UMBC

AN HONORS UNIVERSITY IN MARYLAND

A Simulation of Document Detection Methods and Reducing False Positives for Private Stream Searching

Michael Oehler

University of Maryland Baltimore County (UMBC)

Data Privacy Management

13 September 2013

Motivation

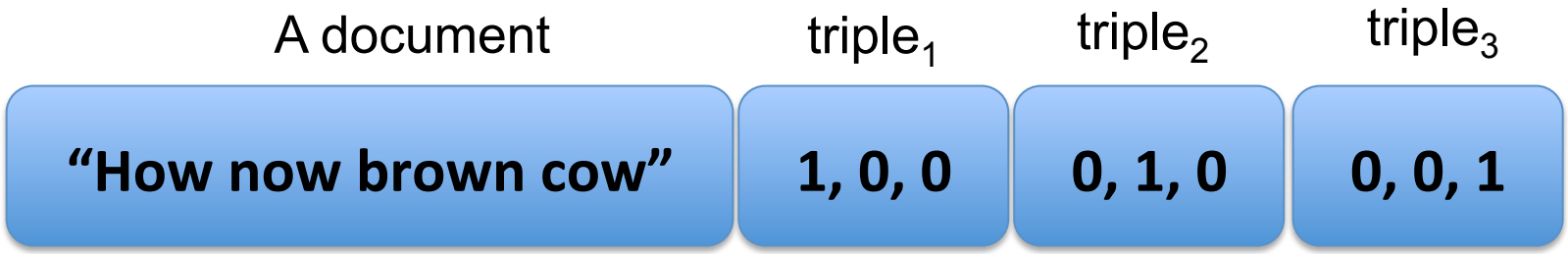
- Private Stream Searching
  - A system of cryptographic methods that
    - Preserves the confidentiality of the search terms and results
  - Operates over a stream of data
    - Results are saved to an output buffer
      - Collisions are possible. False Positives are possible.
      - False positives lead to a non-recoverable error.
- Build a "Better" Document Detection Method

Outline

- Concrete Example
- Introduction
- A “Better” Detection Method
- Simulation Results
- What’s Next

Concrete Example: Document Detection

- "Documents" are tagged with k bits
  - That are Partitioned into k/3 triples (3-bits each)
    - Each triple has a Hamming weight of 1
- A document is detected
  - If each triple has a Hamming weight of 1

