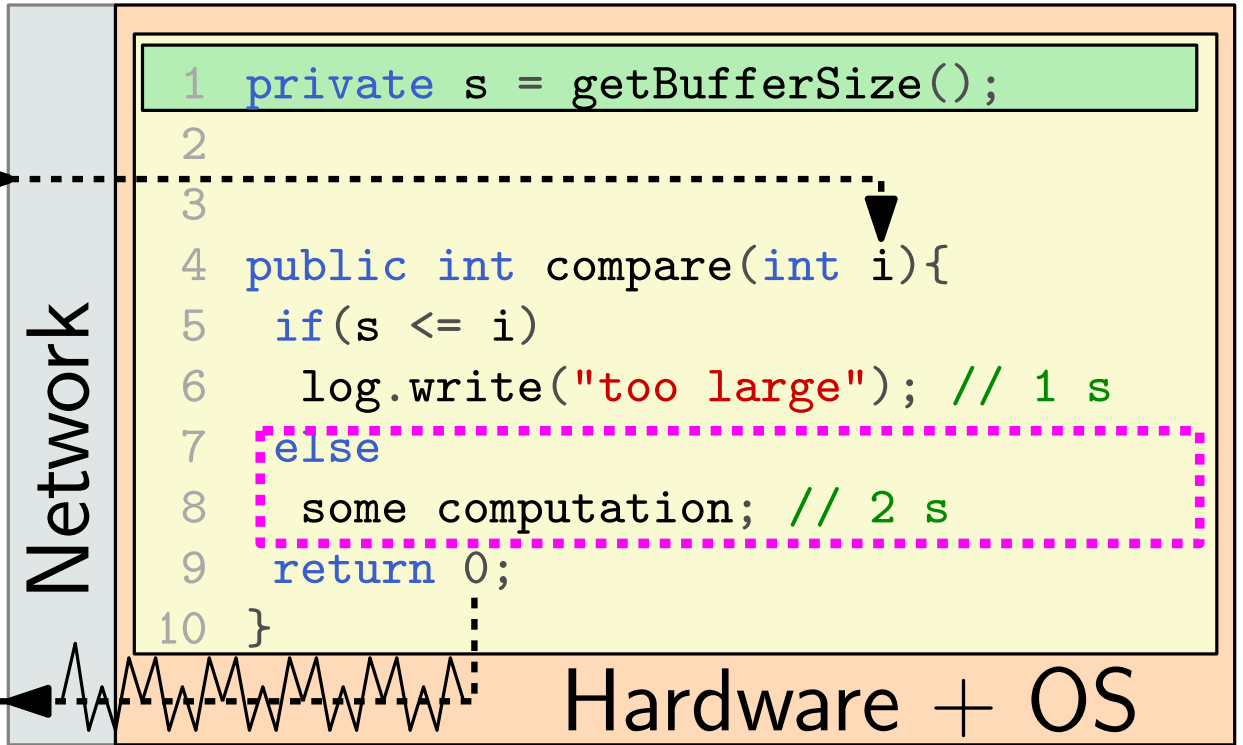
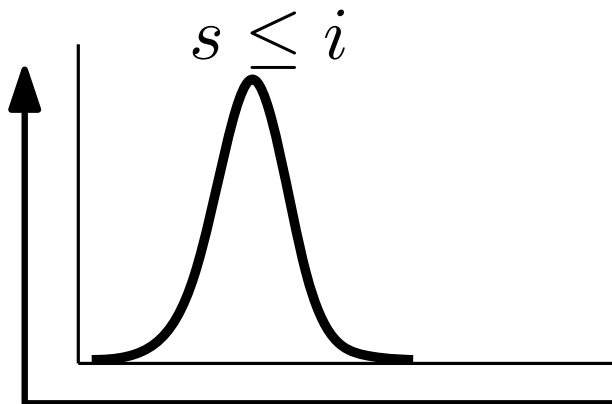
Hardware + OS

~~$s < i \rightarrow 0 = 1$~~



$s?$

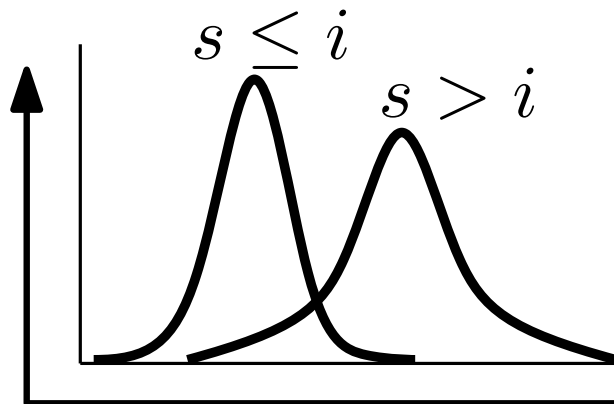
input,  $i$





$s?$

input,  $i$



Network

```
1 private s = getBufferSize();  
2  
3  
4 public int compare(int i){  
5     if(s <= i)  
6         log.write("too large"); // 1 s  
7     else  
8         some computation; // 2 s  
9     return 0;  
10 }
```

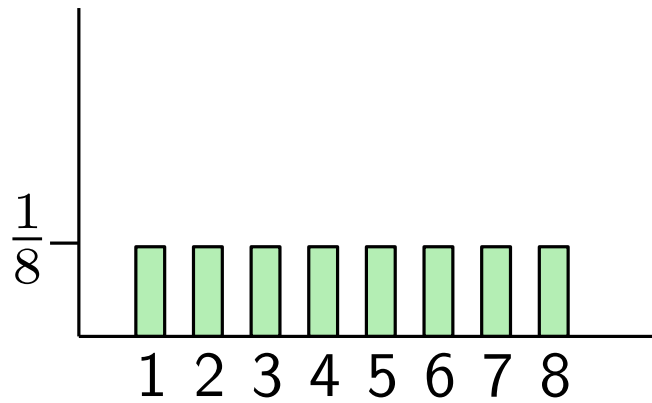
Hardware + OS



Attacker Belief?



# Attacker Belief?

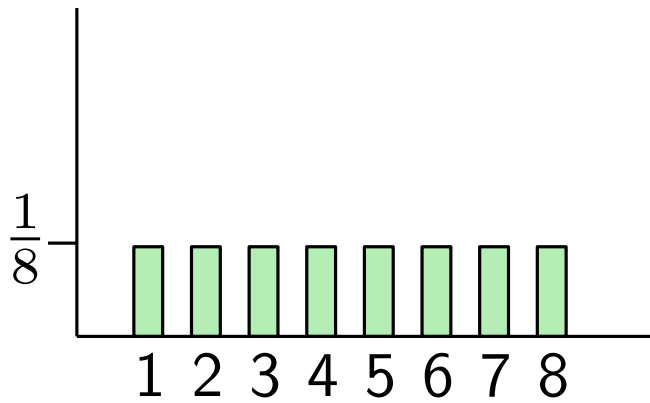


Attacker Belief?

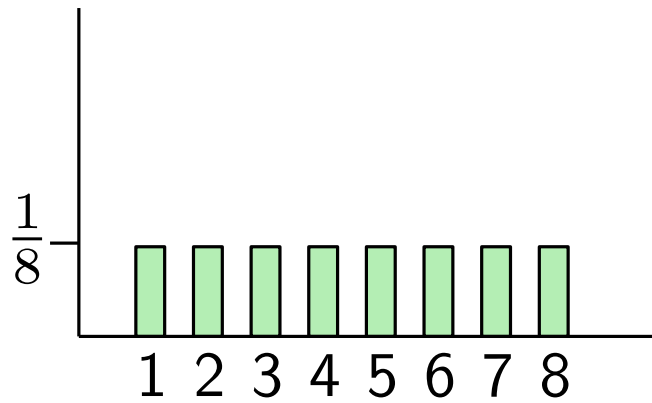
Input Choice?



$i^*$



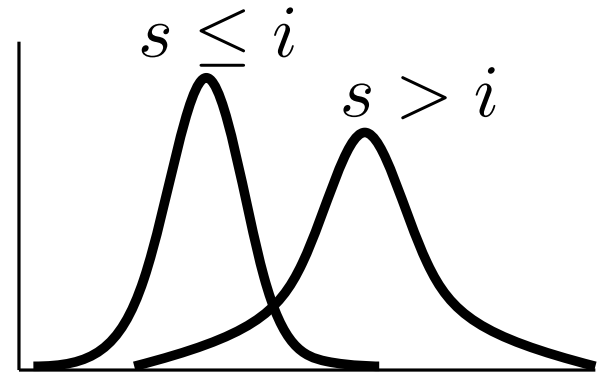
Attacker Belief?



Input Choice?

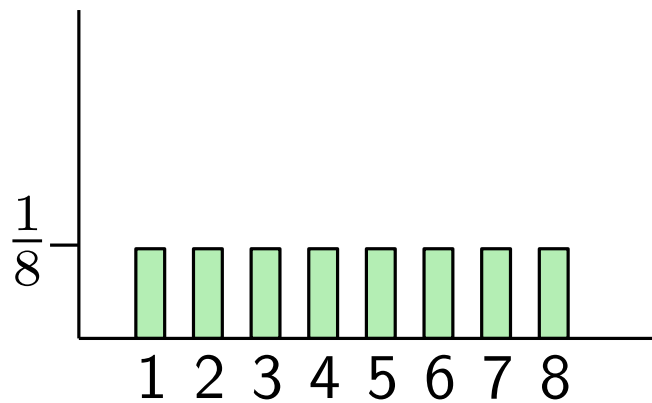
$i^*$

Observation?





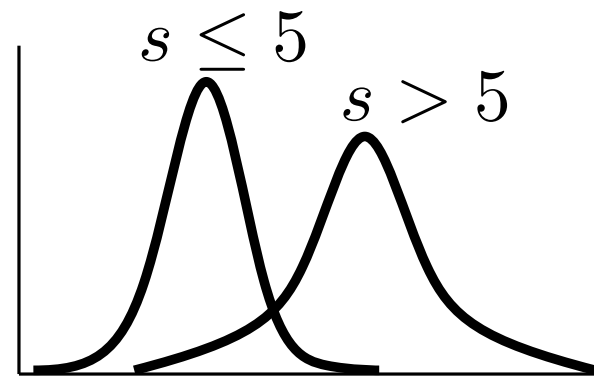
Attacker Belief?



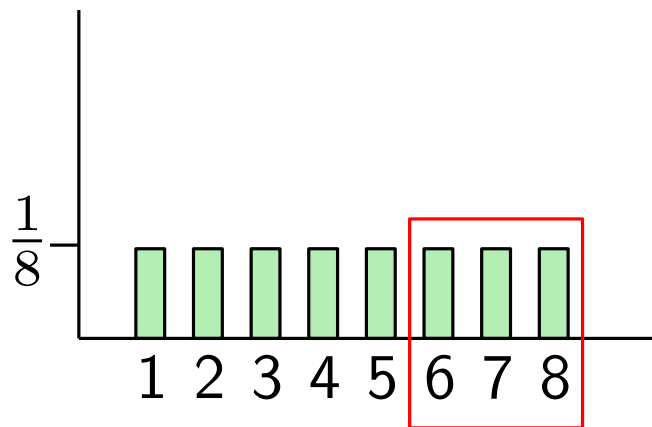
Input Choice?

$$i^* = 5$$

Observation?



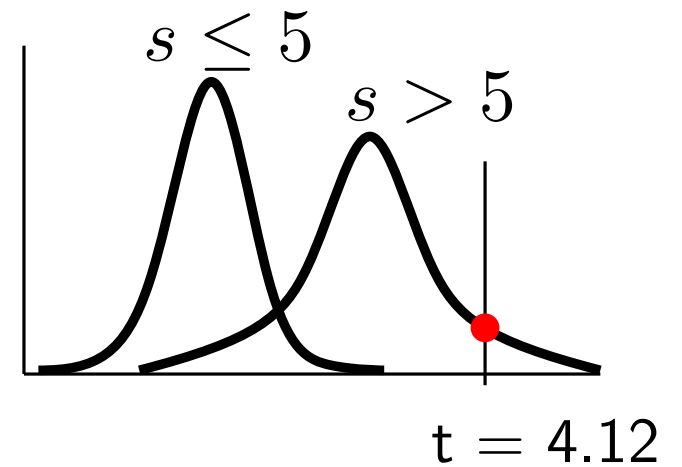
Attacker Belief?



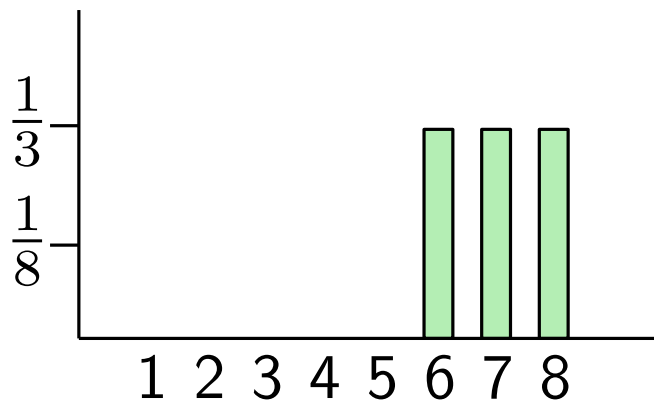
Input Choice?

$$i^* = 5$$

Observation?



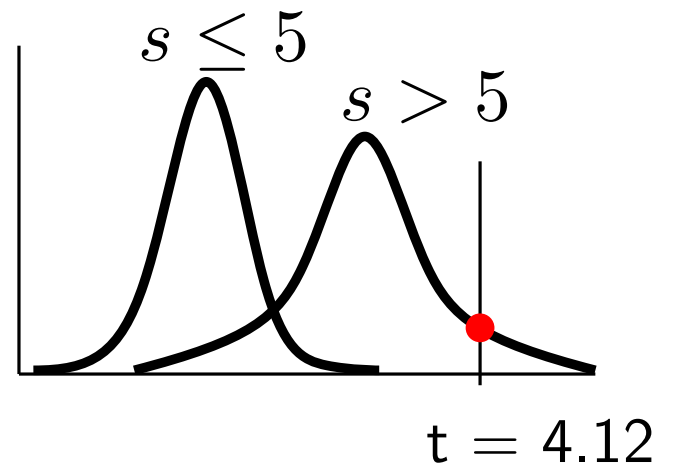
Attacker Belief?



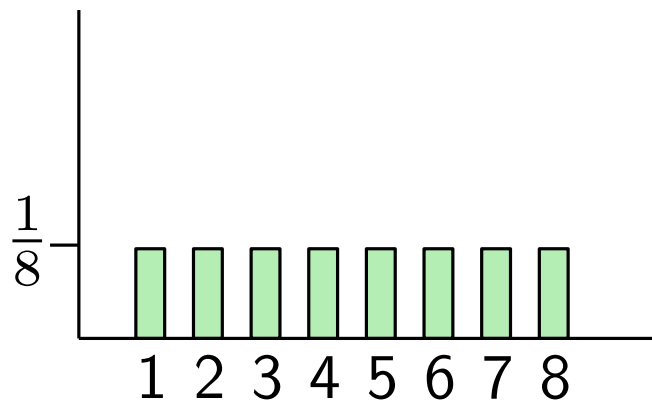
Input Choice?

$$i^* = 5$$

Observation?



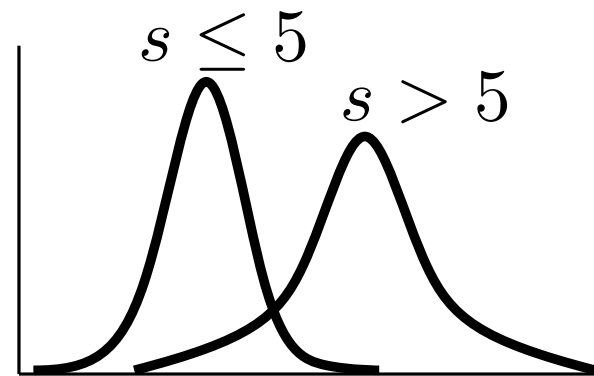
Attacker Belief?



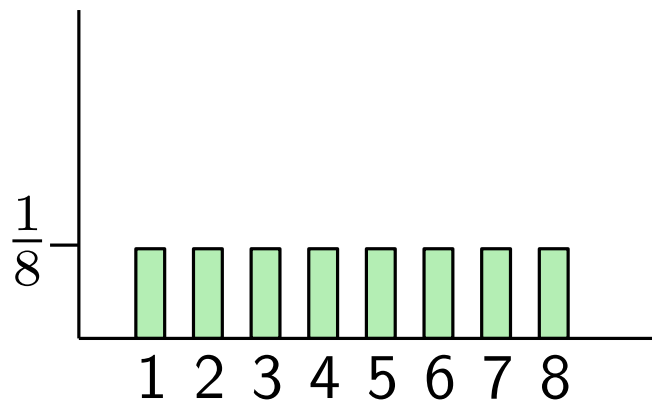
Input Choice?

$$i^* = 5$$

Observation?



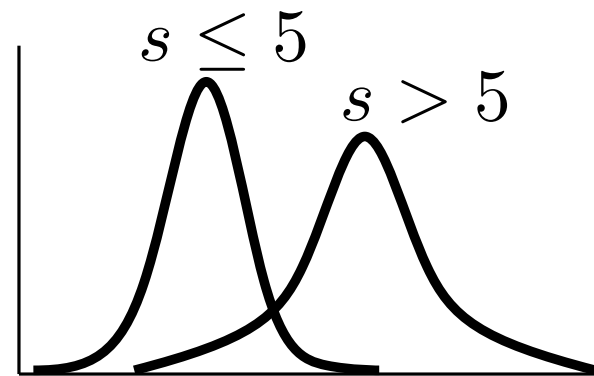
Attacker Belief?



Input Choice?

$$i^* = 5$$

Observation?



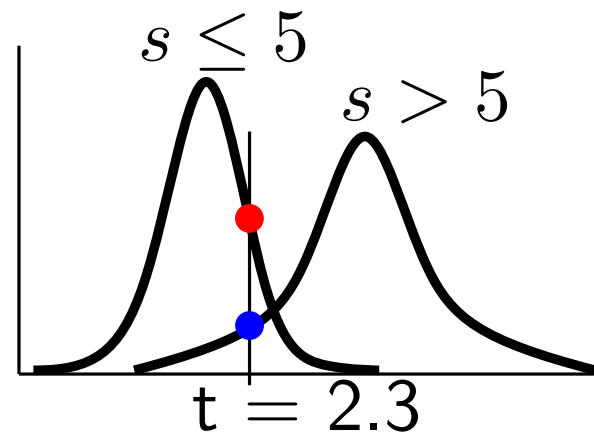
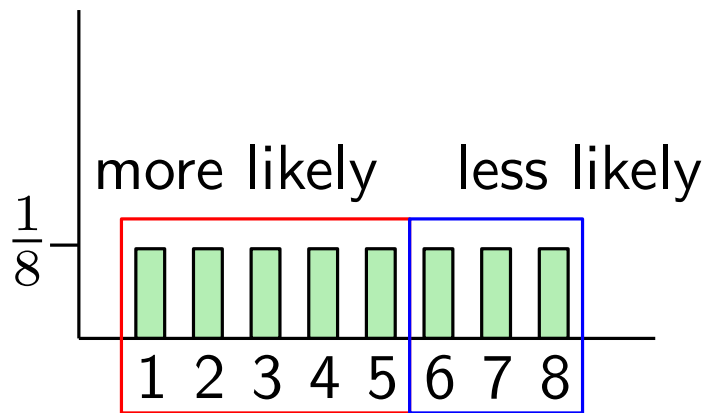
Attacker Belief?

Input Choice?

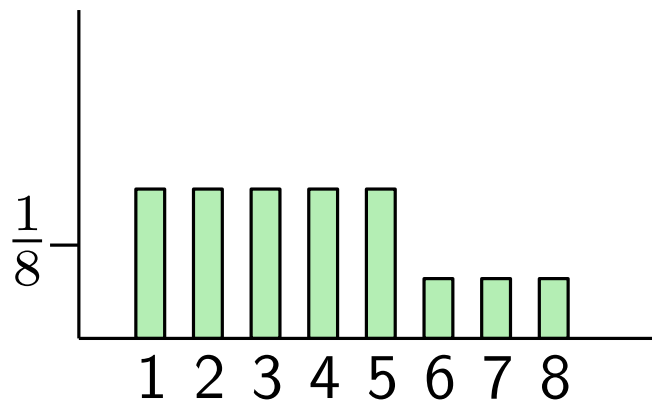
Observation?



$$i^* = 5$$



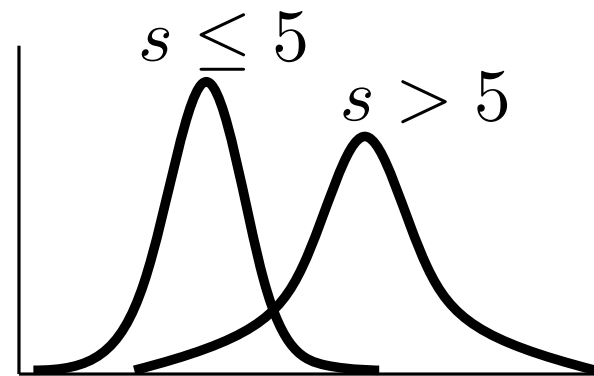
Attacker Belief?



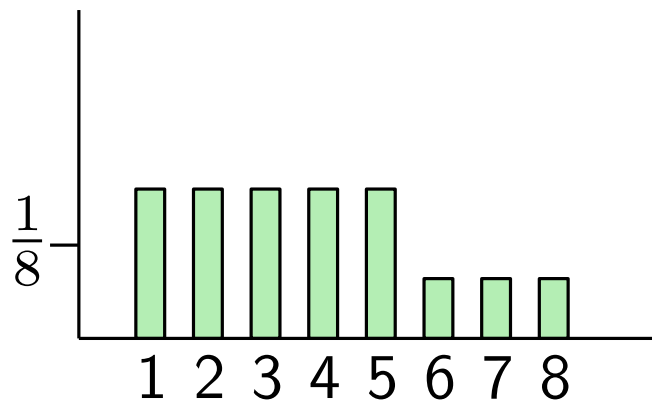
Input Choice?

$$i^* = 5$$

Observation?



Attacker Belief?

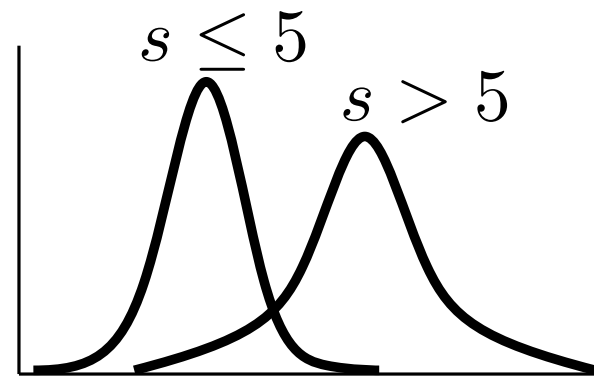


$$p(s|o, i^*)$$

Input Choice?

