BlindBox: Deep Packet Inspection over Encrypted Traffic

Justine Sherry, Chang Lan, Raluca Ada Popa, Sylvia Ratnasamy
Deep Packet Inspection

Many middleboxes inspect packet payloads by reconstructing TCP bytestream.

Parental Filtering

Intrusion Prevention

Exfiltration Detection
Example: Intrusion Prevention

Rule = description of an attack.
Challenge for DPI: Increasing use of Encryption
Is Your ISP Spying On You?

By Lincoln Spector, PCWorld

Sep 3, 2012 7:42 AM

Arctiscid asked the Answer Line forum if his ISP can "sit back... watch a screen, and see everything you are doing at any given time?"

Not quite, but it's frightening close. Your Internet service provider tracks what IP addresses you contact, which effectively means they know the web sites you're visiting. They can also read anything you send over the Internet that isn't encrypted. Whether they actually do that is an open question.

According to Dan Auerbach, a Staff Technologist for the Electronic Frontier Foundation, that the ISP collects "metadata--things like IP addresses and port
Bob does not get the benefits of Intrusion Prevention!
State of the Art Solution? Man in the Middle SSL

Bob has no privacy from middlebox!
Bob: Privacy + Functionality

- Bob has two very conflicting requirements.
  - Privacy
  - In-network functionality
- Can he have his cake and eat it too?
Yes!
BlindBox

The first system to allow DPI middle boxes to inspect traffic without decrypting the traffic.

Detects blacklisted keywords in encrypted connections.

Learns virtually nothing about connections that do not contain blacklisted keywords.
BlindBox

Strong Privacy Guarantee
Proof of security: only learns if and where suspicious substrings match.

Practical Network Performance
Achieves forwarding rates comparable to standard IDS deployments.

Wide Range of Functionality
Supports exact-match detection as well as regular expressions and scripted analysis.
Setup

“Rule Generator”
Threat Model

“Rule Generator”

- Generates rules correctly
- Runs detection functionality correctly
- Aims to protect one honest endpoint from another dishonest endpoint. Does not address two dishonest endpoints.
- Users should not learn McAfee’s ruleset.
- Runs detection functionality correctly but curious to see traffic contents ("honest but curious model")

Alice

Bob
BlindBox HTTPS: Overview

Alice’s Protocol Stack

Message

BB Handshake

SSL Encrypt

BlindBox Encrypt

SSL Traffic

Encrypted Tokens

SSL Decrypt

BlindBox Verify

Bob’s Protocol Stack

BLACKLIST

ATTACK

HACKS

183237
One Slide Introduction to BB Handshake

An exchange between the middlebox and the client at the beginning of a new connection.

- Clients know secret key “K”
- Middlebox learns $AES_K(kw)$ for every keyword in ruleset, iff keyword has a signature from rule generator.
- Middlebox does not learn $K$.
- Middlebox and Rule generator know rules.
- Clients do not learn rules.
- Based on two known techniques: Garbled Circuits and Oblivious Transfer.
BlindBox Encrypt: A Deterministic Strawman

Alice

BLACKLIST

HACKS

ATTACK

183237

AES

Bob

AES\textsubscript{K}(ATTACK)

AES\textsubscript{K}(EXAMPL)

EXAMPLE ATTACK

EXAMPLE

ATTACK

ATTACK

HACKS

183237
A problem with the strawman design

Deterministic Encryption leaks substring frequency.
<table>
<thead>
<tr>
<th>Approach</th>
<th>Security</th>
<th>Detection Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic Searchable Encryption</td>
<td>Weak</td>
<td>Fast: $O(\log(#\text{rules}))$</td>
</tr>
<tr>
<td>[Song et. al ’00]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Randomized Searchable Enc. [Song et. al ’00]</td>
<td>Strong</td>
<td>Slow: $O(#\text{rules})$</td>
</tr>
</tbody>
</table>

**Searchable Encryption Approaches**

**Approach**
- Deterministic Searchable Encryption
- Randomized Searchable Enc. [Song et. al ’00]

**Security**
- Weak
- Strong

**Detection Speed**
- Fast: $O(\log(#\text{rules}))$
- Slow: $O(#\text{rules})$
Searchable Encryption Approaches

<table>
<thead>
<tr>
<th>Approach</th>
<th>Security</th>
<th>Detection Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic Searchable Encryption</td>
<td>Weak</td>
<td>Fast: $O(\log(#\text{rules}))$</td>
</tr>
<tr>
<td>Randomized Searchable Enc. [Song et. al '00]</td>
<td>Strong</td>
<td>Slow: $O(#\text{rules})$</td>
</tr>
</tbody>
</table>