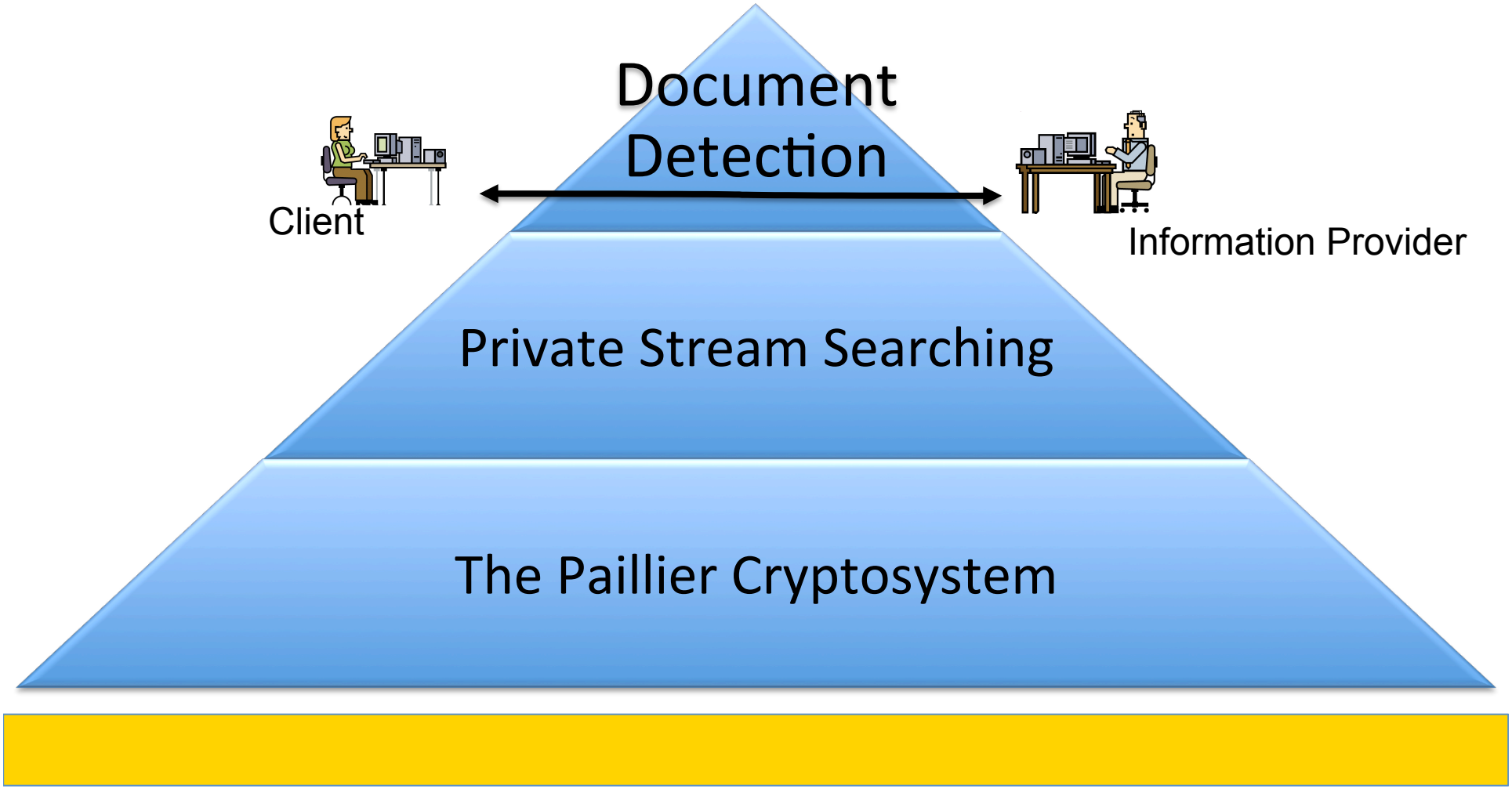


Concrete Example: False Positive

- The client receives an output buffer
  - Interprets this as a valid result

A document	$triple_1$	$triple_2$	$triple_3$
"!#@J%^U*&N!%K"	**1, 0, 0**	**0, 1, 0**	**0, 0, 1**

  - And (improperly) subtracts this false positive from the output buffer
    - Leading to an Error
    - Do "Better"?


Document Detection
Client
Information Provider
Private Stream Searching
The Paillier Cryptosystem

Paillier Encryption

Homomorphic Encryption

Multiplication in the encrypted domain is addition in the plaintext domain:

$$E(x) \times \mathrm{E}(y) \rightarrow \mathrm{E}(x+y)$$

Decryption Reveals the Summation of the plaintext

$$D(E(x+y)) = \mathrm{x+y}$$

The Leap

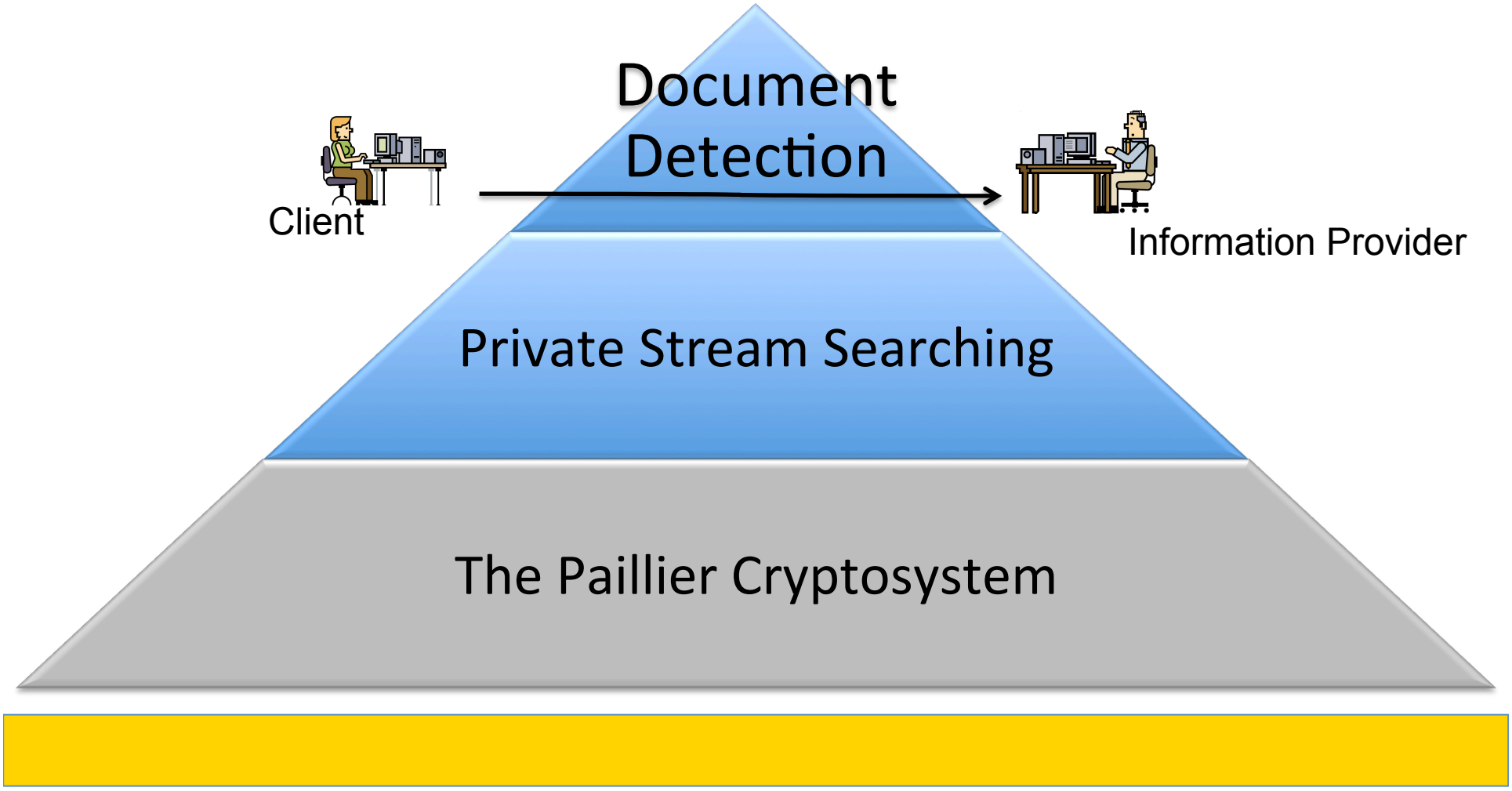
A **one** is used when the packet is of **interest**

$$\mathrm{E}(1)^{document} = E(document)$$

A **zero** is used when the packet is **not relevant**

$$\mathrm{E}(0)^{document} = E(0)$$

Exponentiation is just Multiplication… $\mathrm{E}(1)^{3} = E(1) \times E(1) \times E(1) = E(3)$


Document Detection
Client
Information Provider
Private Stream Searching
The Paillier Cryptosystem

Private Stream Searching

- Ostrovsky and Skeith
  - A system of cryptographic methods
  - That conceals the search criteria and result
  - Based extensively on the homomorphic property of the Paillier Cryptosystem

Private Stream Search: The Query

Client: The Query	
Define Public Dictionary D	$D = \{w_1, w_2, w_3, \ldots\}$
Define Private keywords K	$K \subseteq D$
Create an encrypted Keyword Filter F	$\forall f_i \in F \quad f_i = \begin{cases} E(1) & w_i \in K \\ E(0) & \text{otherwise} \end{cases}$
Send D and F to the Information Provider	

The Query

Create the Encrypted Keyword Filter F

Lets define two private keywords: “dog” and “lazy”

Set the dictionary entries:

“dog” is of interest, $f_3 = E(1)$

“lazy” is of interest, $f_4 = E(1)$

Dictionary	Encrypted Filter F
apple	$f_0=E(0)$
brown	$f_1=E(0)$
cat	$f_2=E(0)$
dog	$f_3=E(1)$
lazy	$f_4=E(1)$
sunshine	$f_5=E(0)$
vanilla	$f_6=E(0)$
zulu	$f_7=E(0)$