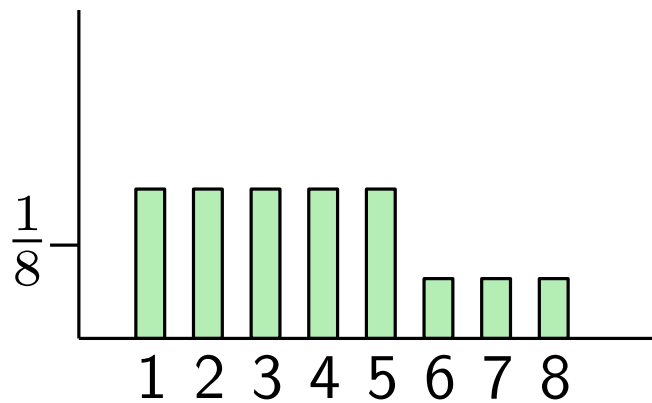
$$i^* = 5$$

Observation?





Attacker Belief?

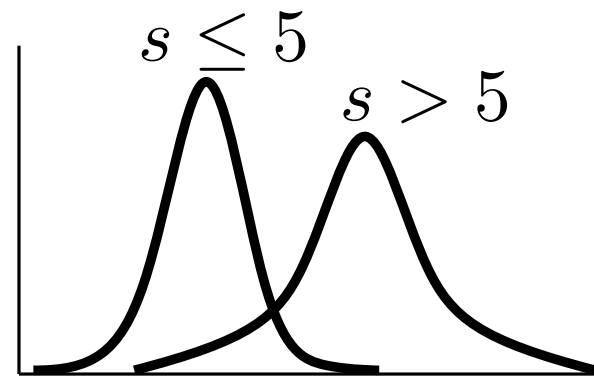


$$p(s|o, i^*)$$

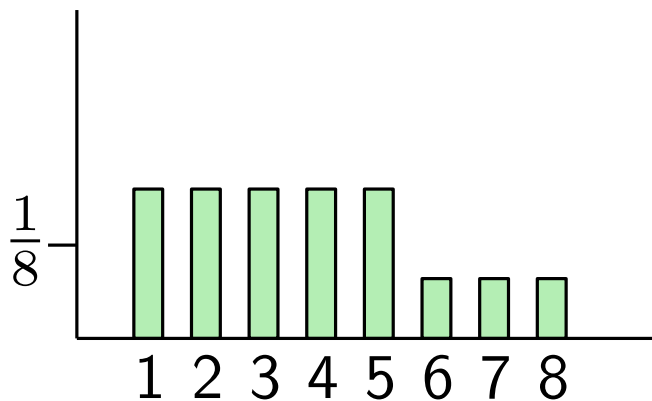
Input Choice?

$$i^* = 5$$

Observation?



Attacker Belief?

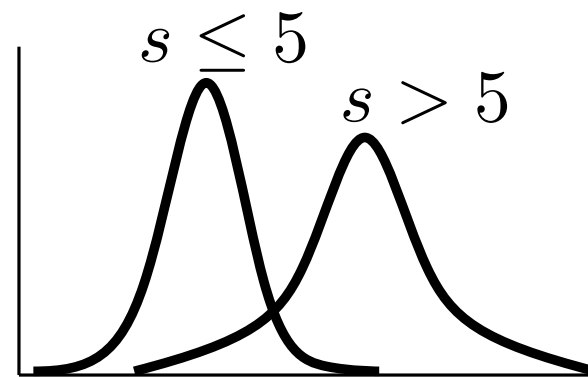


$$p(s|o, i^*)$$

Input Choice?

$$i^* = 5$$

Observation?



$$p(o|s, i)$$

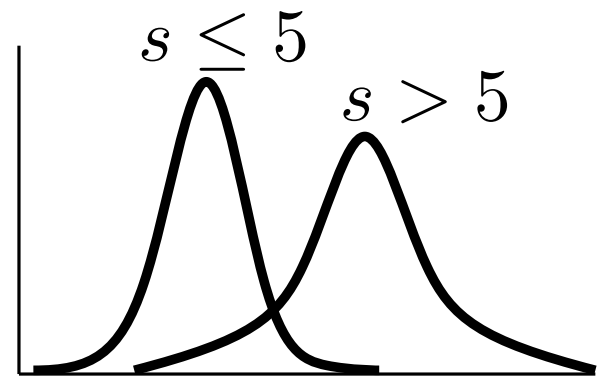
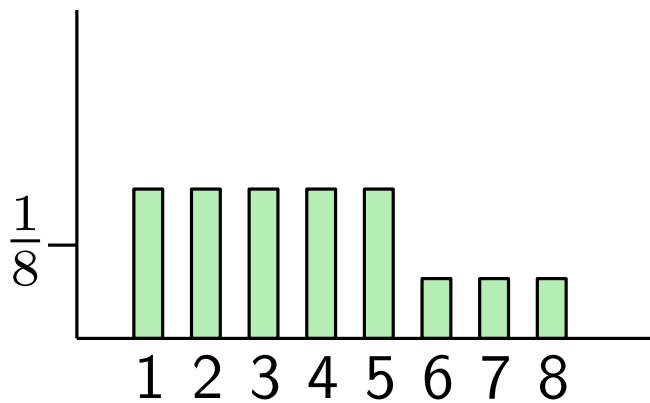
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$



$$p(s|o, i^*)$$



$$p(o|s, i)$$

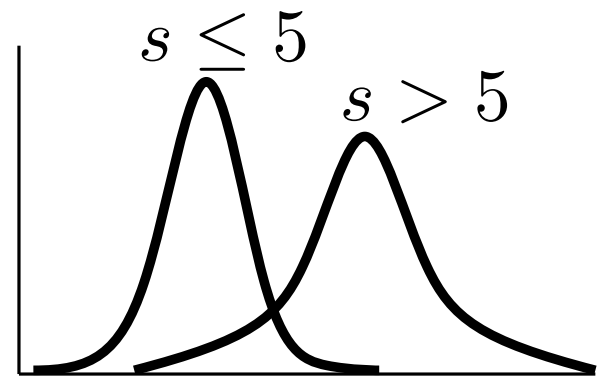
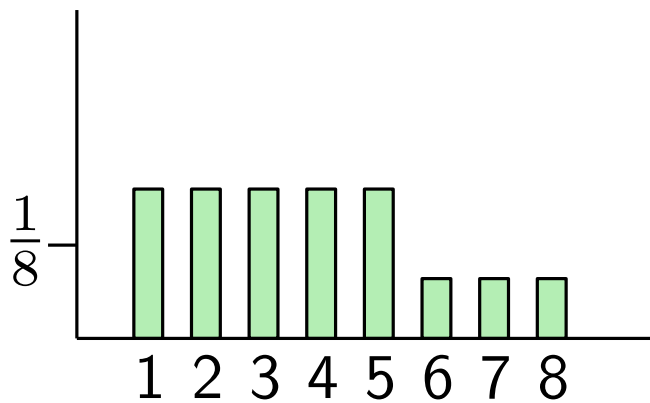
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$



$p(s|o, i^*)$  ←  $p(o|s, i)$  →  $p(o|s, i)$

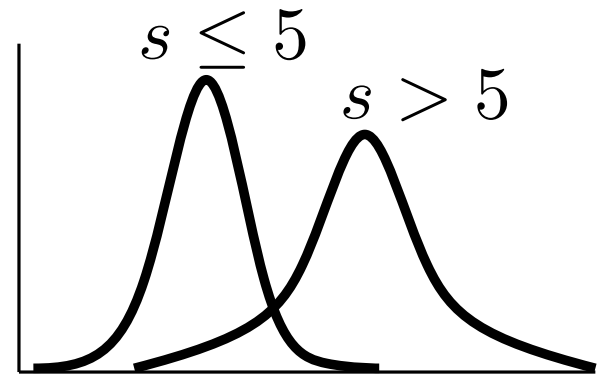
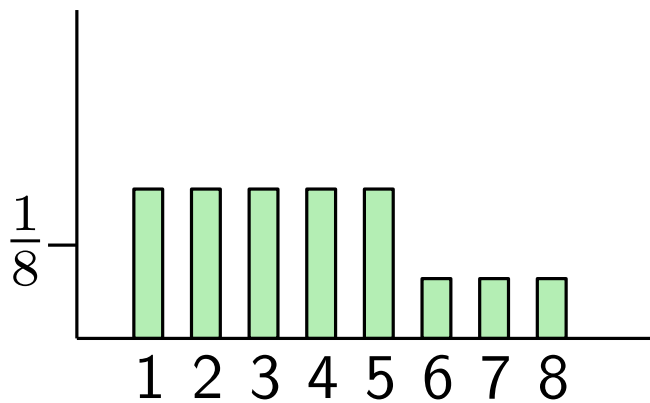
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$



$p(s|o, i^*)$  ←  $p(o|s, i^*)$  →  $p(o|s, i)$

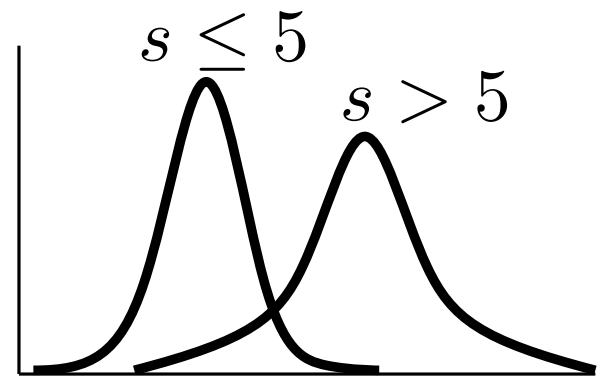
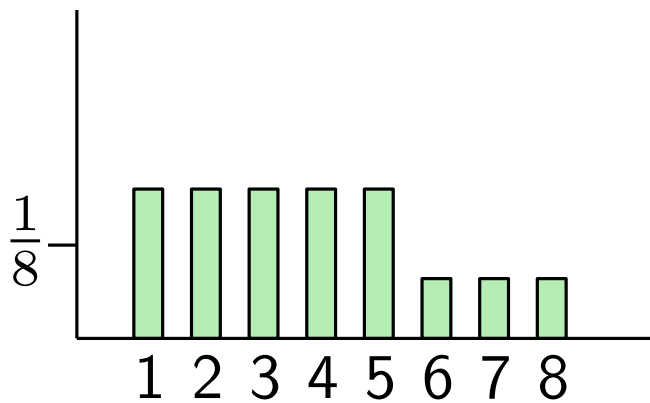
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$



$p(o|s, i)$   $\xleftarrow{p(o|s, i^*)}$   $p(s|o, i^*)$

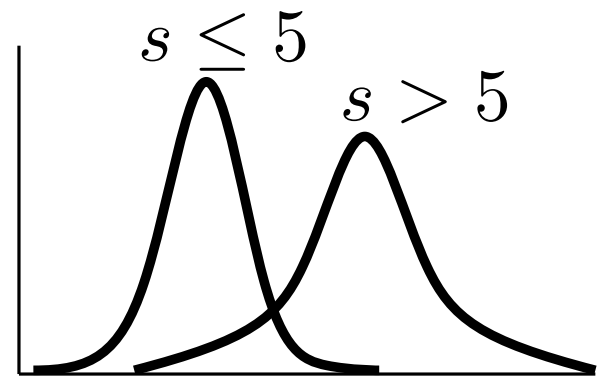
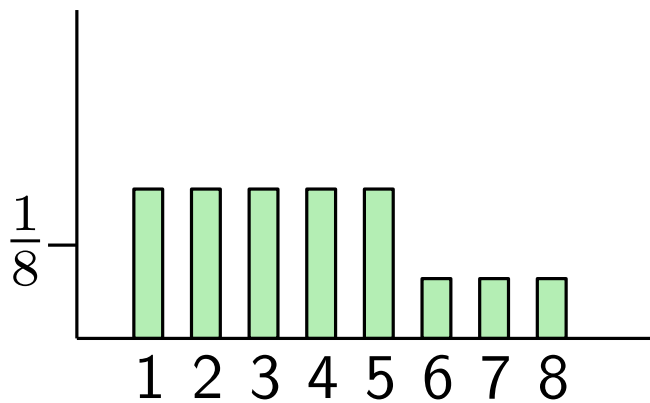
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$



$p(s|o, i^*)$  ←  $p(s|o, i^*)$  →  $p(o|s, i)$

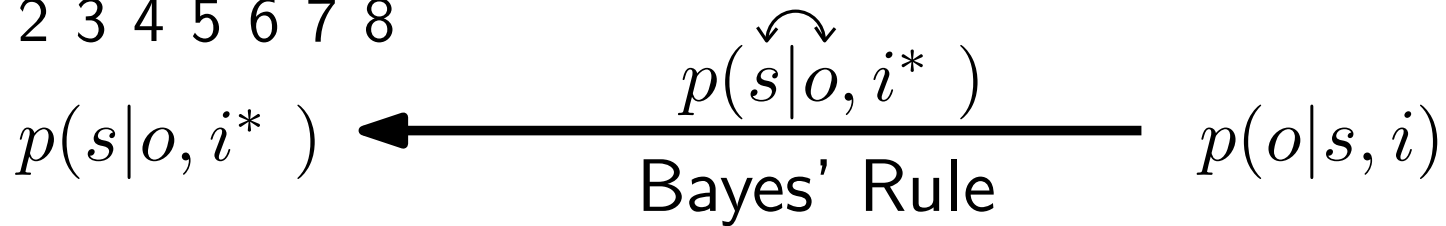
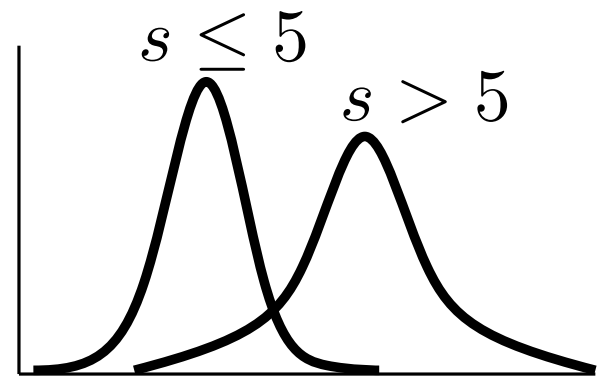
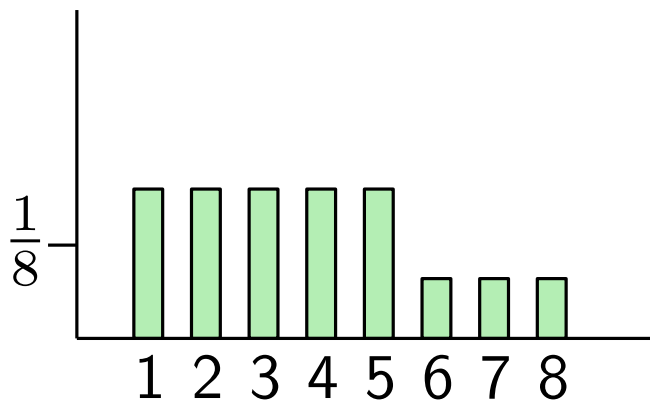
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$





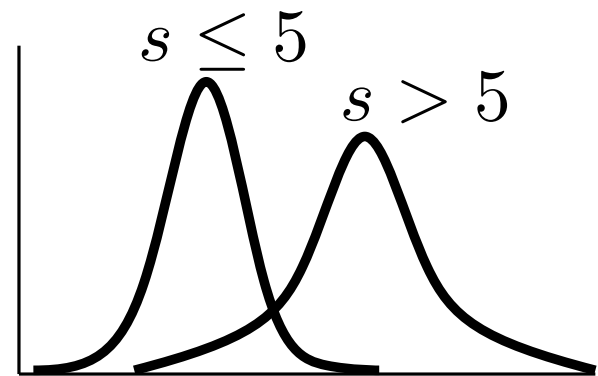
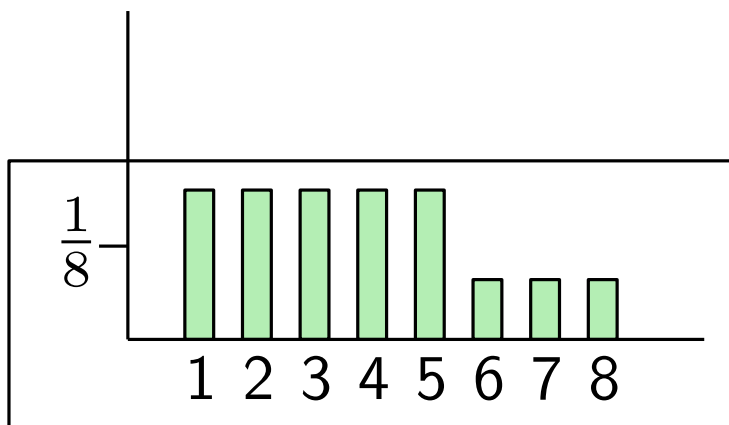
Attacker Belief?

Input Choice?

Observation?



$$i^* = 5$$



$$p(s|o, i^*)$$

$$p(s|o, i^*)$$

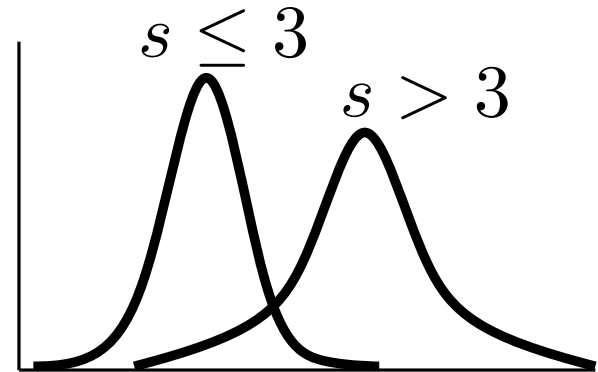
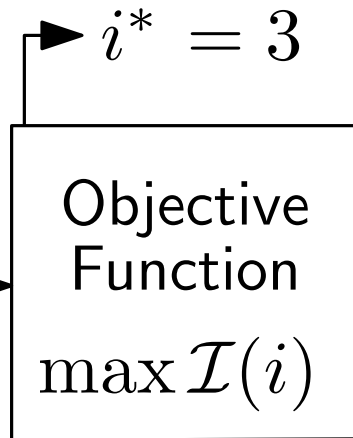
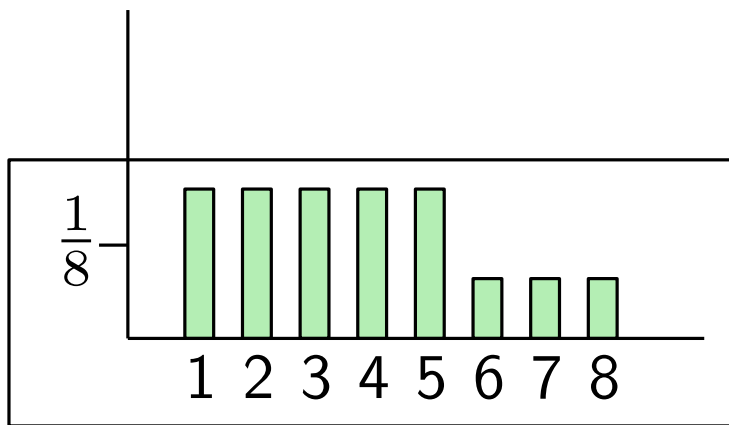
Bayes' Rule

$$p(o|s, i)$$

Attacker Belief?

Input Choice?

Observation?



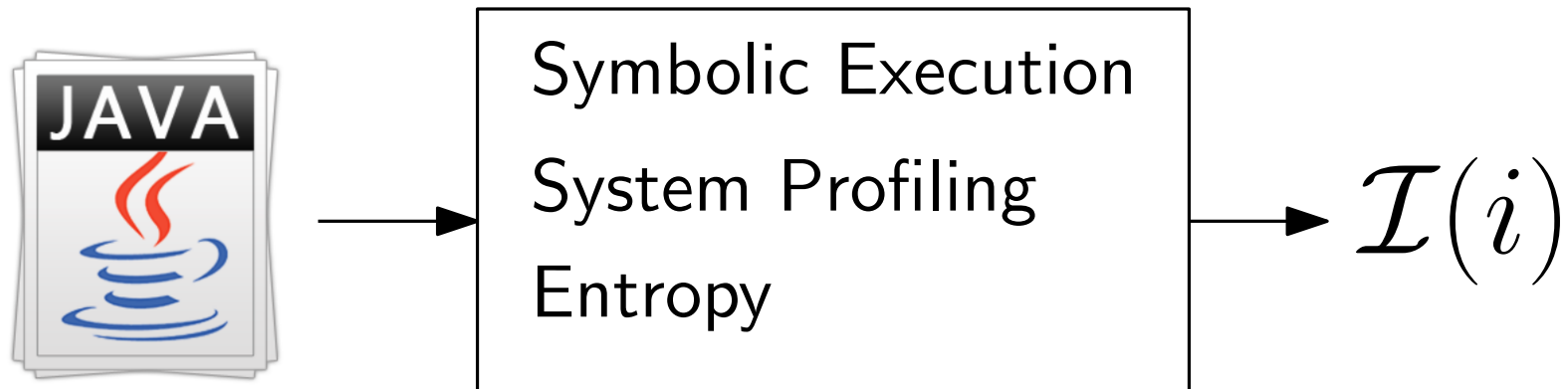
$p(s|o, i^*)$

$p(s|o, i^*)$

Bayes' Rule

$p(o|s, i)$

# Our Approach



$\mathcal{I}(i)$  is a symbolic expression over program inputs  $i$  that measures how much information is gained by an attacker when making input  $i$ .

Find  $i$  that maximizes  $\mathcal{I}(i)$  to get the attacker's best input at every step.