- **Approach**: The types of searchable encryption approaches are determined by their deterministic or randomized nature.
- **Security**: The security level can be weak or strong.
- **Detection Speed**: The speed of detection is either fast or slow.
Searchable Encryption Approaches

<table>
<thead>
<tr>
<th>Approach</th>
<th>Detection Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic Searchable Encryption</td>
<td>Fast: $O(\log(#\text{rules}))$</td>
</tr>
<tr>
<td>Randomized Searchable Enc. [Song et. al ’00]</td>
<td>Slow: $O(#\text{rules})$</td>
</tr>
</tbody>
</table>

BlindBox: speed of deterministic enc, security of randomization
BlindBox Encrypt: At the Client

At session start, Alice pre-computes an initial salt value \( \text{salt}_0 \).

For each term, Alice encrypts token with secret key \( k \).

Alice encrypts salt with \( \text{AES}_K(\text{token}) \) as key.

Tokens: \{I LOVE, LOVE, LOVE M, OVE MU, VE MUF, E MUFF, MUFFI, MUFFIN…\}
BlindBox Encrypt: At the Client

I LOVE MUFFINS. WHAT IS YOUR FAVORITE MUFFIN?

Alice

AES

AES $K(I LOVE)$

$salt_0$

I LOVE
BlindBox Encrypt: At the Client

I LOVE MUFFINS. WHAT IS YOUR FAVORITE MUFFIN?

AES

Use same salt for all tokens.
BlindBox Encrypt: At the Client

**Alice**

I LOVE **MUFFINS**. WHAT IS YOUR FAVORITE MUFFIN?

AES

AES

K

AES

**MUFFIN**

(salt₀)

**MUFFIN**
BlindBox Encrypt: At the Client

I LOVE MUFFINS. WHAT IS YOUR FAVORITE MUFFIN?

Set salt\_1 = salt\_0 + 1

Same encryption value never appears twice!
BlindBox Encrypt: Middlebox Setup

After Handshake, Clients sends middlebox \((\text{salt}_0)\)

- AES \((\text{salt}_0)\)

root

BLACKLIST

- ATTACK
- HACKS
- 18724
BlindBox Encrypt: Middlebox Setup

After Handshake, Clients sends middlebox (salt₀)

Store AES_k(keyword) and salt in tree.
BlindBox Encrypt: Middlebox Setup

After Handshake, Clients sends middlebox \((\text{salt}_0)\)
BlindBox Encrypt: Middlebox Setup

BlindBox can now detect matches in $O(\log(\#\text{rules}))$
On match:
* Respond to match
* Average rule requires three matches to detect attack.
* Then update tree.
BlindBox Encrypt: Middlebox Setup

To Update:

AES (x + 1)

BLACKLIST

root

0xe46...

0x1ae...

0x1ag...

0xe13...

0xe47...
BlindBox Encrypt: Middlebox Setup

To Update:

- AES (ơ + 1)

BLACKLIST
Making BlindBox Encrypt Secure and Fast

<table>
<thead>
<tr>
<th>Approach</th>
<th>Security</th>
<th>Detection Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic Searchable Encryption</td>
<td>Weak</td>
<td>Fast: O(log(#rules))</td>
</tr>
<tr>
<td>Randomized Searchable Enc. [Song et. al '00]</td>
<td>Strong</td>
<td>Slow: O(#rules)</td>
</tr>
<tr>
<td>BlindBox</td>
<td>Strong</td>
<td>Fast: O(log(#rules))</td>
</tr>
</tbody>
</table>
BlindBox HTTPS: Recap

- **Message**: Alice sends a message to Bob.
  - **SSL Encrypt**: The message is encrypted using SSL.
  - **BlindBox Encrypt**: The encrypted message is then passed to BlindBox.
  - **Handshake**: The BlindBox establishes a secure connection with Bob.

- **BB receives encrypted rules**: BlindBox decrypts the encrypted rules it receives from Bob.
- **SSL Traffic**: Secure transaction between Alice and BlindBox.
- **Encrypted Tokens**: Tokens are random, but BlindBox can still do fast, exact-match lookups.
- **SSL Decrypt**: Bob decrypts the message to receive it in plaintext.
Supporting Regular Expressions

Today: discussed exact match detection.
In paper: how to handle regular expressions and scripts.