Information Provider: The search

Receives the Filter	F
Construct the output Buffer	$B = \{\{E(0), E(0)\}, \{E(0), E(0)\}\}$
For each Document do	$d = \text{"See the quick brown fox..."}$
Append the triples	$d' = d \,\|\|\, triples$
Extract dictionary words	$w \in d$
Calculate Match Value	$s = E(m_d) = \prod_k f_k \quad \forall w_k \in d \in D$
Calculate search result	$r = s^{d'} = E(m)^{d'} = E(m \times d')$
Save the result to buffer	$B = \{\{E(m), E(m \times d')\}, \{E(0), E(0)\}\}$
Send the buffer to the Defender	

The Search

Given a document with appended triples:

d' = See the quick brown fox jump over the lazy dog. | 4 | 2 | 1

Calculate the Encrypted Match Value for d':

$$E(m_d) = f_1 \cdot f_3 \cdot f_4$$

$$= E(0+1+1) = E(2)$$

Two words match

The encrypted filter, F

Dictionary	Encrypted Keyword Filter
apple	f_0=E(0)
brown	**f_1=E(0)**
cat	f_2=E(0)
dog	**f_3=E(1)**
lazy	**f_4=E(1)**
sunshine	f_5=E(0)
vanilla	f_6=E(0)
zulu	f_7=E(0)