# 1. Offline Static Analysis

1. Offline Static Analysis

2. Offline Dynamic Analysis

1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis

1. Offline Static Analysis

2. Offline Dynamic Analysis

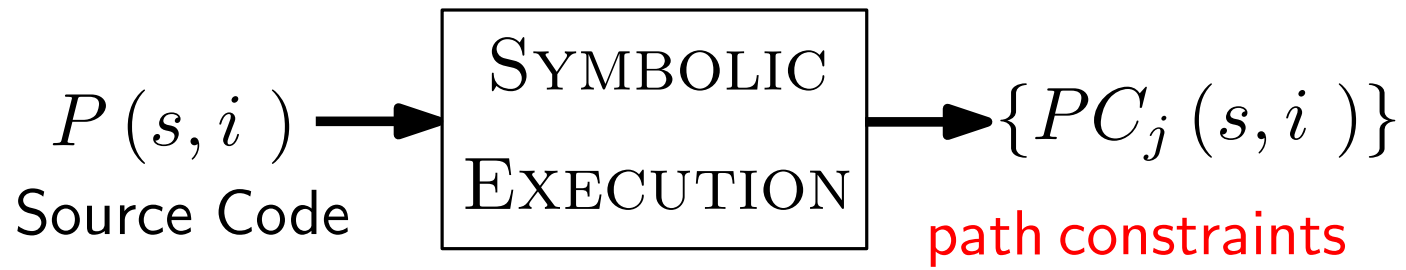
3. Online Attack Synthesis

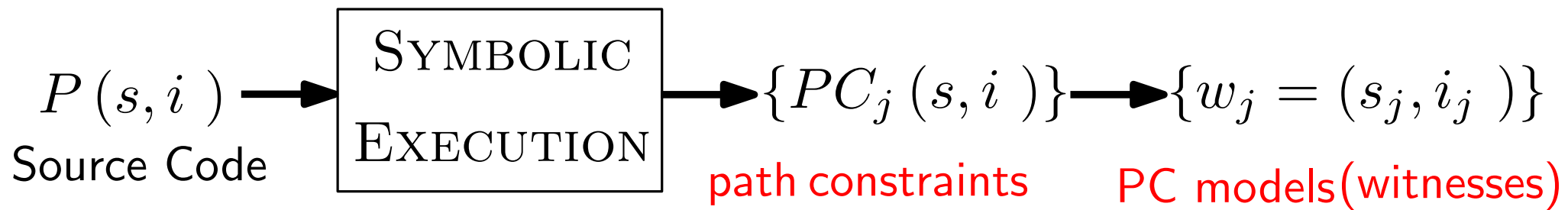


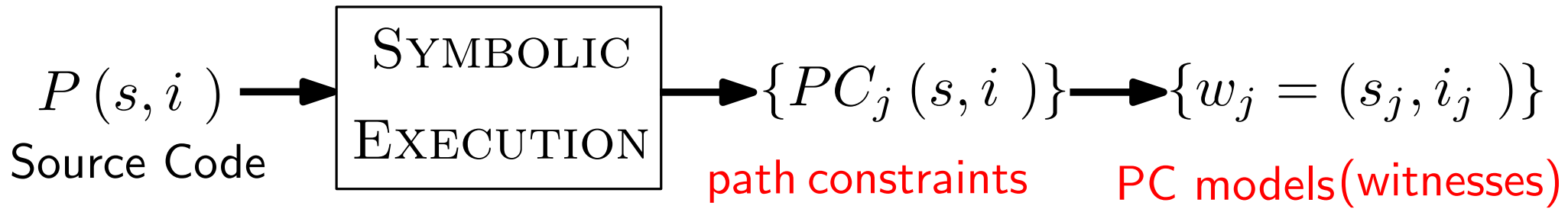


$P(s, i)$

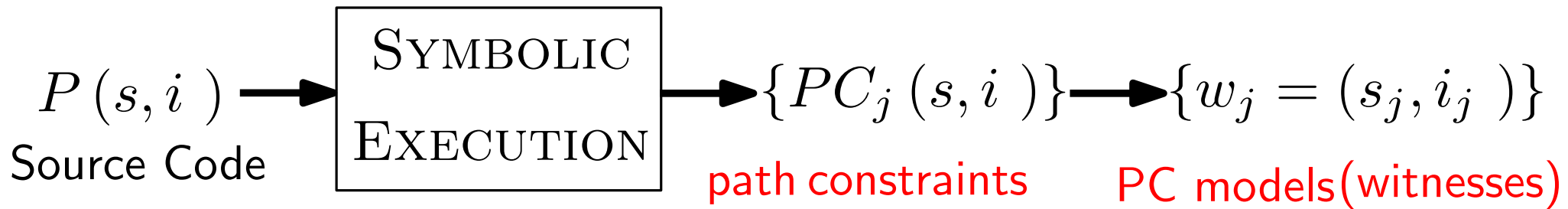
Source Code







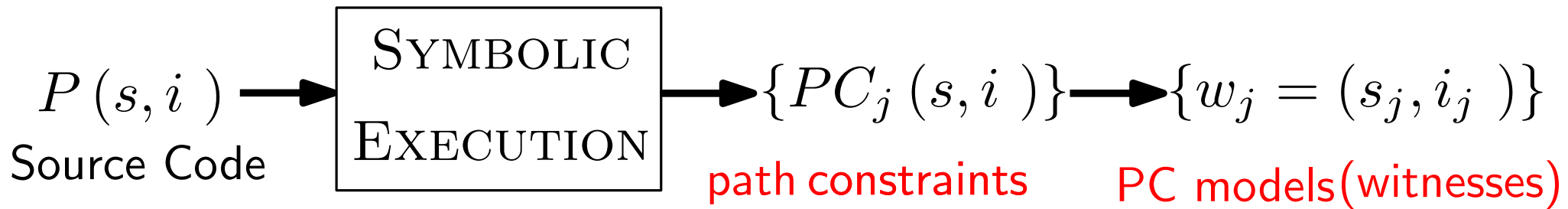
Each PC characterizes an observable program behavior



Each PC characterizes an observable program behavior

$$(s, i) \models PC_j$$

$$(s', i') \models PC_j$$



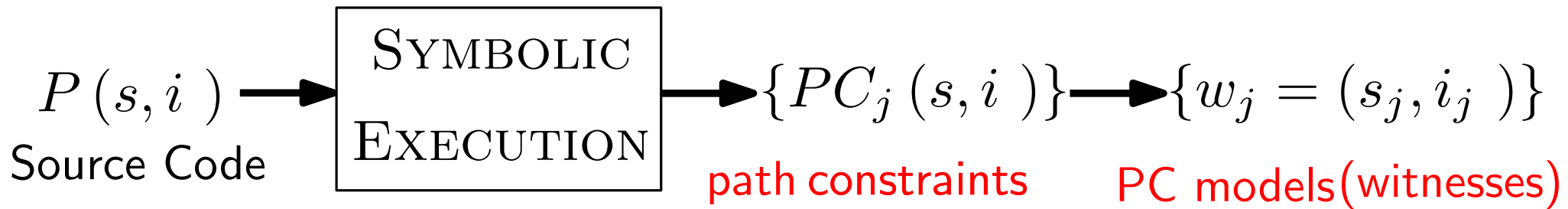
Each PC characterizes an observable program behavior

$$(s, i) \models PC_j$$

$$(s', i') \models PC_j$$

$$P(s, i)$$

$$P(s', i')$$



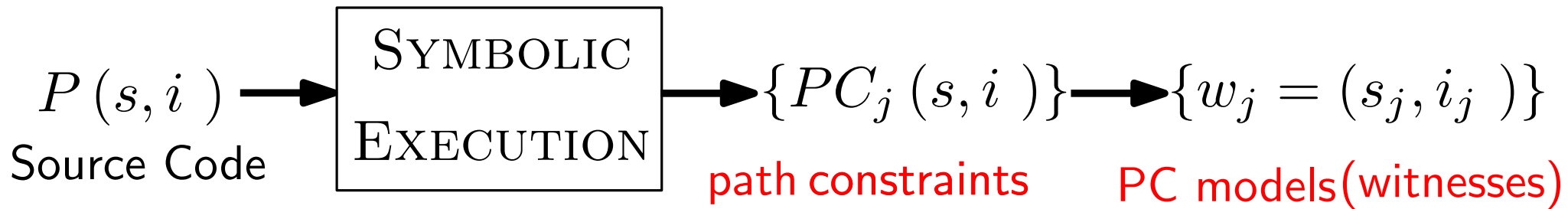
Each PC characterizes an observable program behavior

$$(s, i) \models PC_j$$

$$(s', i') \models PC_j$$

$P(s, i)$     ?        ?     $P(s', i')$





Each PC characterizes an observable program behavior

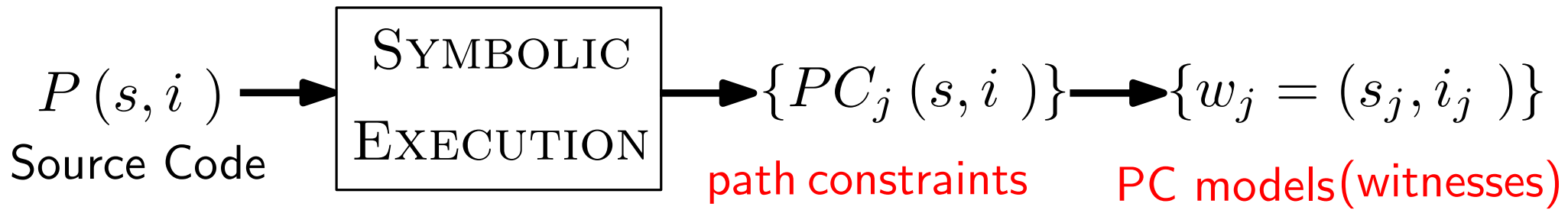
$$(s, i) \models PC_j$$

$$(s', i') \models PC_j$$

$$P(s, i) \quad ? \quad \text{👹} \quad ? \quad P(s', i')$$

$PC_j(s, i)$  characterizes indistinguishable behaviors

$P(s, i)$  is a representative of all behaviors in that class



1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis

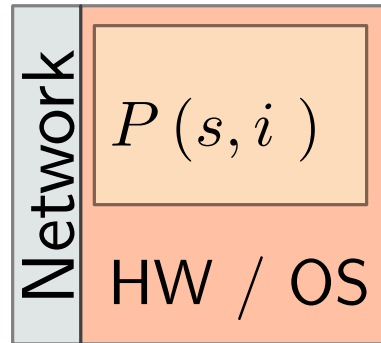
1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis

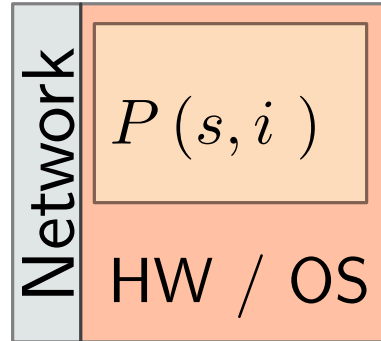
Characterize effect of noise on each class of program behaviors using the witness for that behavior.

Characterize effect of noise on each class of program behaviors using the witness for that behavior.

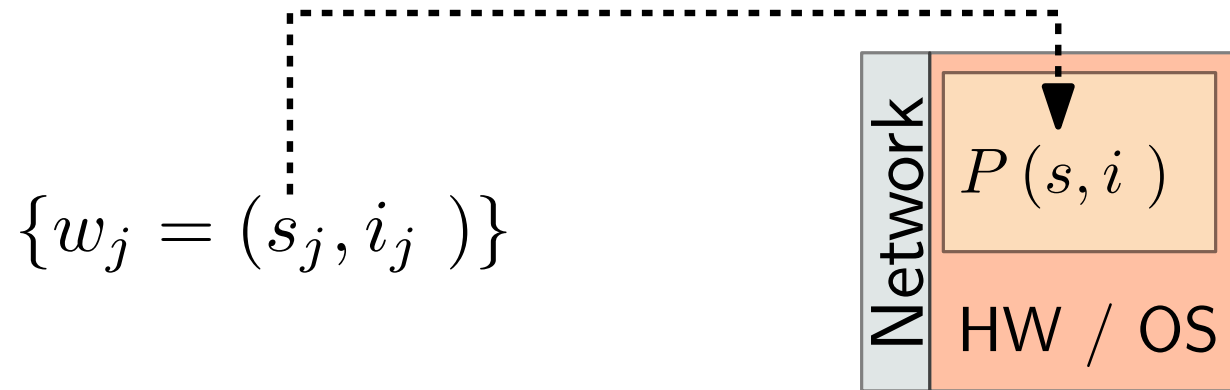


Characterize effect of noise on each class of program behaviors using the witness for that behavior.

$$\{w_j = (s_j, i_j)\}$$

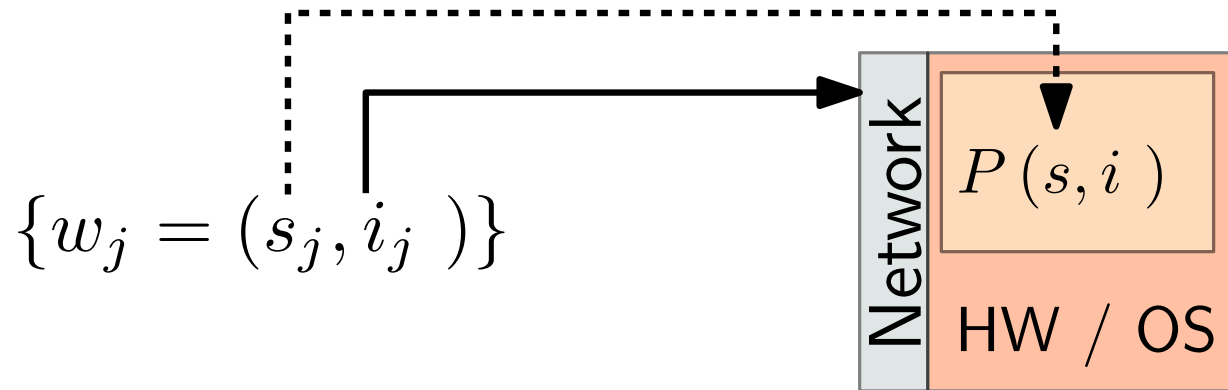


Characterize effect of noise on each class of program behaviors using the witness for that behavior.

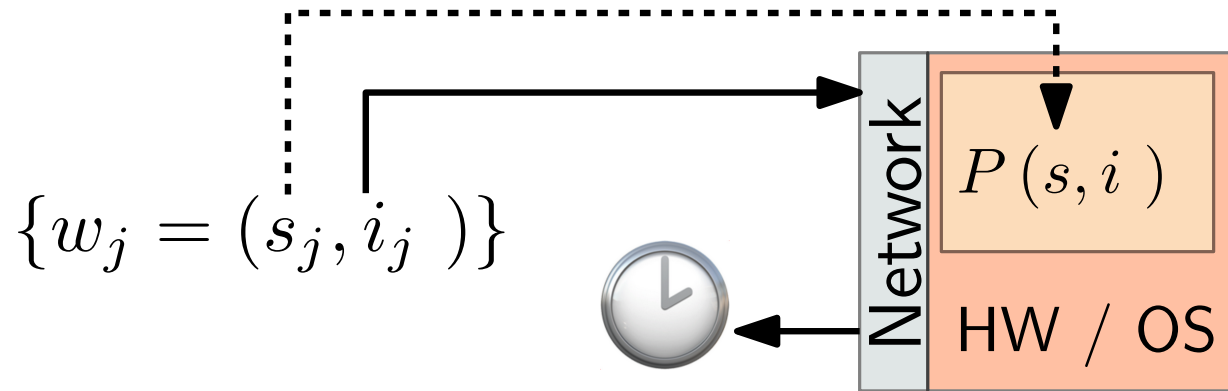




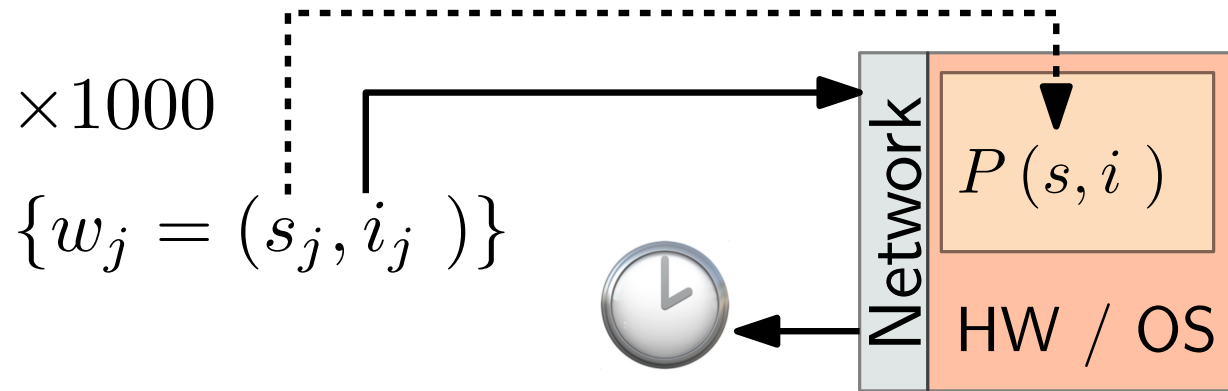
Characterize effect of noise on each class of program behaviors using the witness for that behavior.



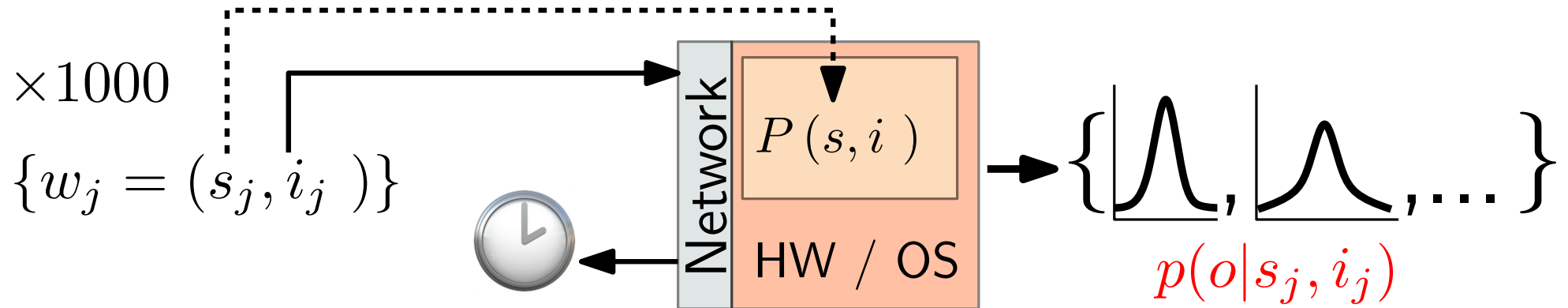
Characterize effect of noise on each class of program behaviors using the witness for that behavior.



Characterize effect of noise on each class of program behaviors using the witness for that behavior.



Characterize effect of noise on each class of program behaviors using the witness for that behavior.



1. Offline Static Analysis

2. Offline Dynamic Analysis

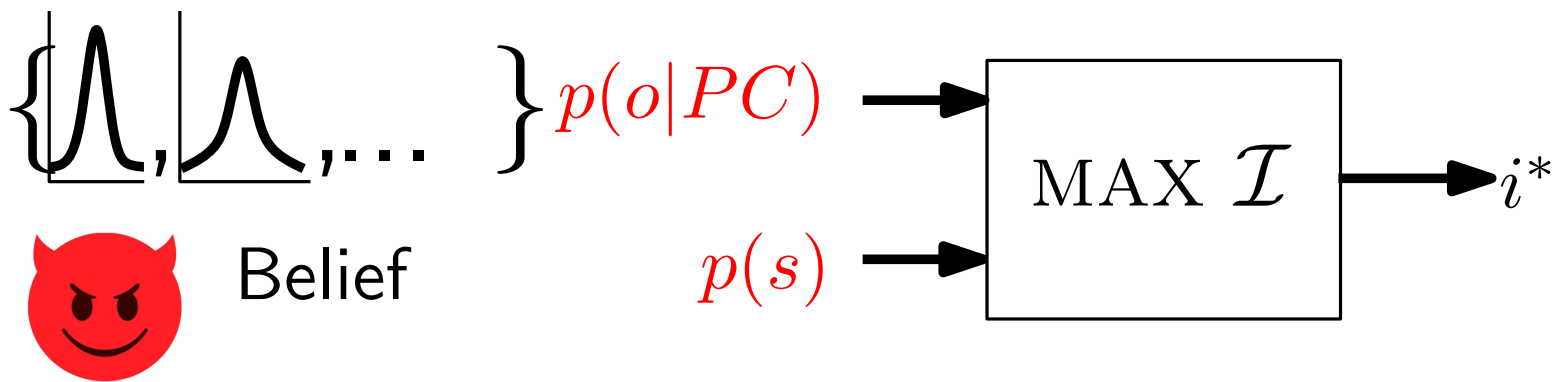
3. Online Attack Synthesis

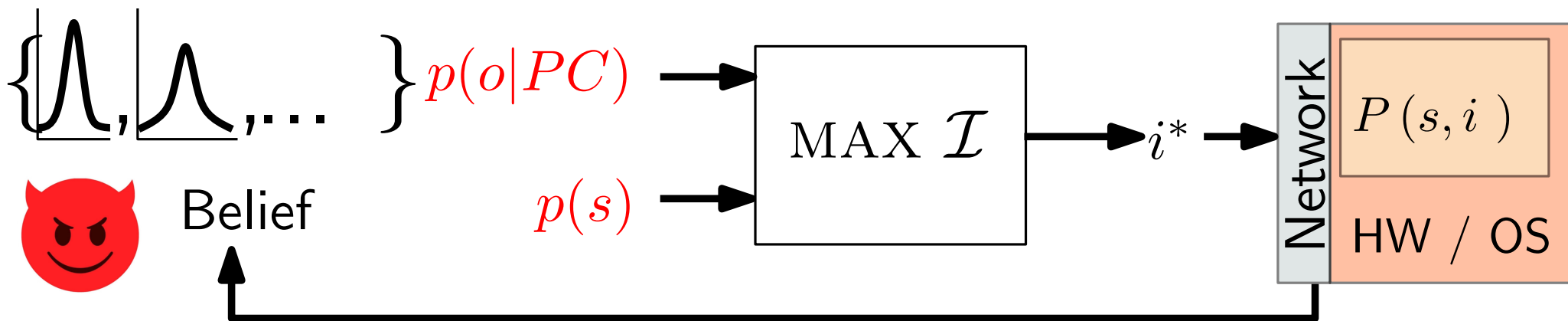
$$\{ \underbrace{\quad}, \underbrace{\quad}, \dots \} p(o|PC)$$