Key Idea: “Probable Cause Privacy”
A weaker privacy model which allows scripted analysis.
See our paper for:

Optimizations to reduce bandwidth overhead.

Details on BB Handshake, Garbled Circuits, and Oblivious Transfer.

Detailed evaluation and comparison against alternative crypto schemes.

“Exact Match” vs “Probable Cause” Privacy Models.

See our paper or come chat with me after the session!
Evaluation Highlights: Functionality & Performance
## Evaluating Functionality

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Without probable</th>
<th>With probable cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document watermarking</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Parental filtering</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Snort community (HTTP)</td>
<td>67%</td>
<td>100%</td>
</tr>
<tr>
<td>Snort Emerging Threats (HTTP)</td>
<td>42%</td>
<td>100%</td>
</tr>
<tr>
<td>StoneSoft (McAfee) IDS</td>
<td>40%</td>
<td>100%</td>
</tr>
<tr>
<td>LastLine IDS</td>
<td>29%</td>
<td>100%</td>
</tr>
</tbody>
</table>
Performance Highlights

**Forwarding Rate**
186Mbps
Comparable to Snort in existing IDS deployments.

**Setup Time**
97s for LastLine
Reasonable for long-lived/persistent connections ONLY

**Page Load Times**
+15-100%
Within normal variation depending on conn. quality.

3 orders of magnitude faster than Searchable Enc.
10 orders of magnitude faster than Functional Encryption.
Conclusion

BlindBox: the first system to allow DPI middleboxes to inspect traffic without decrypting the traffic.

Future work: Can we generalize BlindBox to a protocol to support all middleboxes without sacrificing privacy?

contact: justine@eecs.berkeley.edu | @justinesherry
In Comparison: mcTLS and BlindBox

**mcTLS:**
Allows MB to read arbitrary values from fields client chooses to reveal.

**BlindBox:**
Allows MB to read suspicious keywords only from entire bytestream.

**Example Messages:**
- THIS IS AN EXAMPLE ATTACK MESSAGE!
- THIS IS AN EXAMPLE INNOCENT MESSAGE!
Download Times

Home Networks:

Datacenter Network:
Bandwidth Inflation

CDF

Tokenization Overhead Ratio

Delim Tokenization : Plaintext
Window Tokenization : Plaintext
Delim Tokenization : gzip
Window Tokenization : gzip

14x

2.5x
Can’t functional encryptions solve this?

- Existing schemes don’t fit our needs:
  - Wrong security model: all parties learn all of the middlebox rules
  - Missing functionality: no approach to address rules which are regular expressions
  - Prohibitive performance: Performing IDS detection over a single packet requires over 1 day of computation on our servers!

Microbenchmarks

<table>
<thead>
<tr>
<th>Client</th>
<th>Encrypt (128 bits)</th>
<th>Encrypt (1500 bytes)</th>
<th>Setup (1 Keyword)</th>
<th>Setup (3K Rules)</th>
<th>Detection:</th>
<th>1 Rule, 1 Token</th>
<th>1 Rule, 1 Packet</th>
<th>3K Rules, 1 Token</th>
<th>3K Rules, 1 Packet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>13ns</td>
<td>3µs</td>
<td>73ms</td>
<td>73ms</td>
<td>NP</td>
<td>NP</td>
<td>NP</td>
<td>NP</td>
<td>NP</td>
</tr>
<tr>
<td></td>
<td>70ms</td>
<td>15s</td>
<td>N/A</td>
<td>N/A</td>
<td>170ms</td>
<td>36s</td>
<td>8.3 minutes</td>
<td>5.7 days</td>
<td>157ms</td>
</tr>
<tr>
<td></td>
<td>2.7µs</td>
<td>257µs</td>
<td>N/A</td>
<td>N/A</td>
<td>1.9µs</td>
<td>52µs</td>
<td>5.6ms</td>
<td>157ms</td>
<td>20ns</td>
</tr>
<tr>
<td></td>
<td>69ns</td>
<td>90µs</td>
<td>N/A</td>
<td>N/A</td>
<td>20ns</td>
<td>5µs</td>
<td>137ns</td>
<td>33µs</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Connection and detection micro-benchmarks comparing Vanilla HTTPS, the functional encryption (FE) strawman, the searchable strawman, and BlindBox HTTPS. NP stands for not possible. The average rule includes three keywords.