The Search

Continuing our “quick brown fox” example

Calculate the result

$$r_i = s^{d_i} = E(2)^{d'}$$

$$= E(2 \times d')$$

Document and triples are scaled by the number of matching keywords: 2

The Search

Save the result

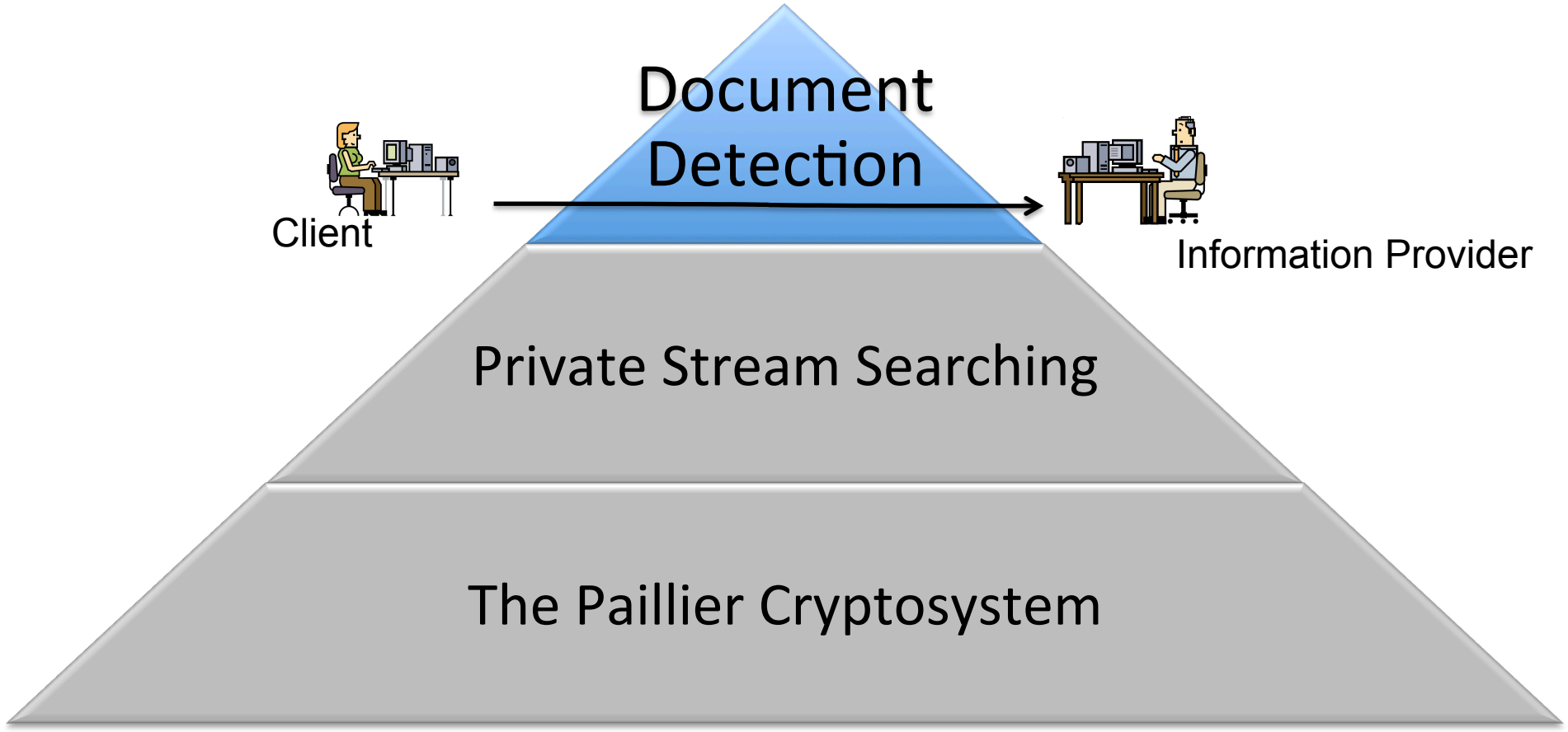
$$\{E(2), E(2) \times E(d')\}$$

Into **three** buffer positions:

Match Value	Document
E(2)	**E(2×d')**
0	0
0	0
E(2)	**E(2×d')**
E(2)	**E(2×d')**
0	0
0	0
0	0

Private Stream Search: The Result

Client: The Result	
Receive the Buffer	
Decrypt the Buffer	$B = D\left(\left\{\left\{E(2), E(2 \times d')\right\}, \left\{E(0), E(0)\right\}\right\}\right)$
Detect Document	To extract a matching document Each triple must have a Hamming weight of one
Return Matching Documents	$= \{\text{"See the quick brown fox..."}\}$


Document Detection
Client
Information Provider
Private Stream Searching
The Paillier Cryptosystem

False Positives with Triples

For example, let $d_1=576_8$. Let the triples be 4, 2, 1.
And there is a match on 2 keywords

576_8	4_8	2_8	1_8

Then $\{E(2), E(1375042_8)\}$ Gets saved

Let $d_2=675_8$. Let the triples be 1, 2, 4.
And a match on 5 keywords

675_8	1_8	2_8	4_8

Then $\{E(5), E(4261644_8)\}$ Gets saved

False Positives with Triples

If the provider saved both of these results to the same buffer position:

$$b_0 = \{E(2), E(1375042_8)\} \times \{E(5), E(4261644_8)\}$$
$$= \{E(7), E(5656706_8)\}$$

Multiple results can be saved to the same buffer position forming a linear combination

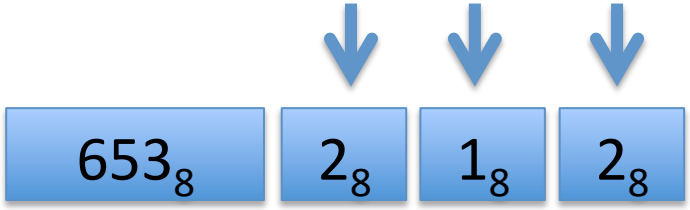
False Positives with Triples

Then and for Document Detection, the client divides:

$$= 5656706_8 / 7$$

$$= 653212_8$$

And Confirms the Hamming weight on each triple is 1



Returns the document:

653_8 a False Positive

UMBC

AN HONORS UNIVERSITY IN MARYLAND

Consequence: non-recoverable error

Client subtracts the false positive. Induces an error.
Preventing further document detection

Original Decrypted Output Buffer B

Match Value	Document
7	**653212**
0	0
0	0
1	888421
2	1776222
0	0
0	0
0	0

Subtract

After Subtracting The New Buffer
holds errant values

Match Value	Document
0	0
0	0
0	0
1	888421
-5	**1123010**
-7	**-653212**
0	0
0	0

A New Document Detection Method

- Information Provider Appends a k-bit truncated hash of the document:

$$d' = d \,||\, (H(d) \,\&\, (2^k - 1))$$

- Client Document Detection: If the append hash matches a calculated hash
  - Subtle change with big effect

Simulation: Parameters

- Simulation addresses a variety of parameters
  - Number of bits: k = {9,12,15,21, …}
    - the number of triples or size of the truncated hash
      - More bits decreases the rate of false positives
  - Number of documents (1000 in each trial)
    - Number of documents added in a collision (2,3,4,5)
  - Ten (10) trials over each set of 1000