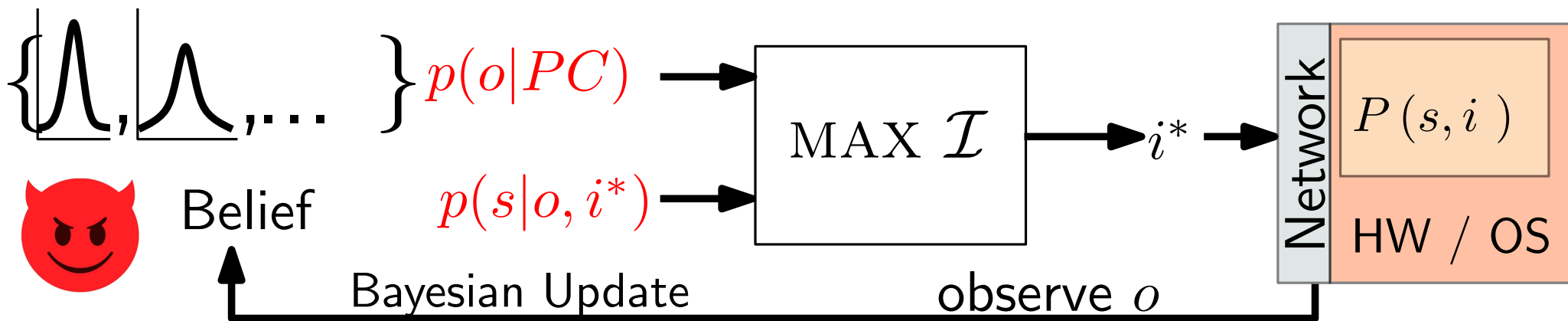
Belief

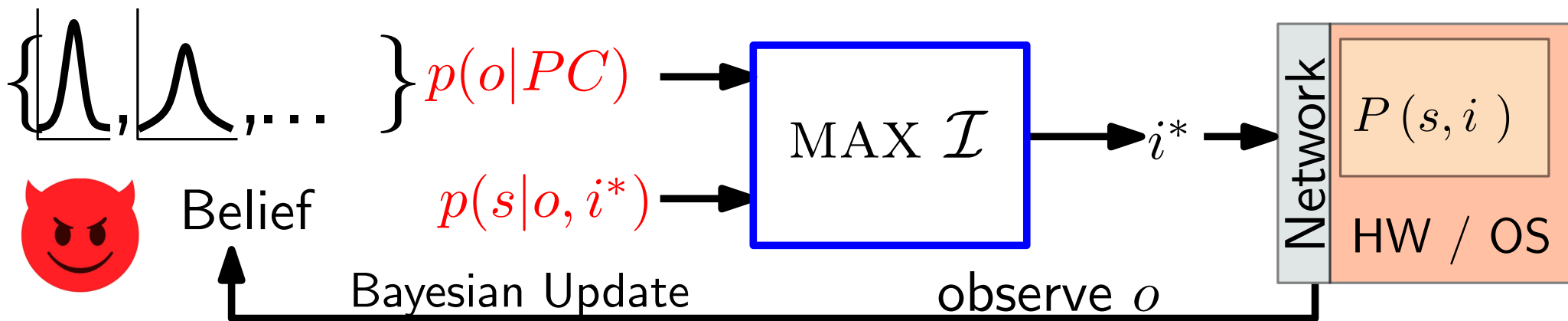
$$p(s)$$

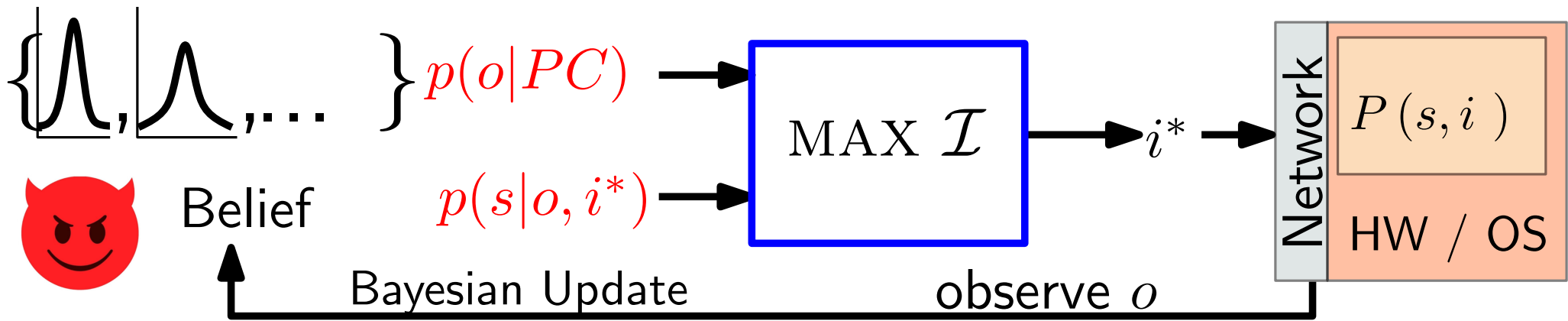








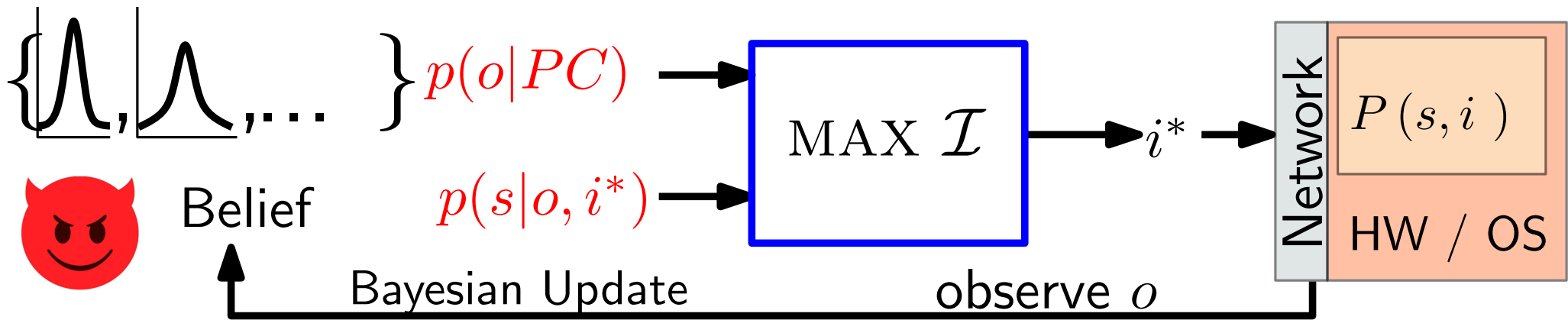




$$\mathcal{I}(s; PC_j | i) = - \sum_{j=1}^n \underline{p(PC_j | i)} \log_2 \underline{p(PC_j | i)}$$

Expected info gain given attacker input

Path constraint probabilities

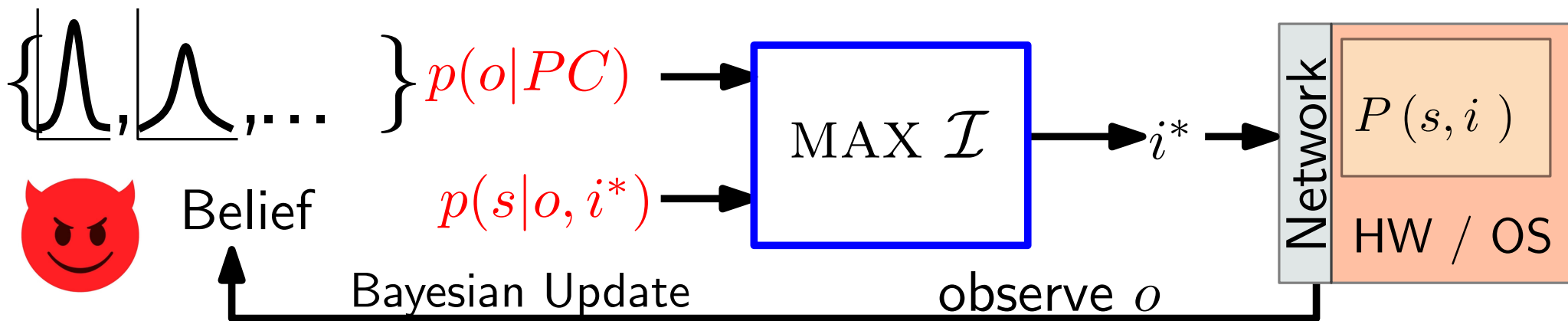


$$\mathcal{I}(s; PC_j | i) = - \sum_{j=1}^n \underbrace{p(PC_j | i)} \log_2 \underbrace{p(PC_j | i)}$$

Expected info gain  
given attacker input

Path constraint probabilities

$$\#(PC_j | i) = \sum_{s \in S} \begin{cases} 1 & \text{if } (s, i) \models PC_j \\ 0 & \text{otherwise} \end{cases}$$



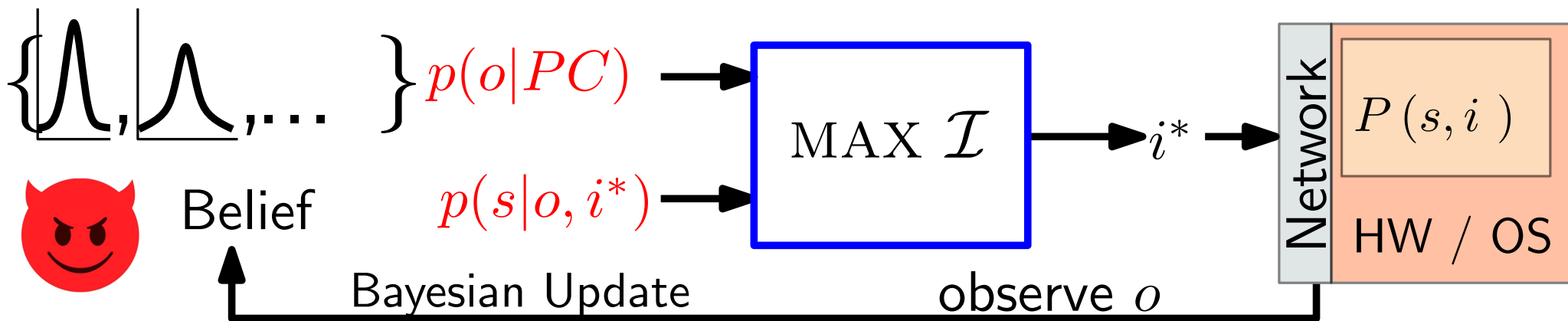
$$\mathcal{I}(s; PC_j | i) = - \sum_{j=1}^n \underbrace{p(PC_j | i)} \log_2 \underbrace{p(PC_j | i)}$$

Expected info gain  
given attacker input

Path constraint probabilities

$$\#(PC_j | i) = \sum_{s \in S} \begin{cases} 1 & \text{if } (s, i) \models PC_j \\ 0 & \text{otherwise} \end{cases}$$

Model Counting



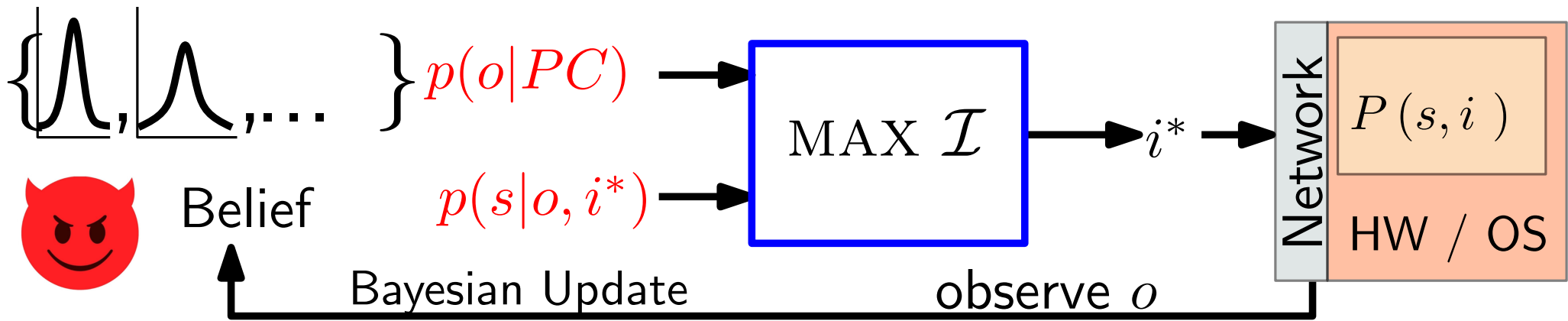
$$\mathcal{I}(s; PC_j | i) = - \sum_{j=1}^n \underbrace{p(PC_j | i)} \log_2 \underbrace{p(PC_j | i)}$$

Expected info gain  
given attacker input

Path constraint probabilities

$$p(PC_j | i) = \sum_{s \in S} p(s) \times \begin{cases} 1 & \text{if } (s, i) \models PC_j \\ 0 & \text{otherwise} \end{cases}$$

Weighted Model Counting



$$\mathcal{I}(s; PC_j | i) = - \sum_{j=1}^n \underbrace{p(PC_j | i)} \log_2 \underbrace{p(PC_j | i)}$$

Expected info gain  
given attacker input

Path constraint probabilities

$$p(PC_j | i) = \sum_{s \in S} p(s) \times \begin{cases} 1 & \text{if } (s, i) \models PC_j \\ 0 & \text{otherwise} \end{cases}$$

Weighted Model Counting

BARVINOK

1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis



# Implementation

**NASA Symbolic  
PathFinder (SPF)**

**Z3 Constraint Solver**

**Python**

**Profiler Client**

**Intel**

**NUC Server**

$P(s, i)$

**Barvinok**

**Weighted Symbolic**

**Model Counting**

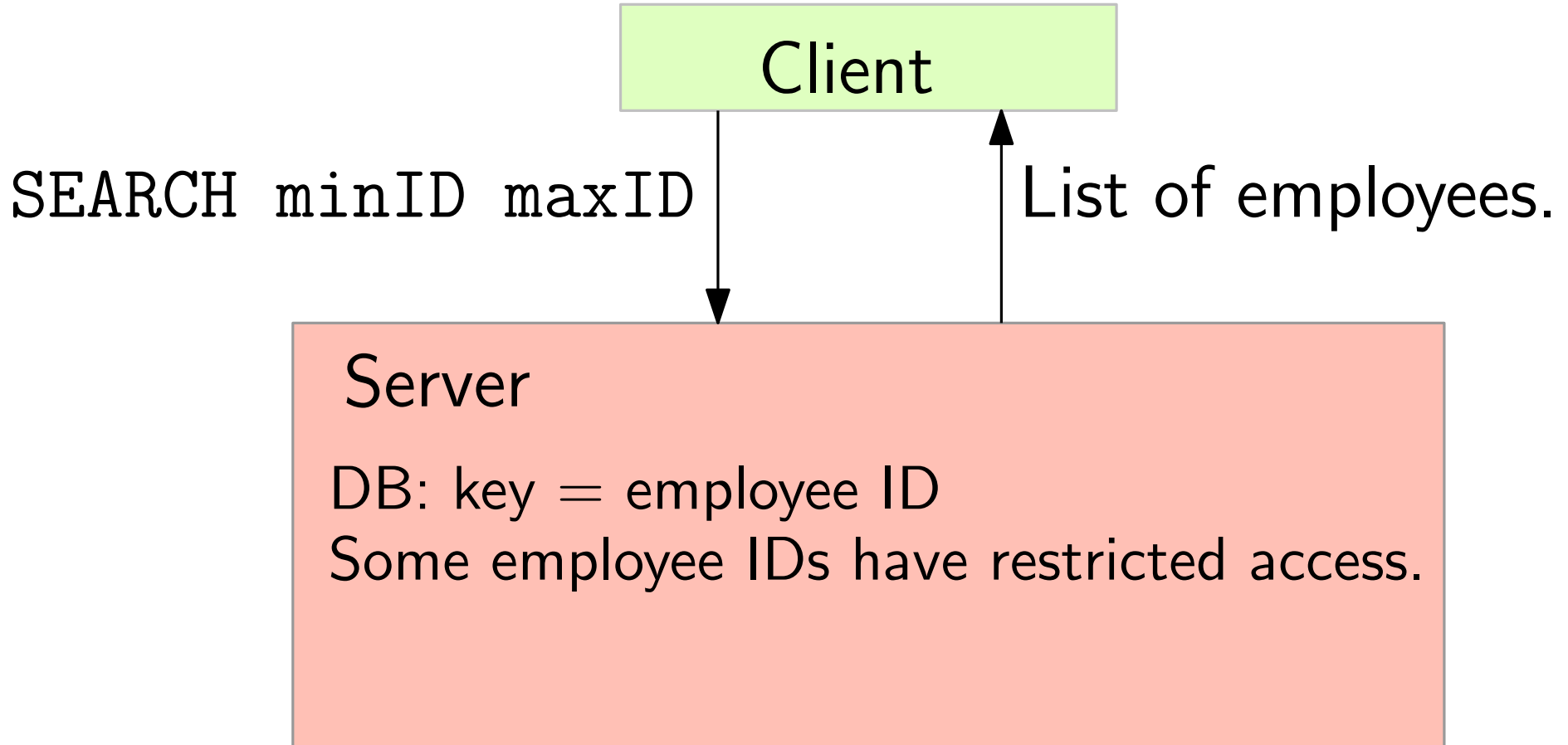
**Mathematica**

**Symbolic Entropy Computation**

**Numeric Maximization**

# Case Study: LawDB

From Defense Advanced Research Projects Agency (DARPA)  
Space-Time Analysis for Cybersecurity (STAC) Project