UMBC

AN HONORS UNIVERSITY IN MARYLAND

Simulation: Results for Triples

Maximum number of false positives encountered per 1000 collisions

	1 Document Collision	2 Document Collision	3 Document Collision	4 Document Collision	5 Document Collision
9 bits 3 triples	13	36	93	58	64
12 bits 4 triples	6	1	40	23	21
15 bits 5 triples	3	5	24	10	10
18 bits 6 triples	1	3	14	5	4
21 bits 7 triples	1	1	6	3	1
24 bits 8 triples	0	1	3	1	1

UMBC

AN HONORS UNIVERSITY IN MARYLAND

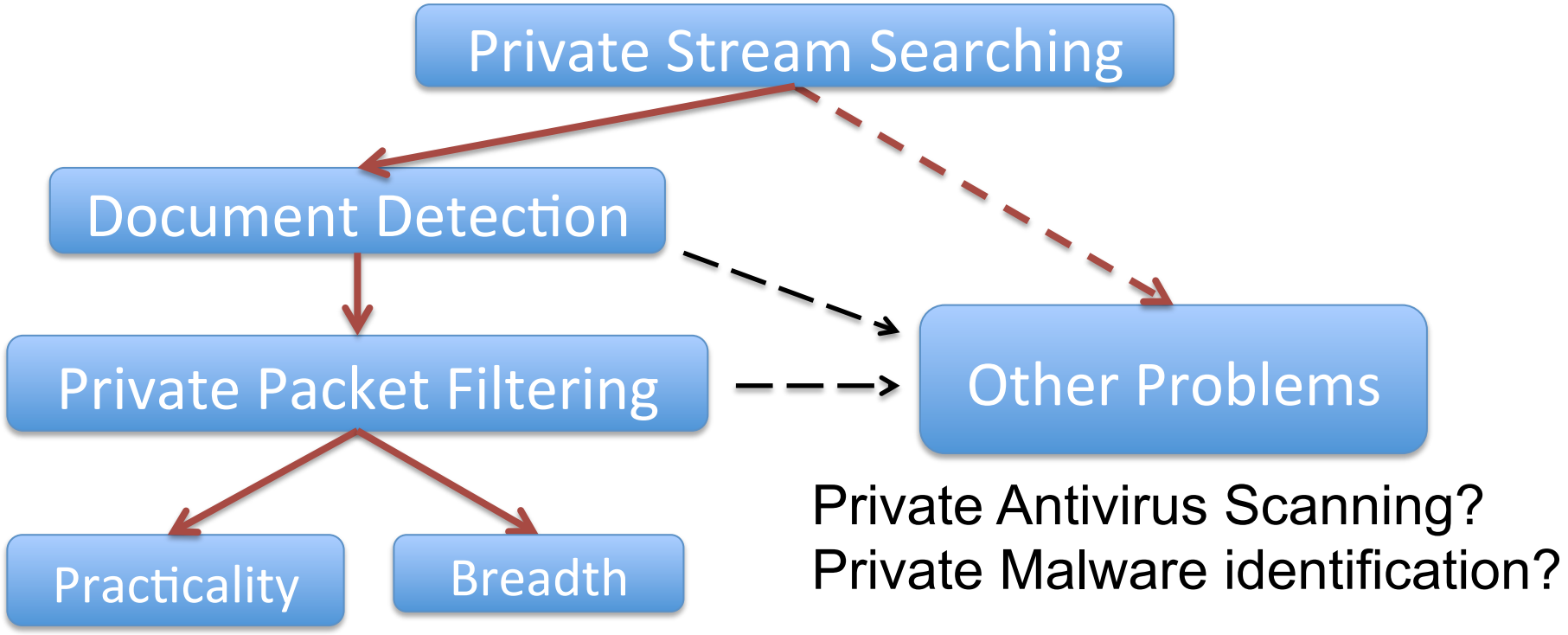
Simulation: Results for Truncated Hash

Maximum number of false positives encountered per 1000 collisions

	1 Document Collision	2 Document Collision	3 Document Collision	4 Document Collision	5 Document Collision
9 bits	2	1	1	0	1
12 bits	1	1	0	0	0
15 bits	0	0	0	0	0
18 bits	0	0	0	0	0
21 bits	0	0	0	0	0
24 bits	0	0	0	0	0

What's Next

- New Research directions are possible…



UMBC

AN HONORS UNIVERSITY IN MARYLAND

Michael Oehler

Michael.Oehler@umbc.edu

University of Maryland Baltimore County (UMBC)

Cyber Defense Laboratory (CDL)