Writes to log file depending on

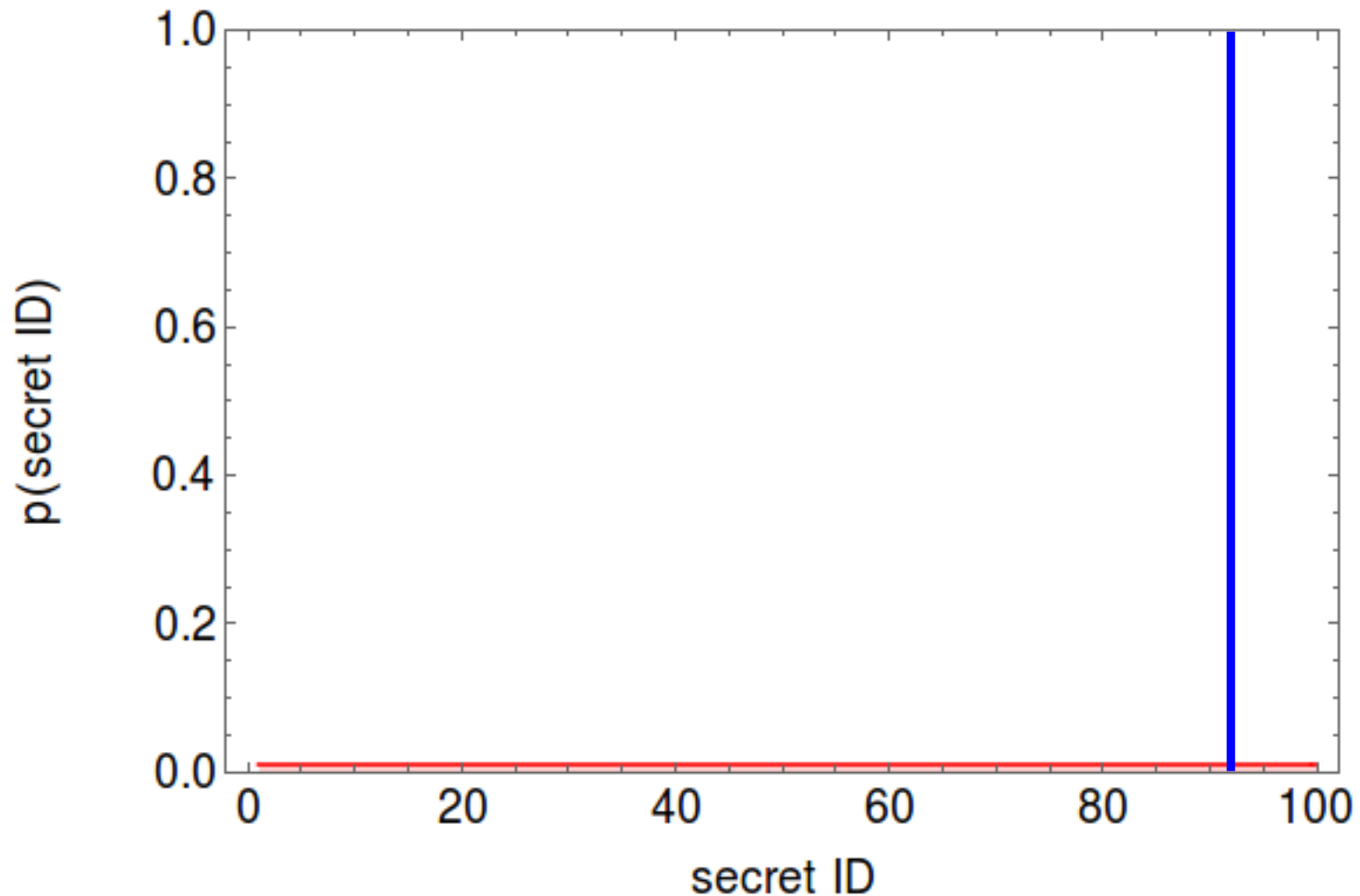
$$ID_{res} \in [minID, maxID]$$

$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 0: SEARCH --

Observed time: --

Entropy = 6.64386

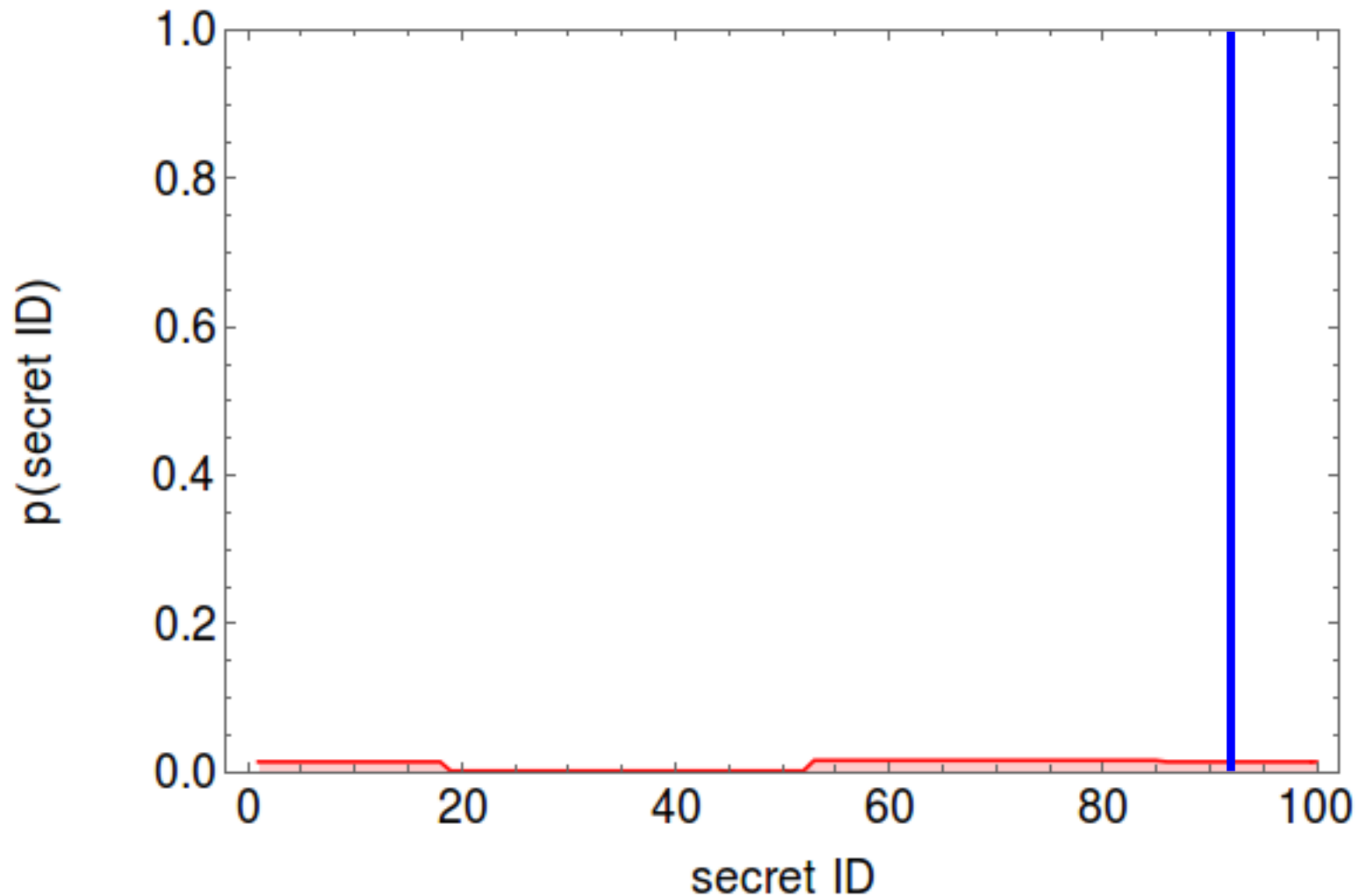


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 1: SEARCH 19 52

Observed time:0.00444

Entropy = 6.27408

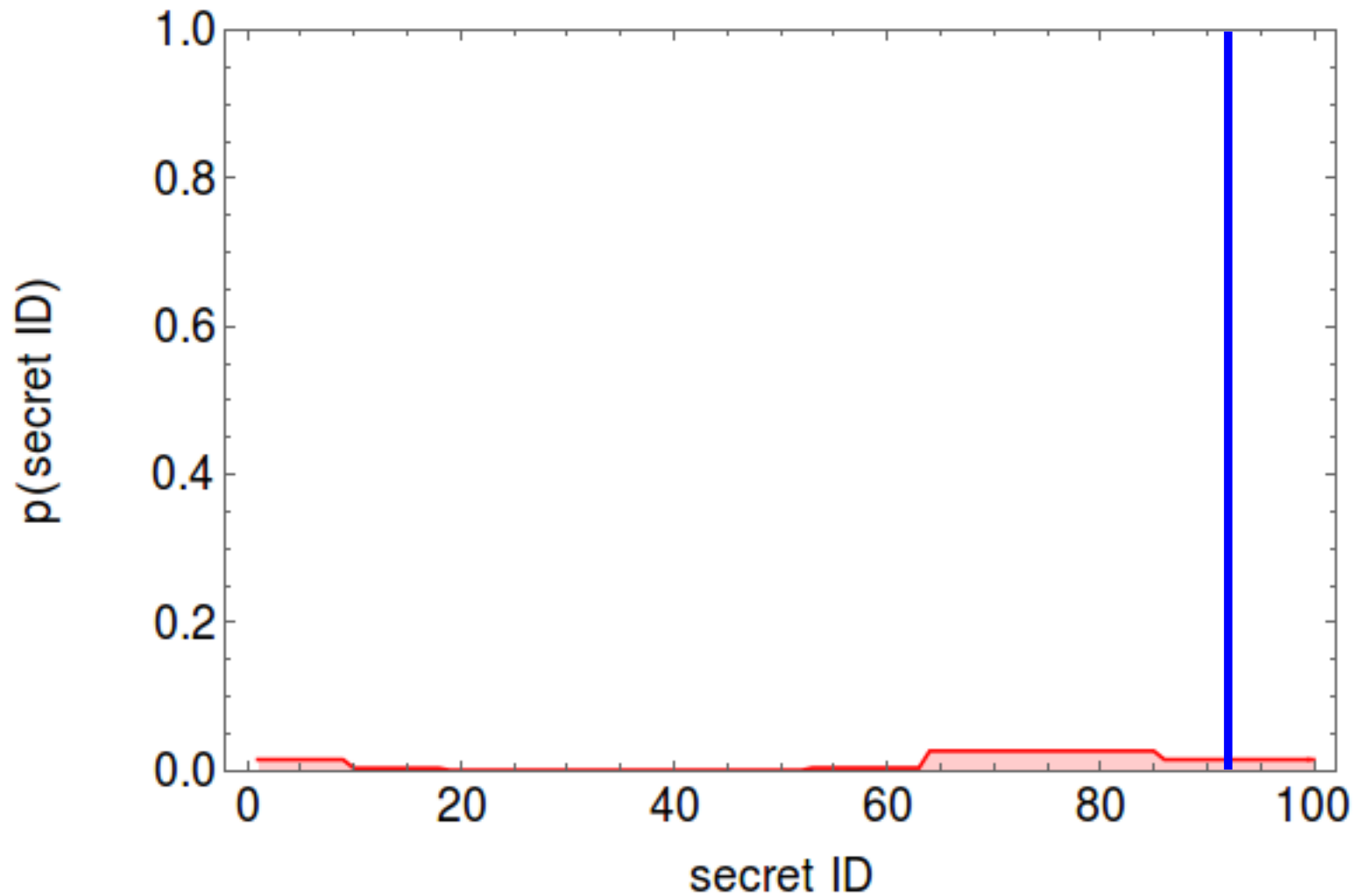


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 2: SEARCH 10 63

Observed time:0.00436

Entropy = 5.81014

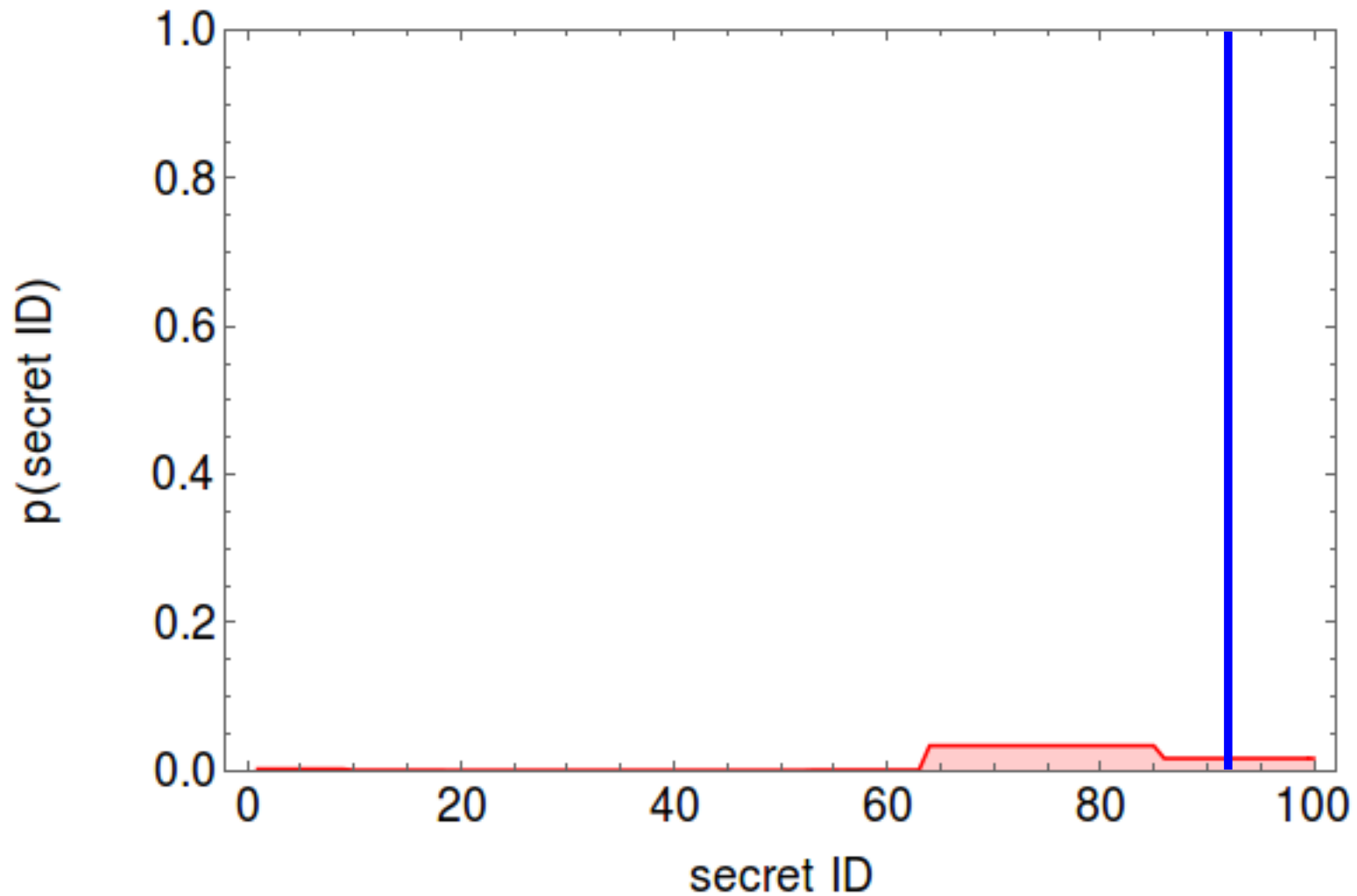


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 3: SEARCH 1 63

Observed time:0.0043

Entropy = 5.28658

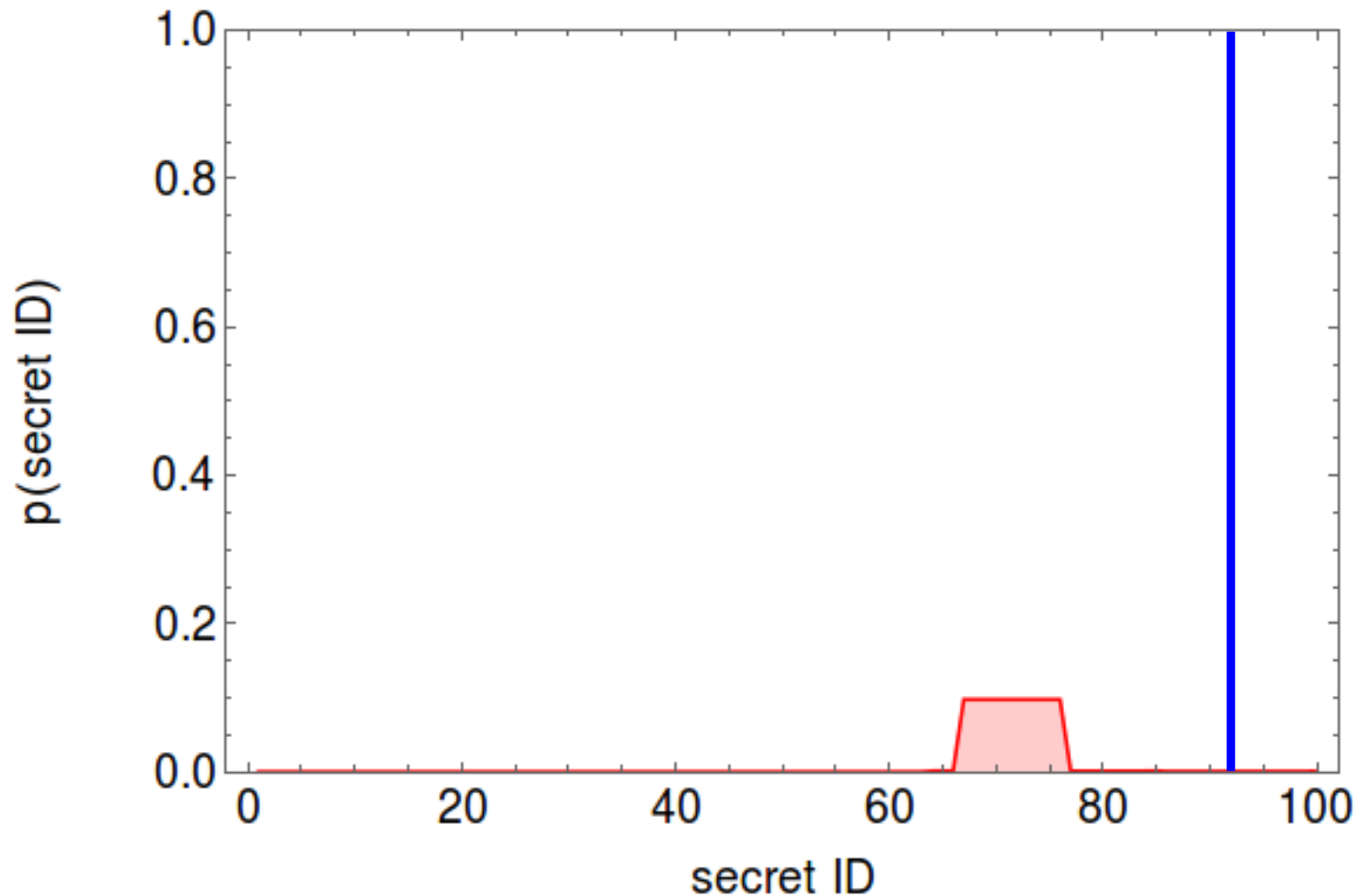


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 4: SEARCH 63 85

Observed time:0.00733

Entropy = 3.53218

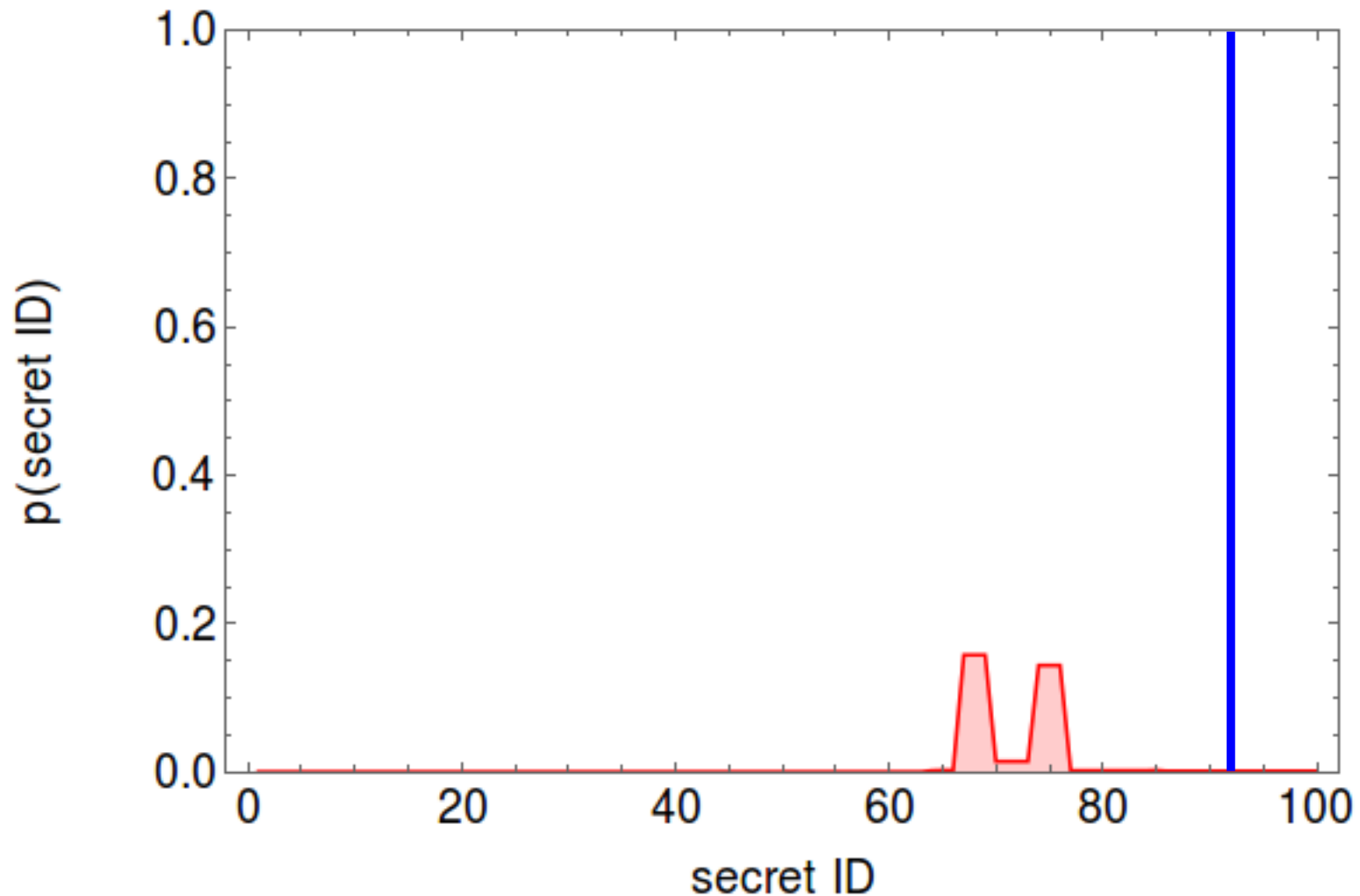


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 5: SEARCH 70 73

Observed time:0.00447

Entropy = 3.19249



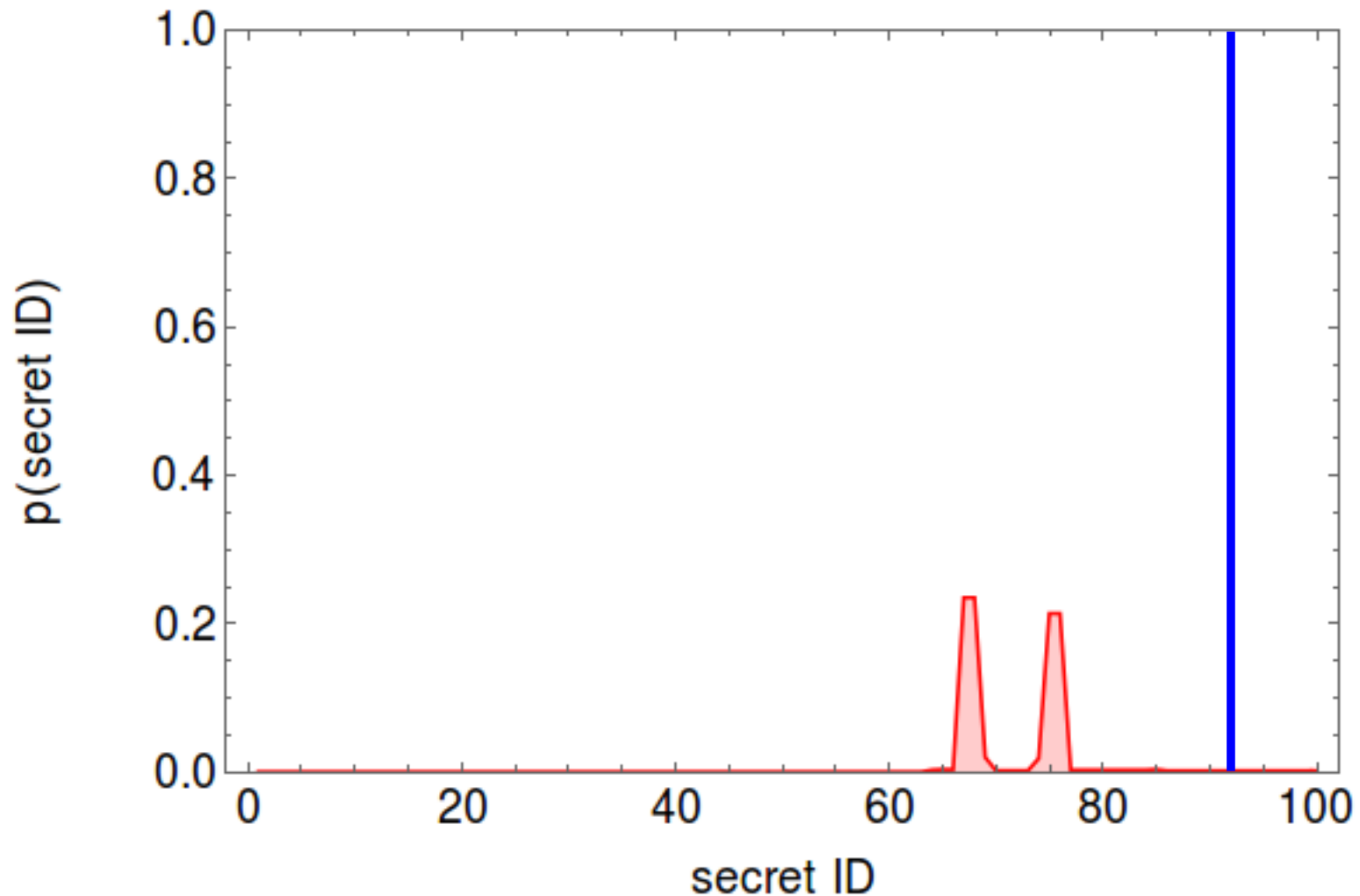


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 6: SEARCH 67 74

Observed time:0.00427

Entropy = 2.74012

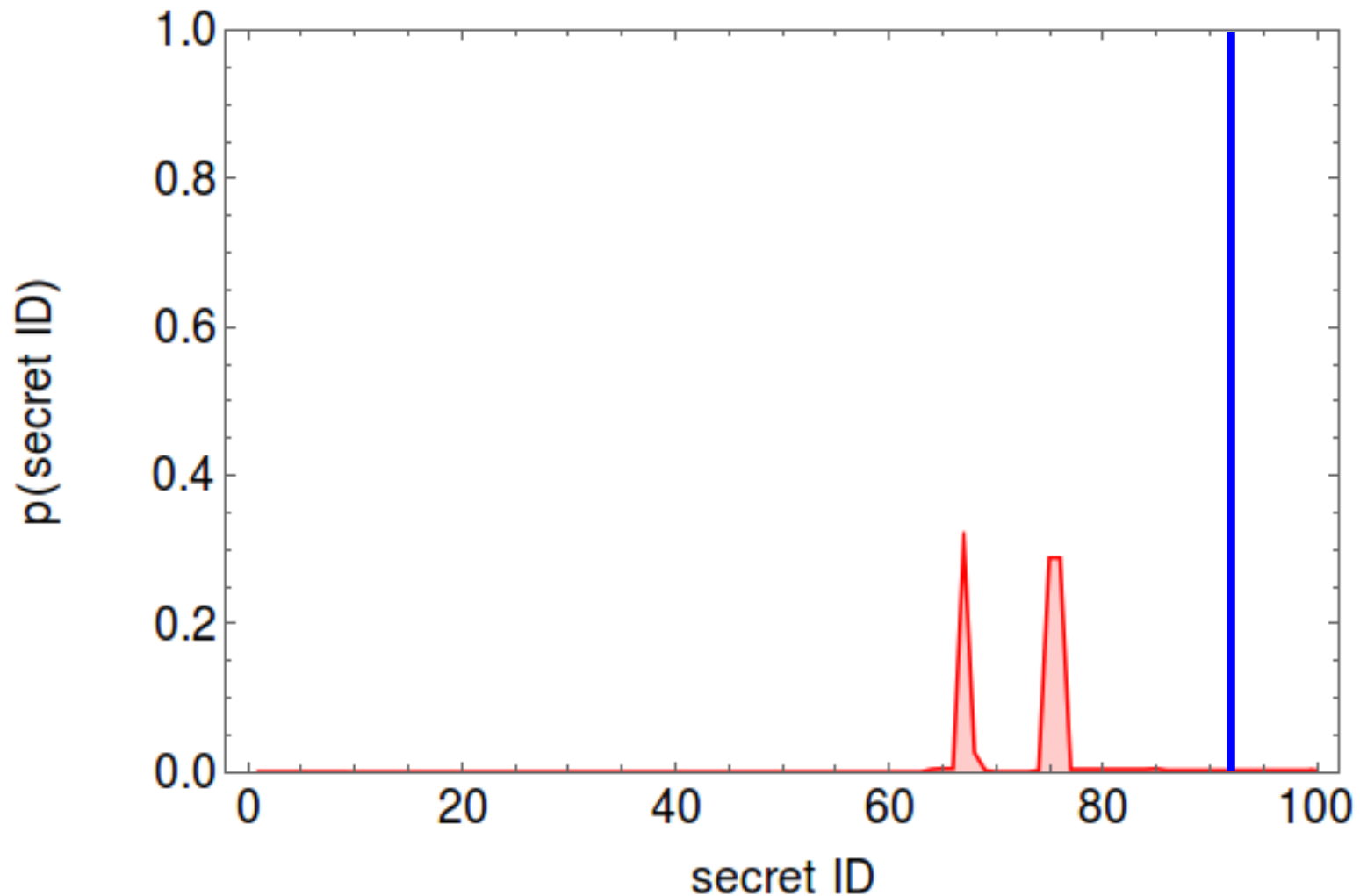


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 7: SEARCH 63 74

Observed time:0.00452

Entropy = 2.41548

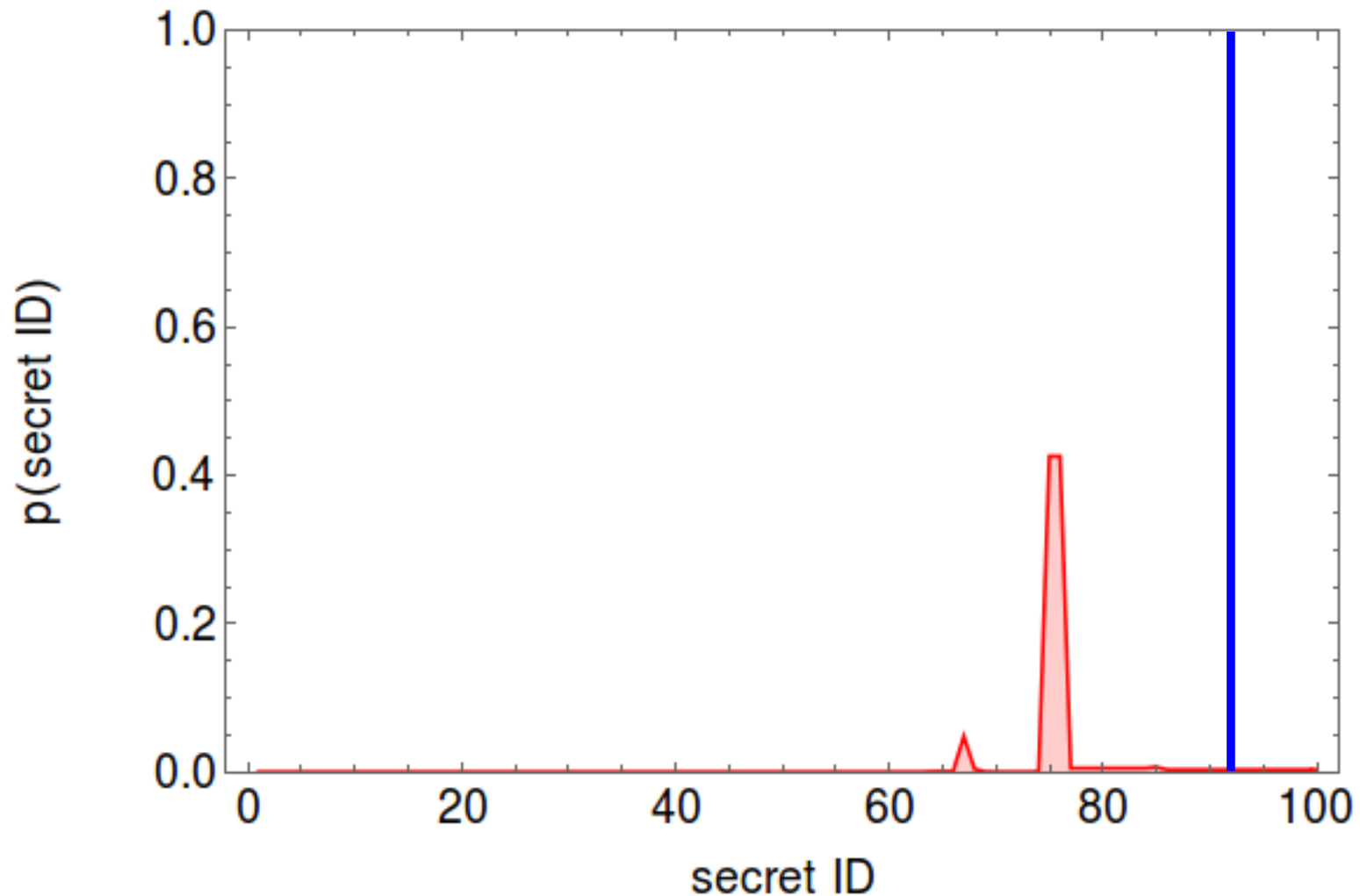


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 8: SEARCH 63 70

Observed time:0.00435

Entropy = 2.07286

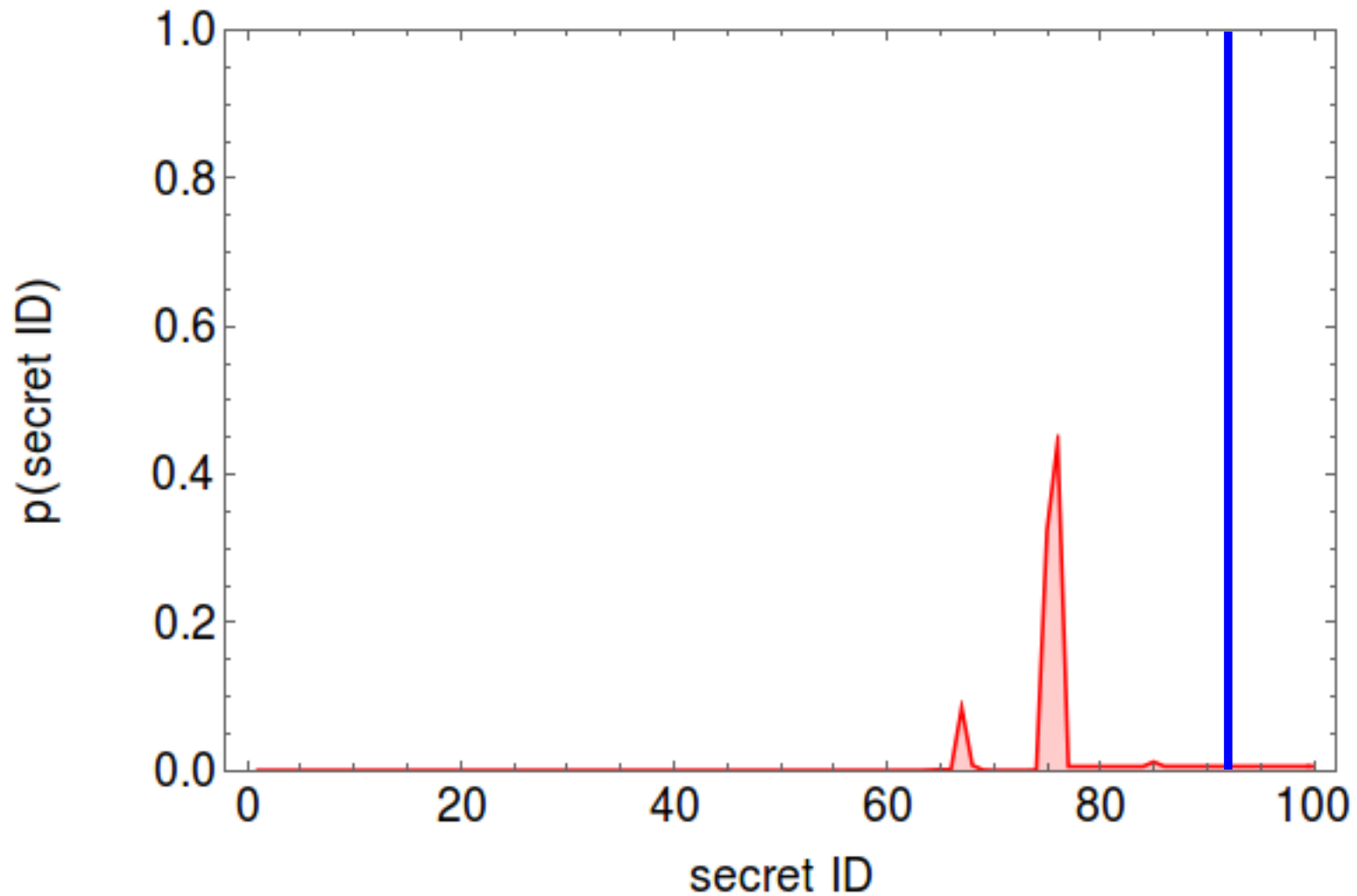


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 9: SEARCH 74 75

Observed time:0.00431

Entropy = 2.46103

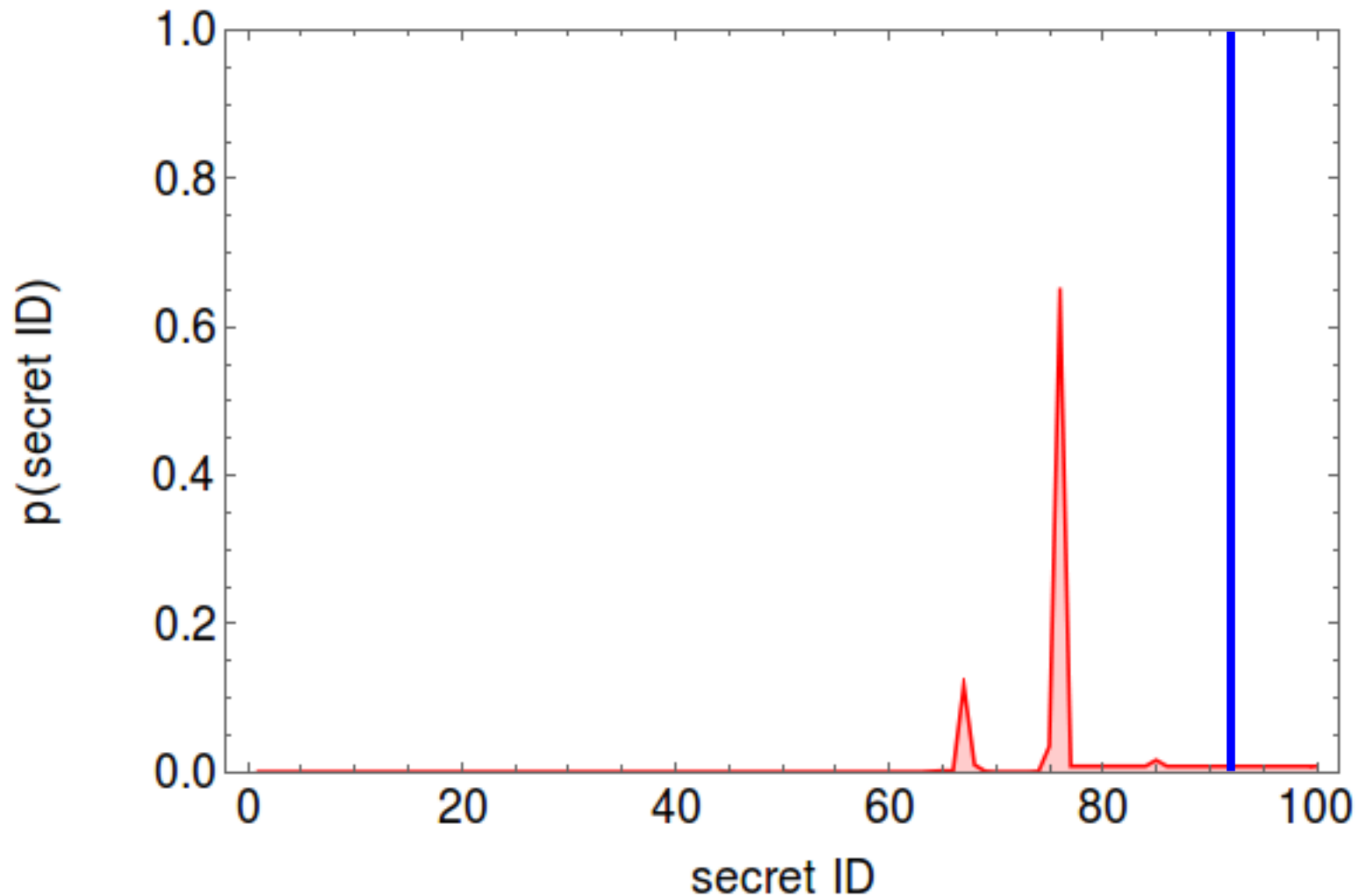


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 10: SEARCH 74 75

Observed time:0.00435

Entropy = 2.39414

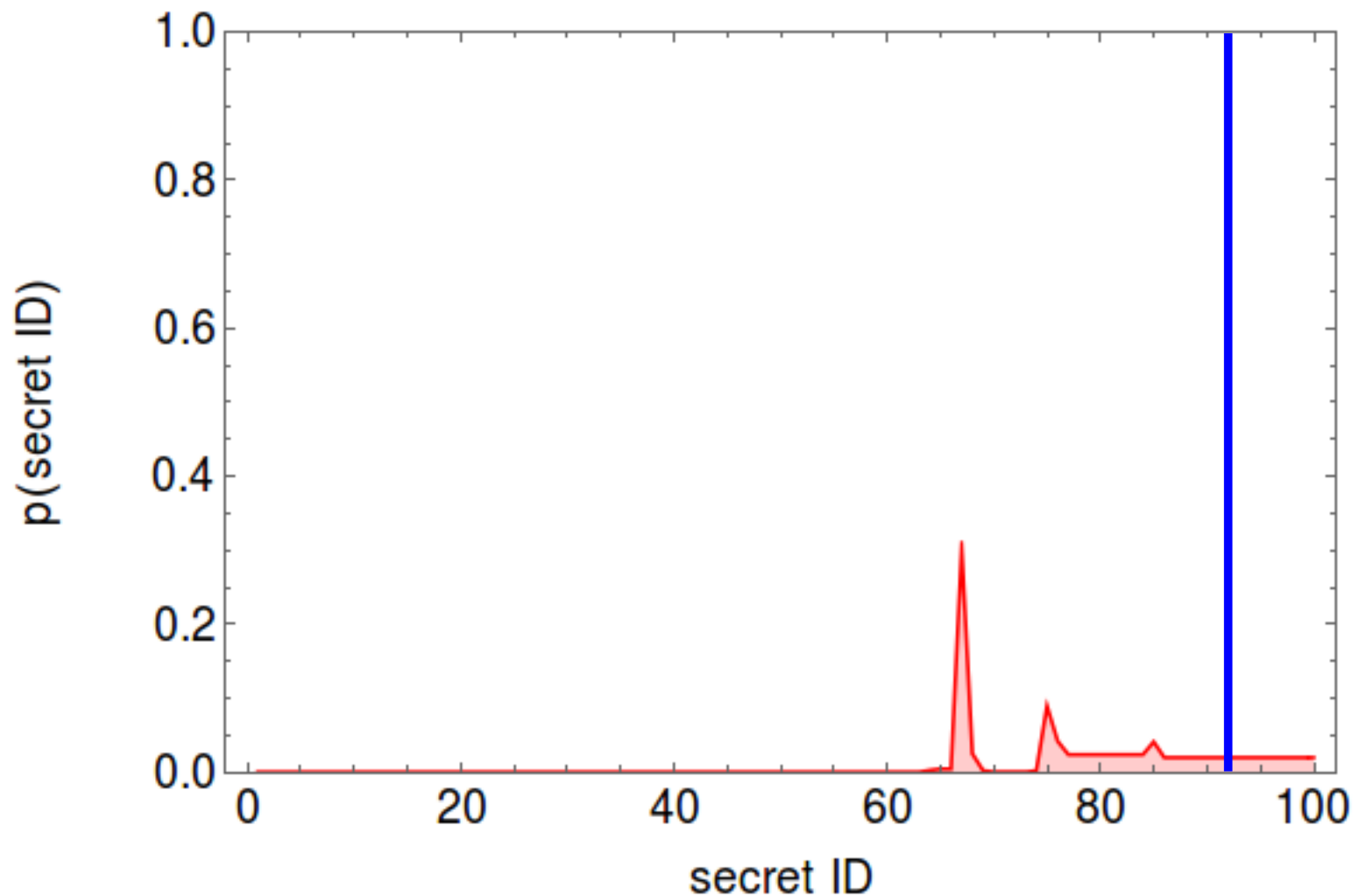


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 11: SEARCH 63 100

Observed time:0.00732

Entropy = 4.19456

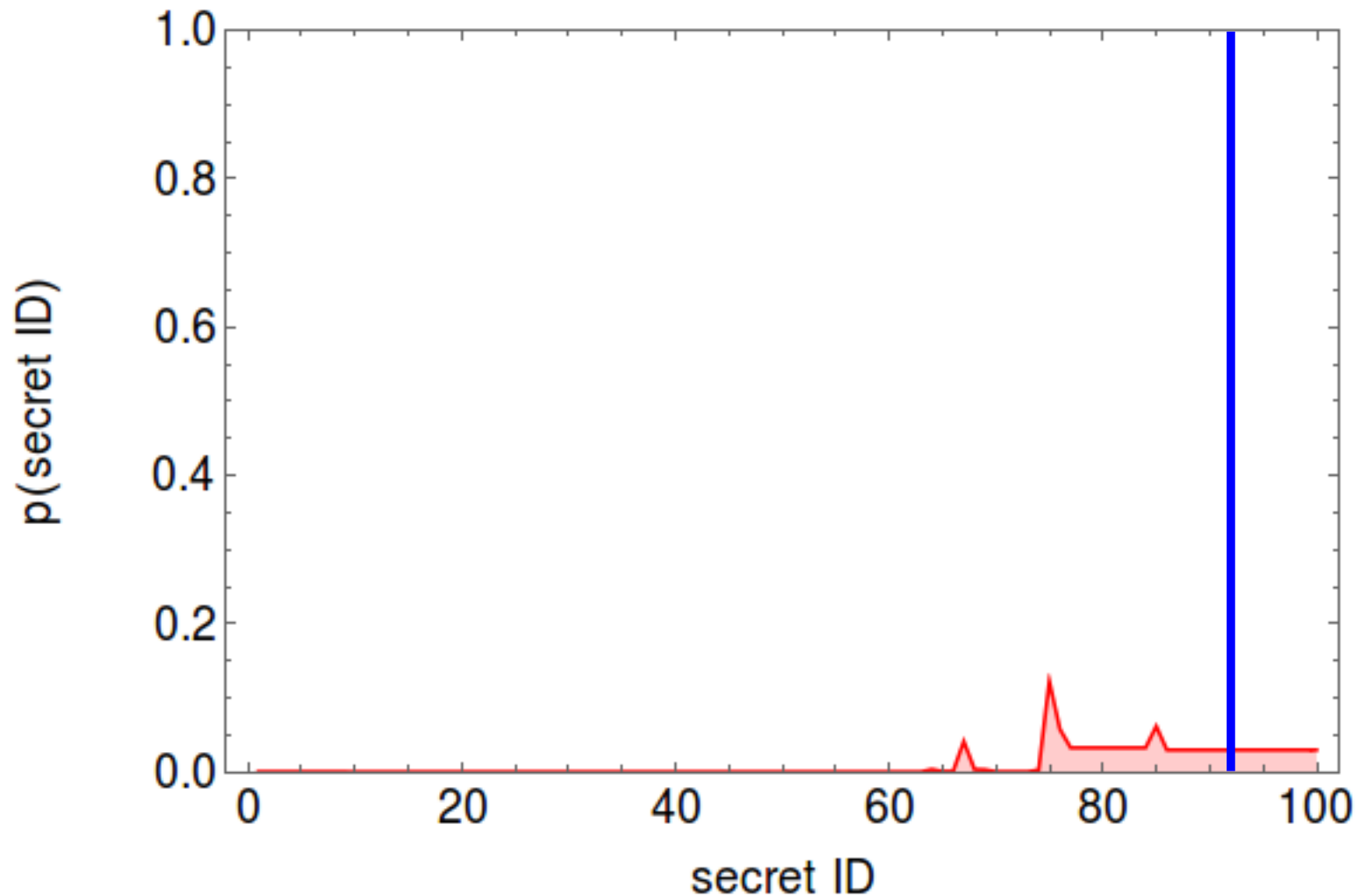


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 12: SEARCH 74 100

Observed time:0.00743

Entropy = 4.73142

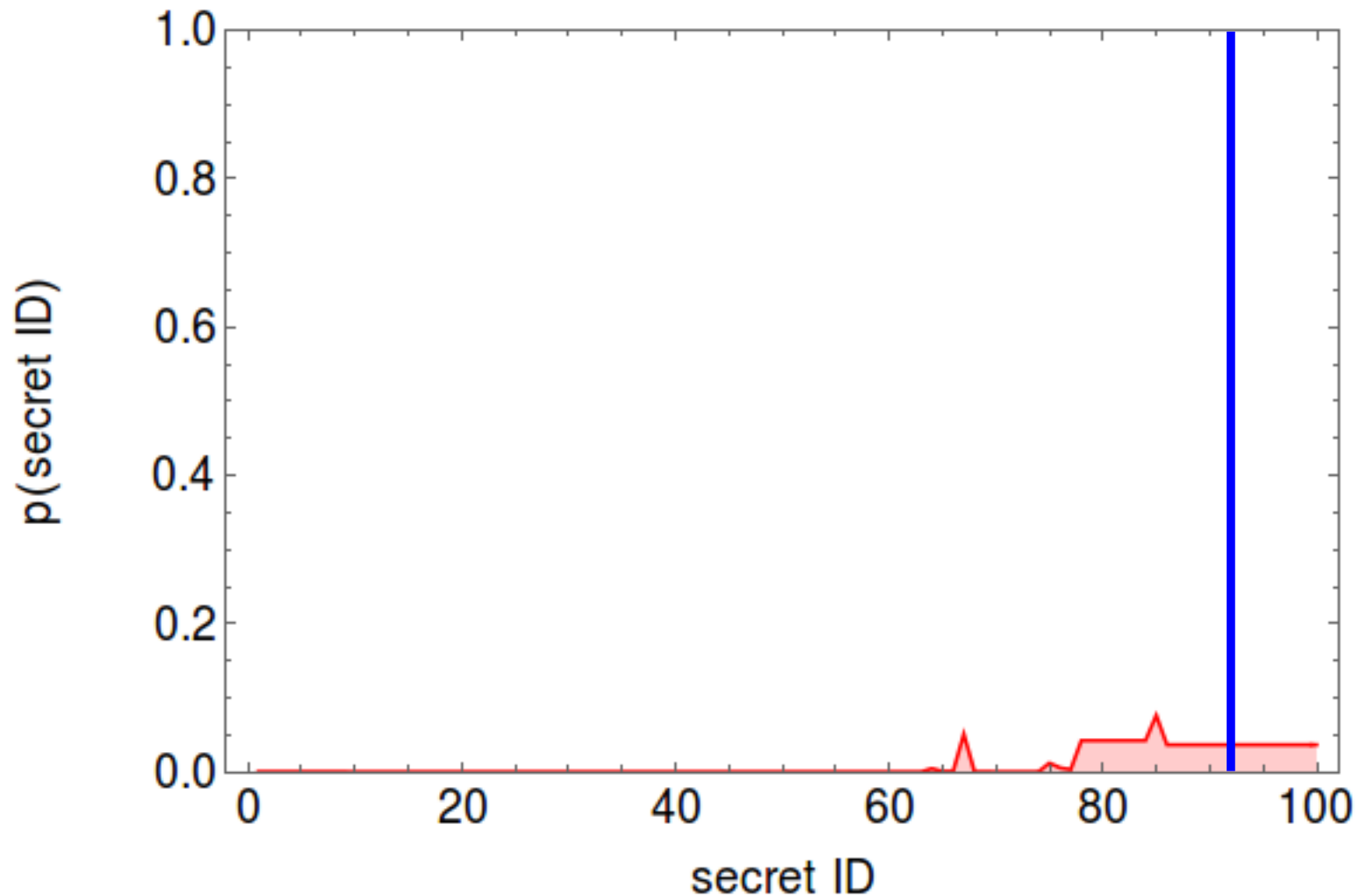


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 13: SEARCH 78 100

Observed time:0.00733

Entropy = 4.70767



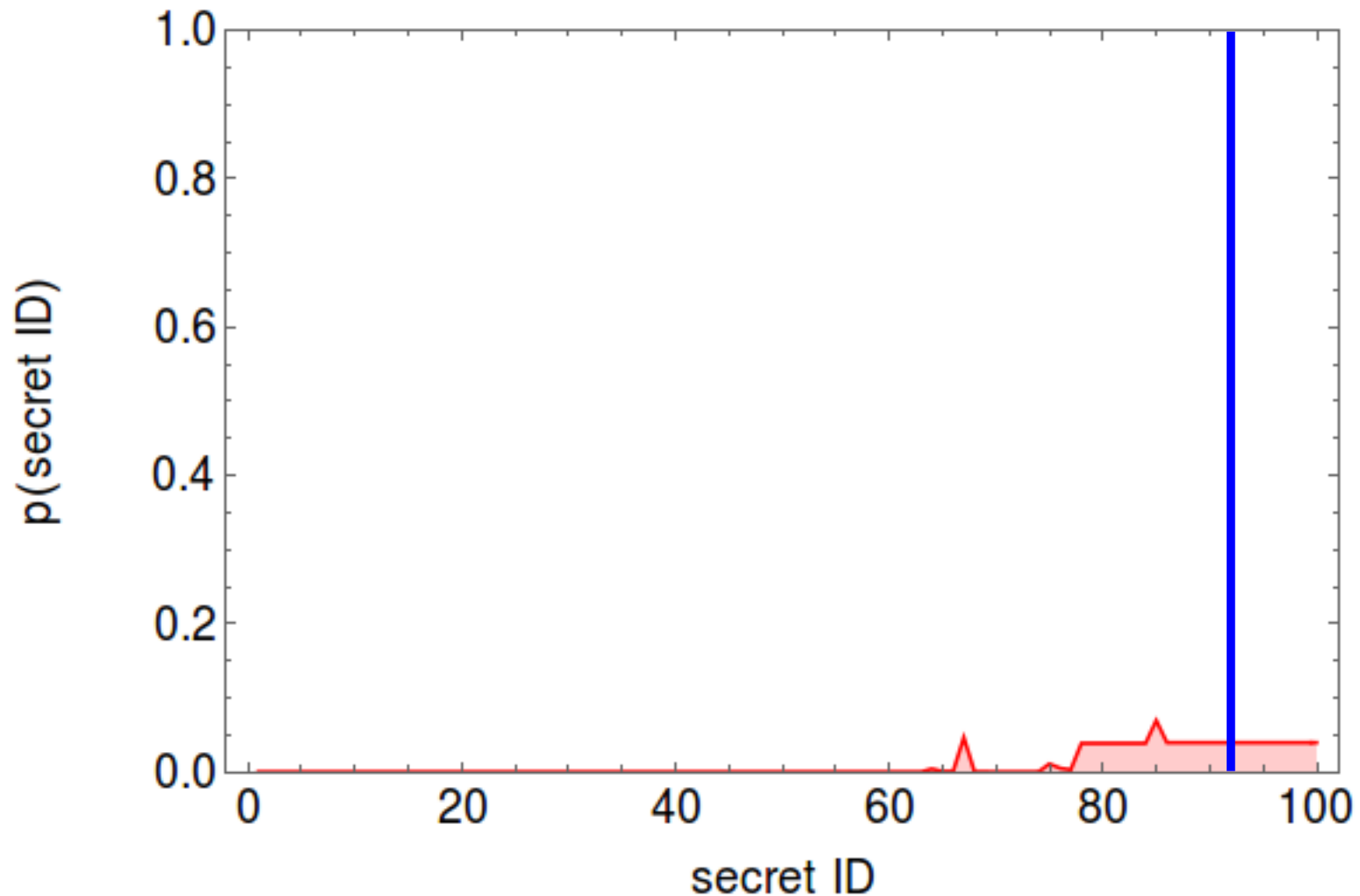


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 14: SEARCH 86 100

Observed time:0.00728

Entropy = 4.68363

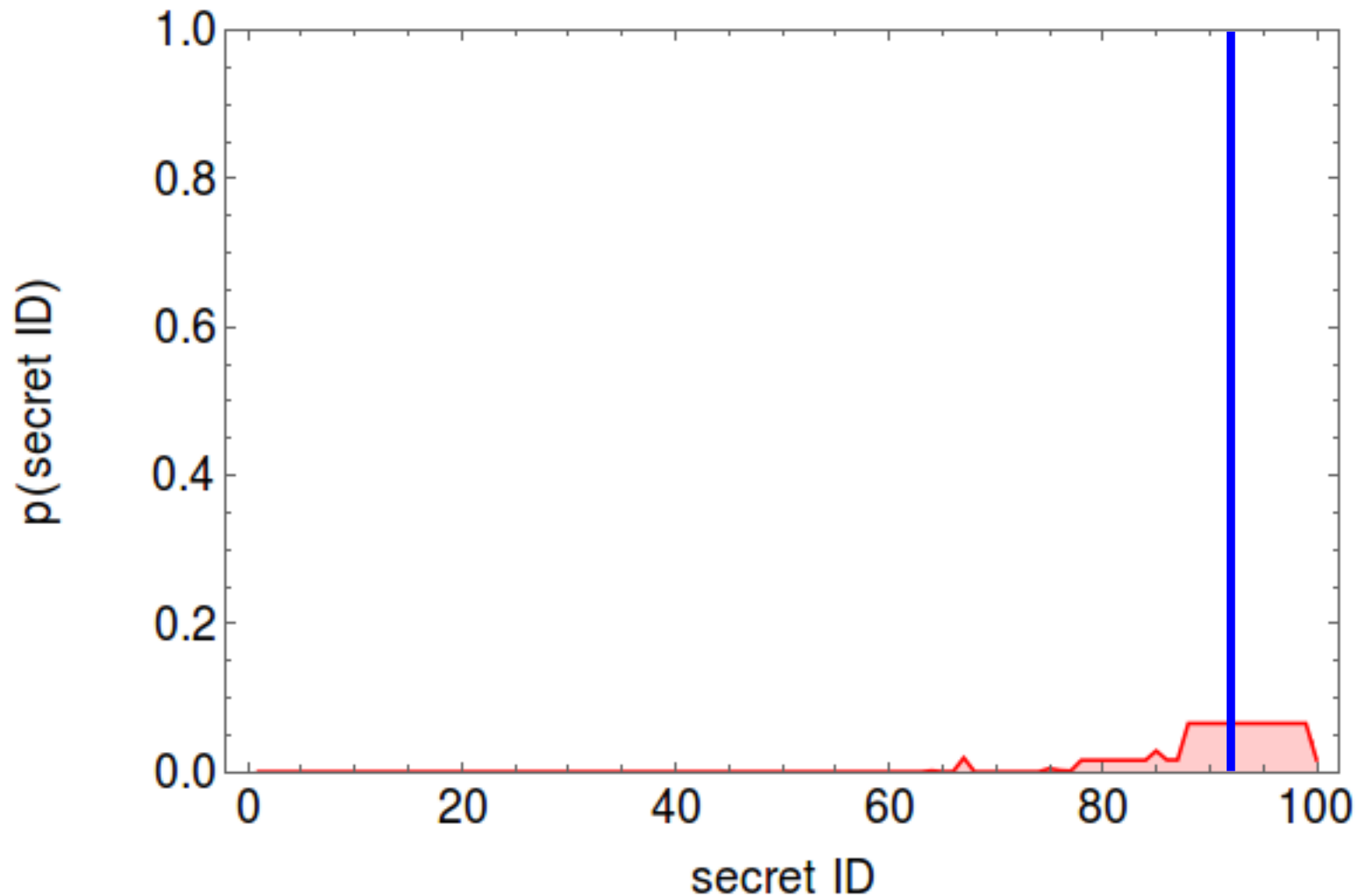


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 15: SEARCH 87 99

Observed time:0.00716

Entropy = 4.37901

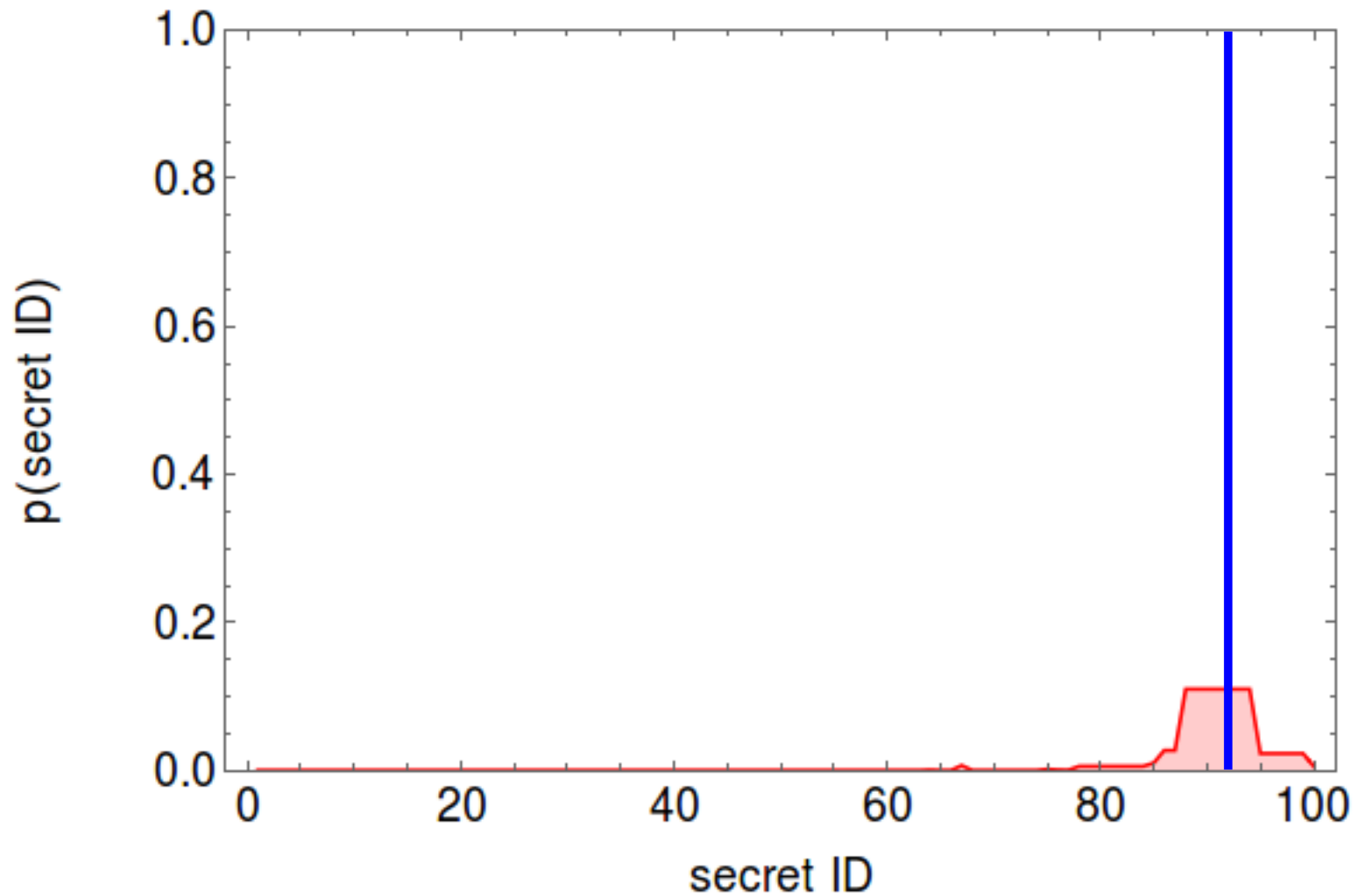


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 16: SEARCH 87 95

Observed time:0.00727

Entropy = 3.83405

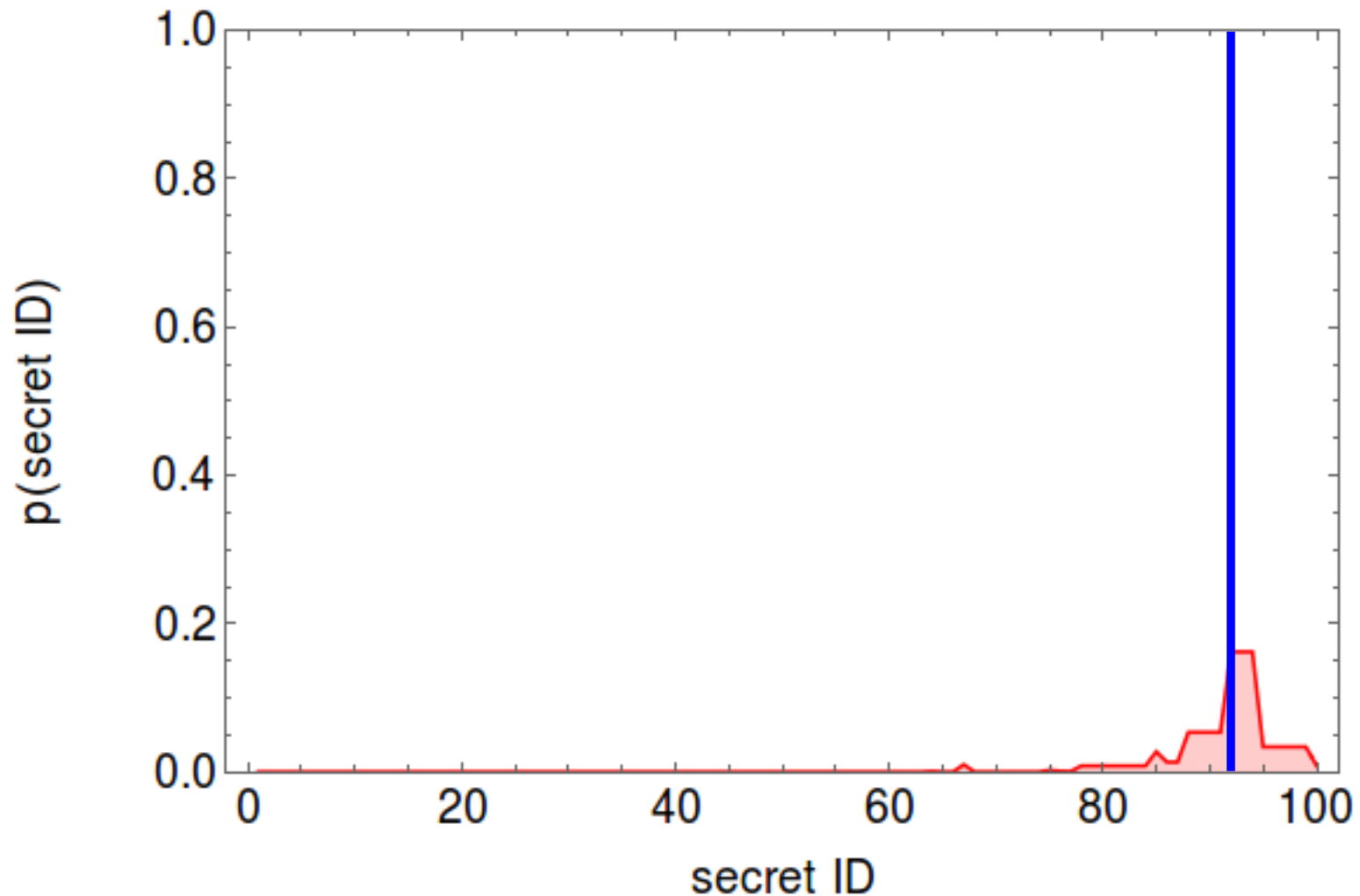


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 17: SEARCH 91 95

Observed time:0.00731

Entropy = 3.87438

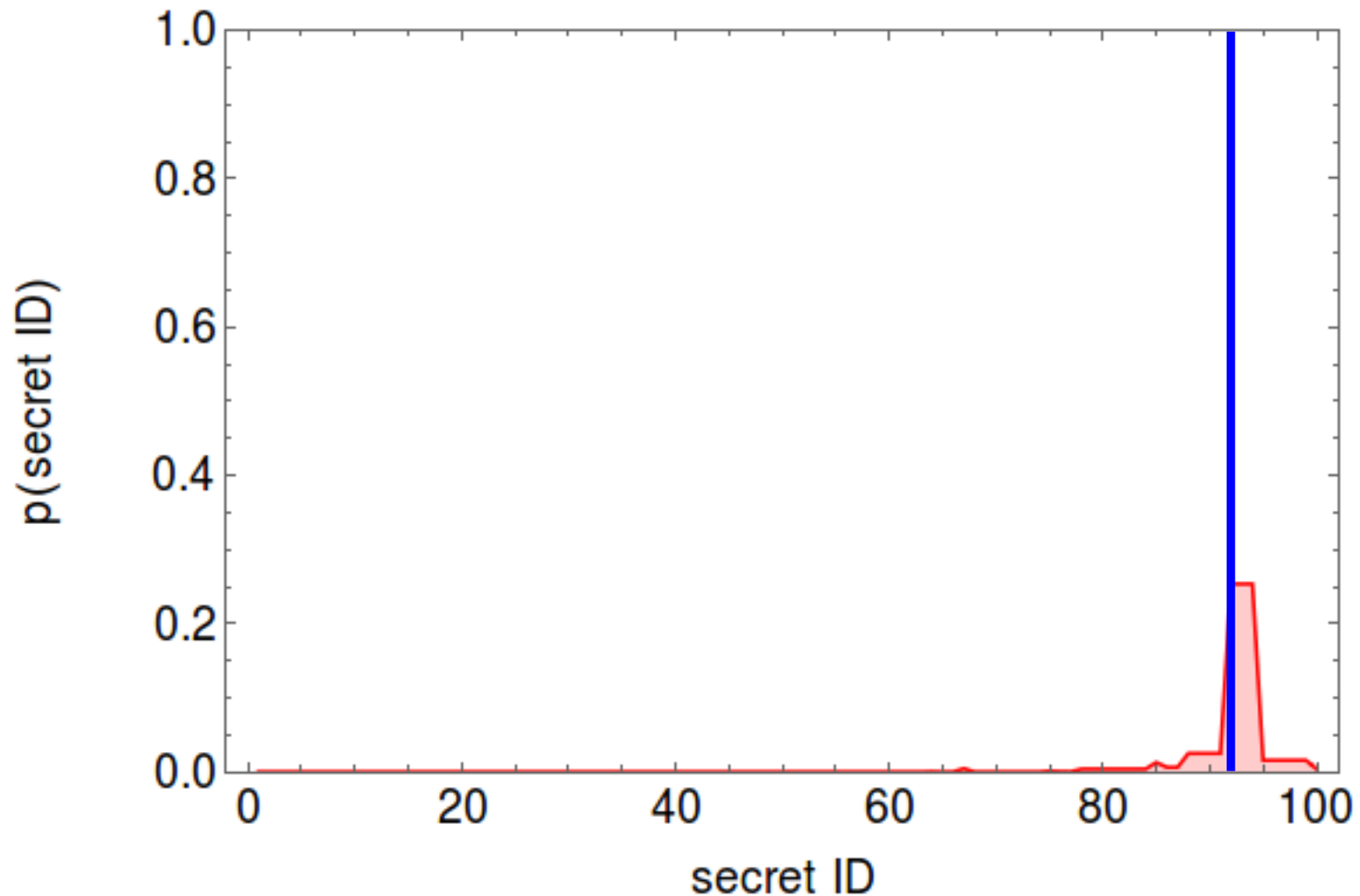


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 18: SEARCH 92 95

Observed time:0.0072

Entropy = 2.9822

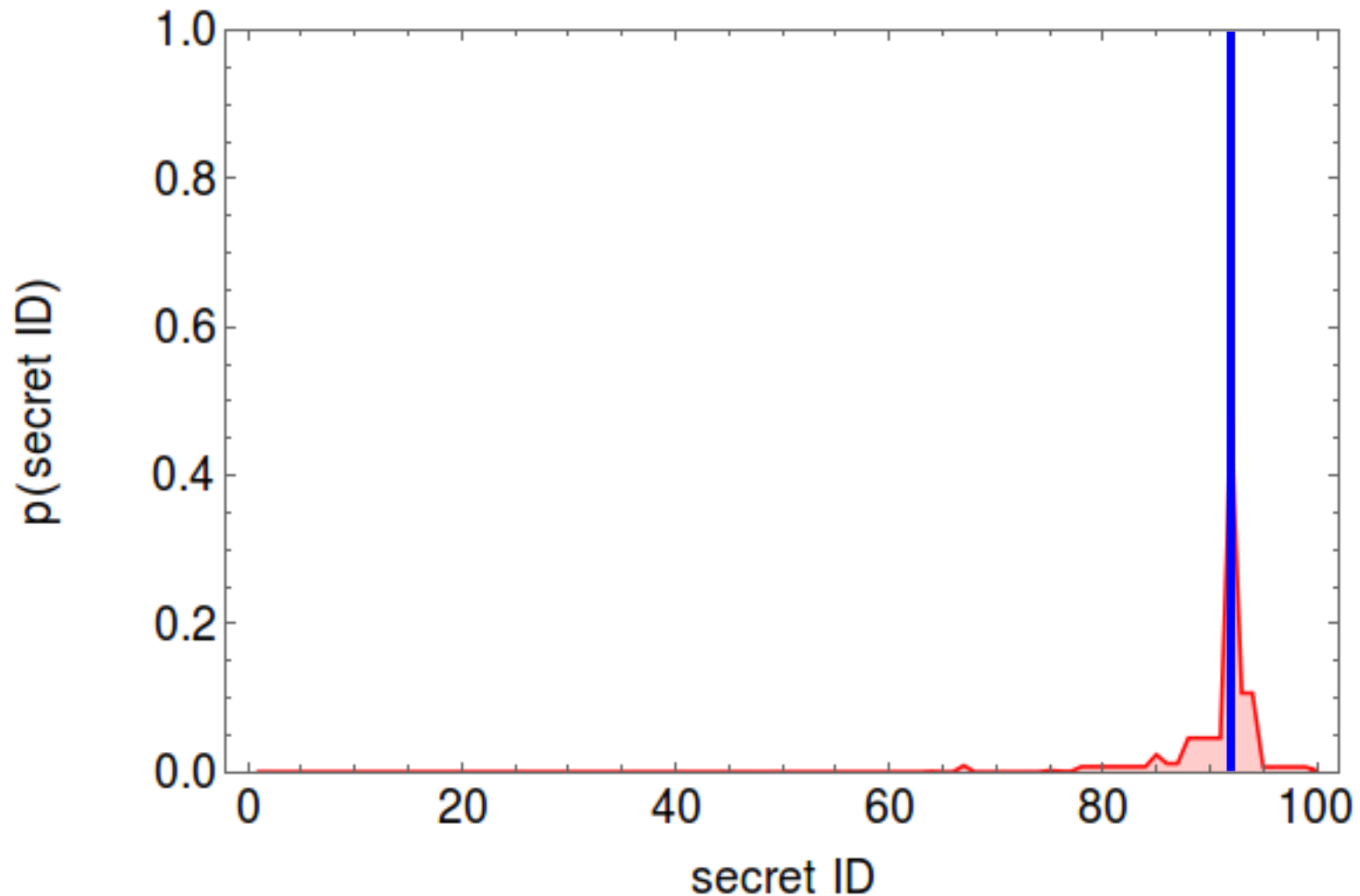


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 19: SEARCH 92 94

Observed time:0.00729

Entropy = 2.98878

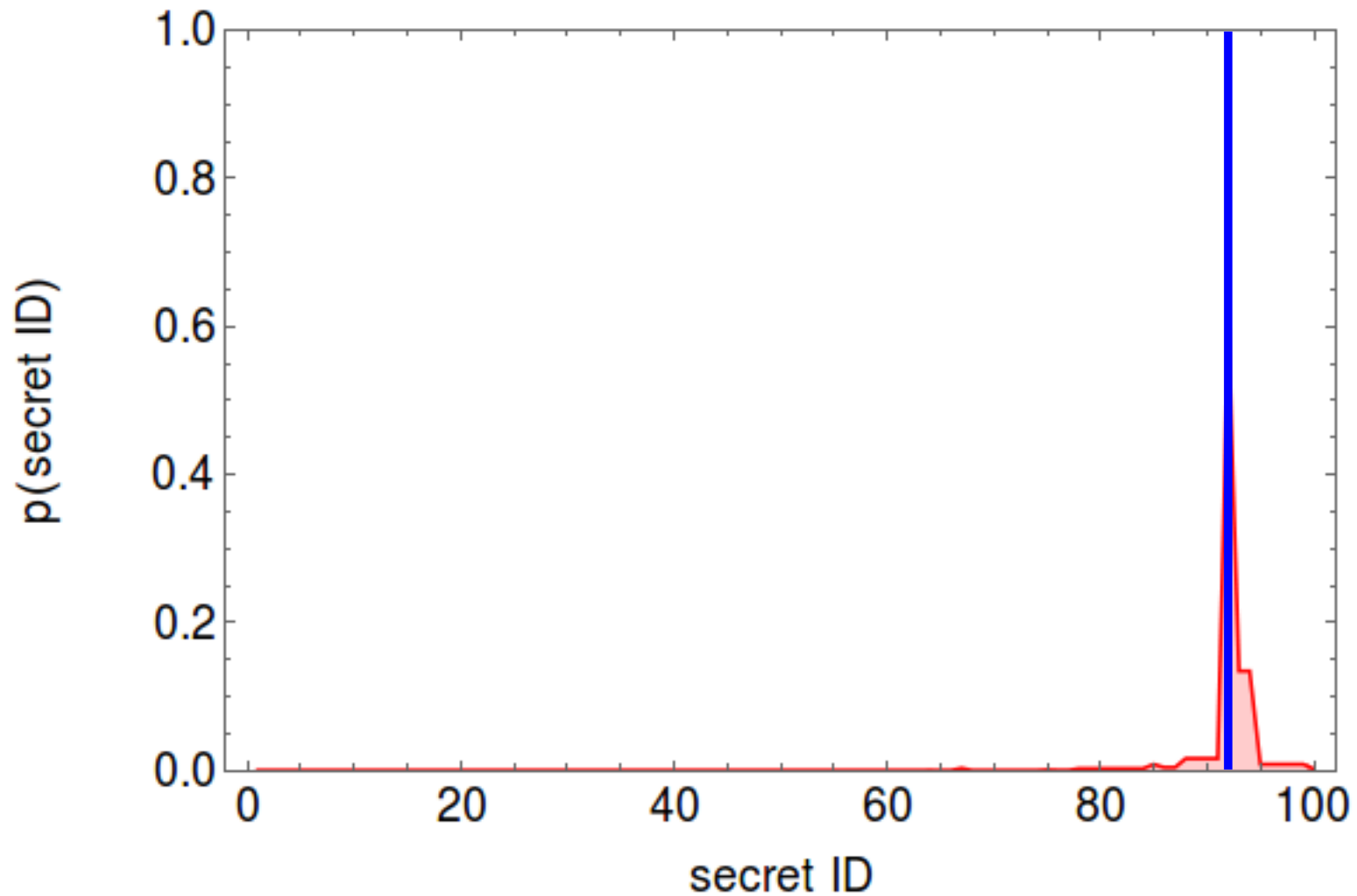


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 20: SEARCH 92 93

Observed time:0.00735

Entropy = 2.22644

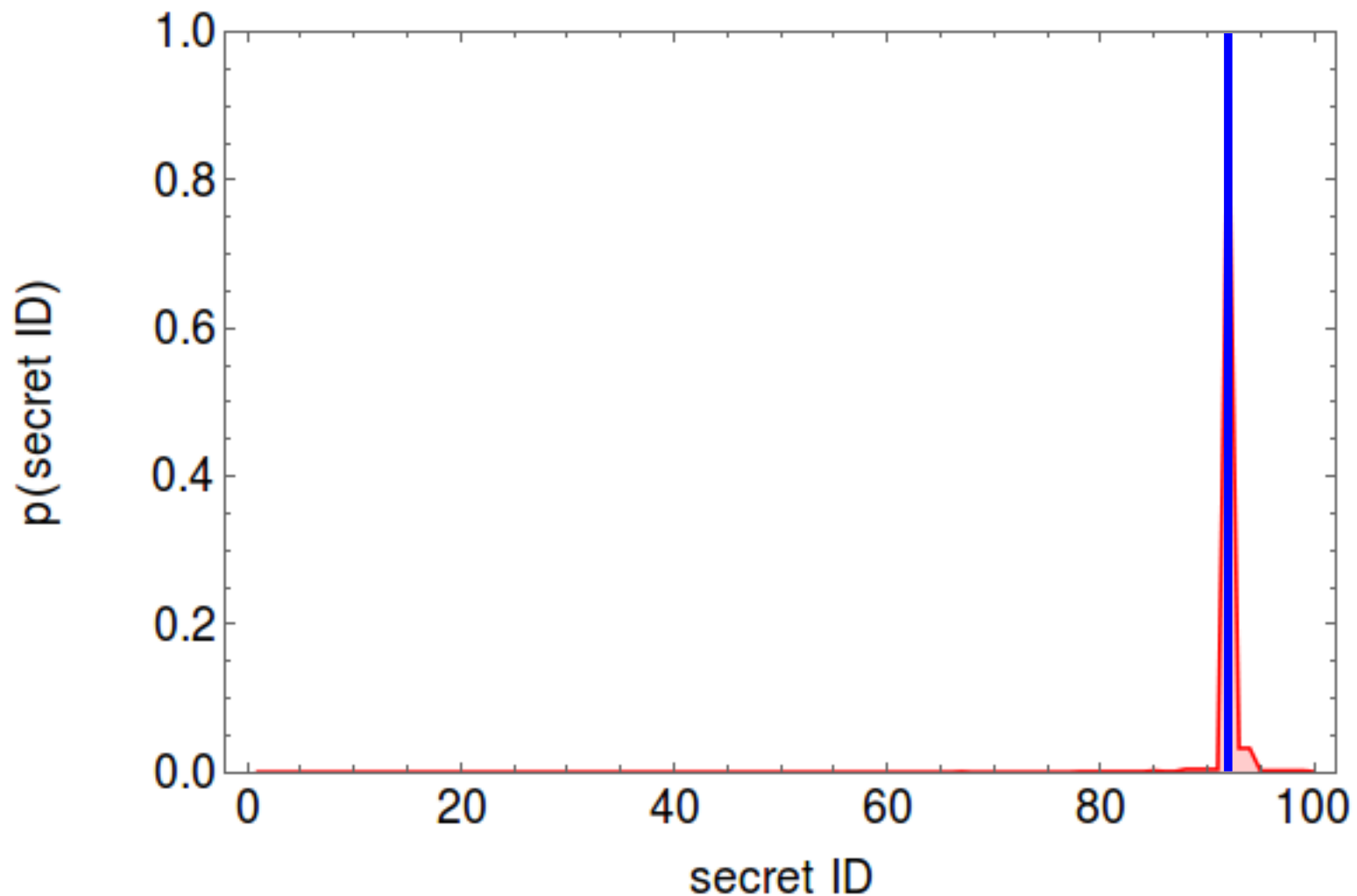


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 21: SEARCH 92 92

Observed time:0.00739

Entropy = 0.767476



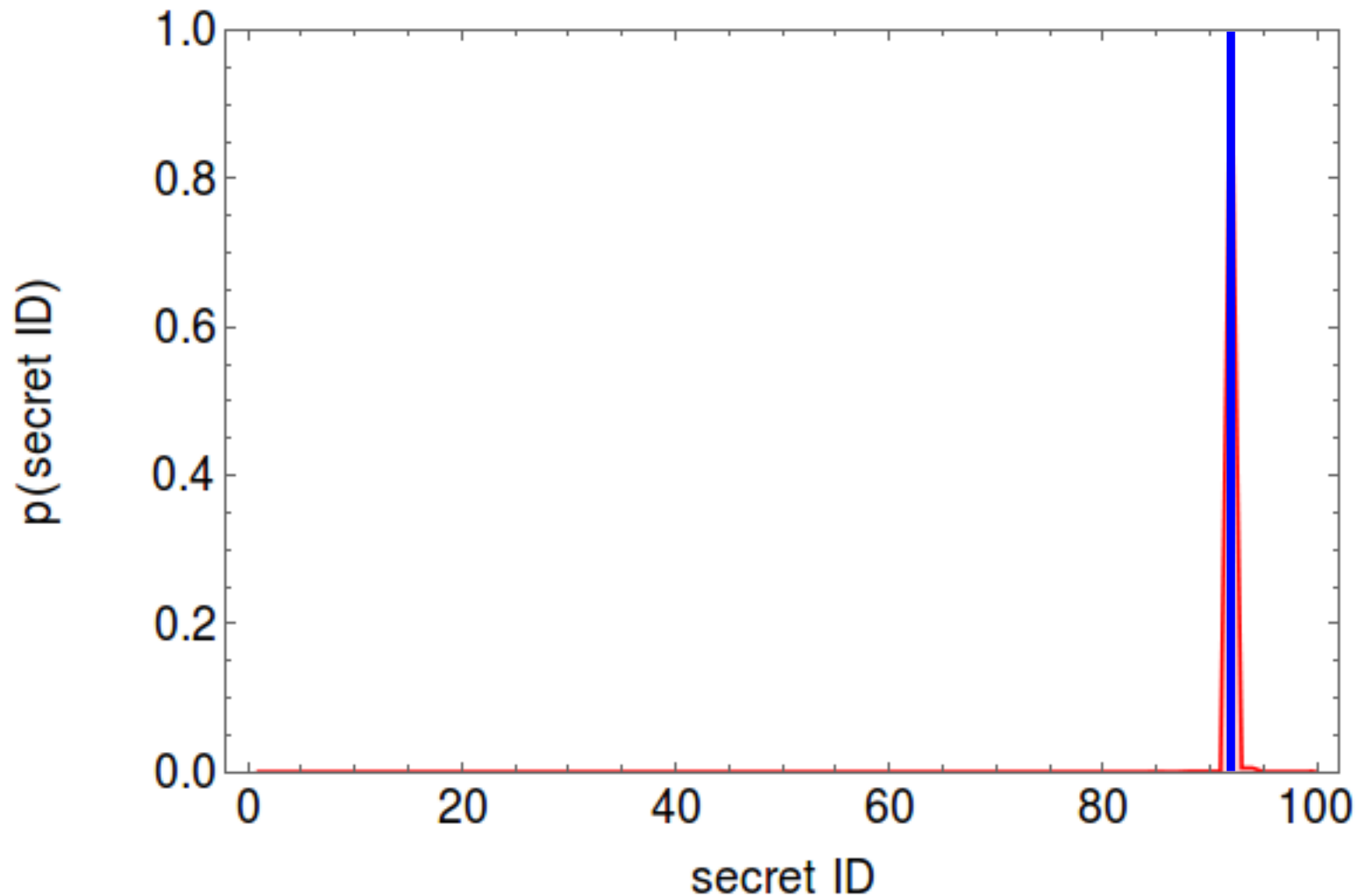


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 22: SEARCH 92 92

Observed time:0.00715

Entropy = 0.170871

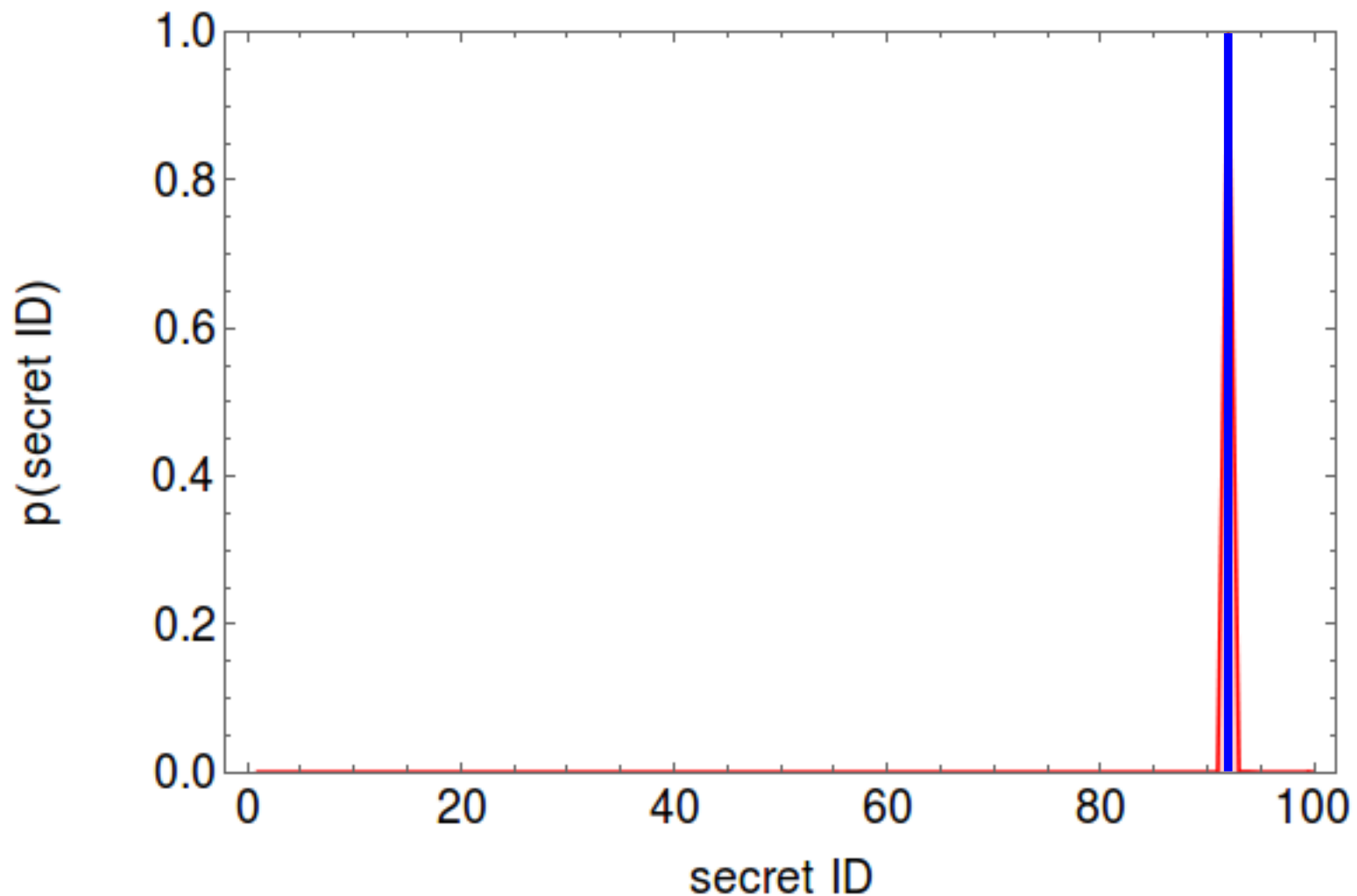


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 23: SEARCH 92 92

Observed time:0.00746

Entropy = 0.026079

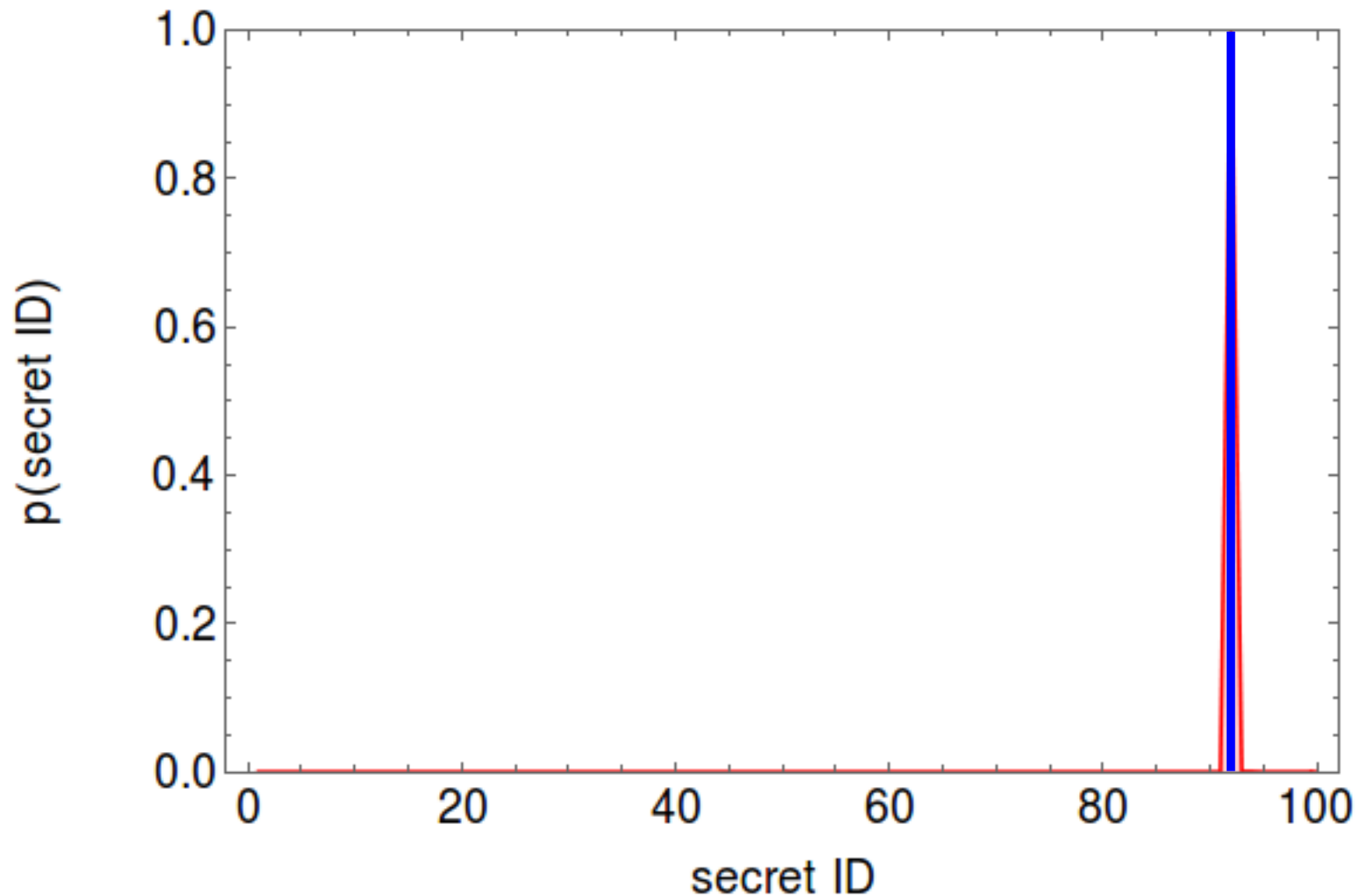


$$1 \leq ID \leq 100 \quad ID_1 = 64 \quad ID_2 = 85 \quad ID_{res} = 92$$

STEP 24: SEARCH 92 92

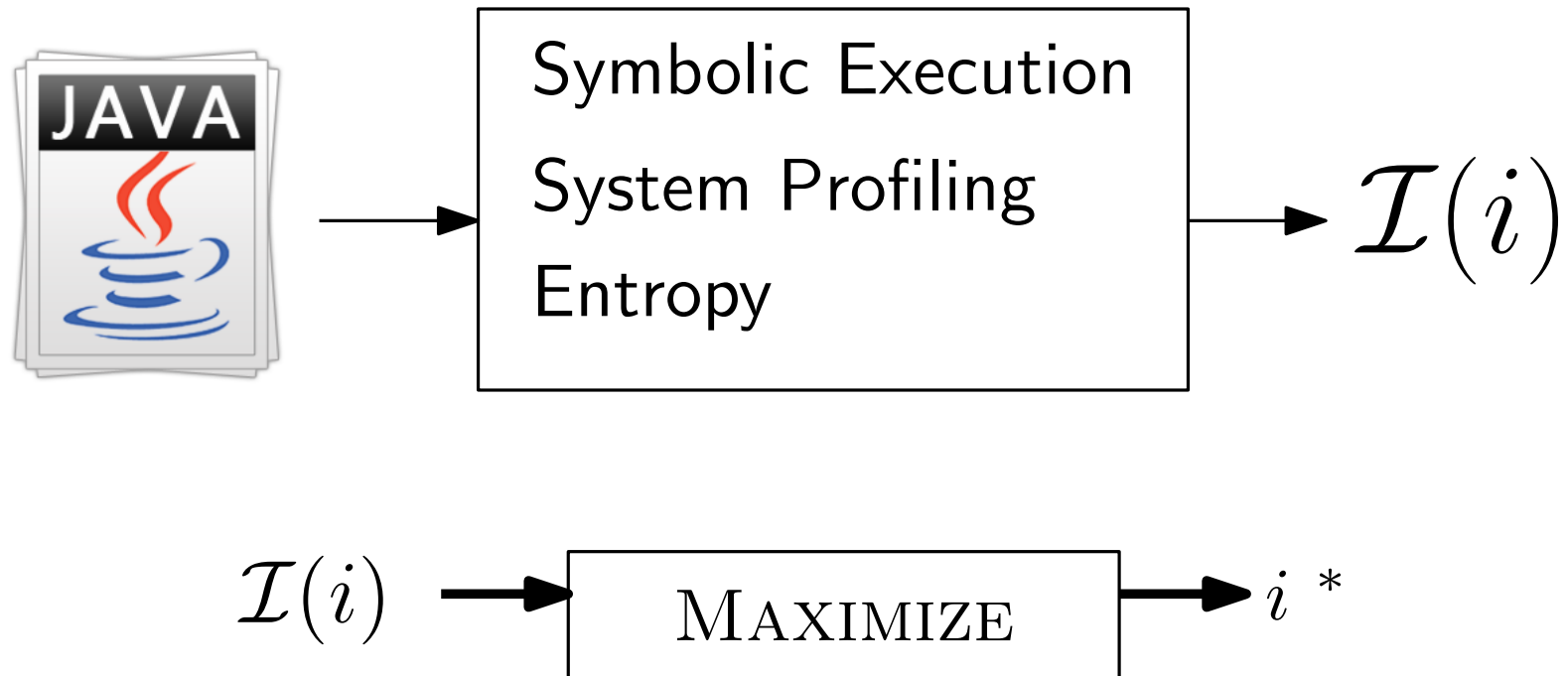
Observed time:0.00721

Entropy = 0.026084



ID Range	# Employees	Offline Analysis	Attack	
			time (m)	# steps
1-100	3	57s	2m38s	25
1-10000	4	2m21s	2m43s	45
1-10000	5	6m30s	3m08s	48
1-10000	10	42m09s	4m31s	77

# Our Approach



Automatically synthesize side-channel attacks!

Thanks!

Questions?