SECRET: On the Feasibility of a Secure, Efficient, and Collaborative Real-Time Web Editor

Dennis Felsch, Christian Mainka, Vladislav Mladenov, Jörg Schwenk
Real-Time Web Editing Tools
Operational Transforms (OT)

• Maintain a consistent view on a document
• Automatically resolve editing conflicts
• Whole area of research on its own
Motivation

• Established tools do not apply (cryptographic) protection to documents
• Previous academic proposals with encryption either
  – Require large overheads
  – Are not real-time collaborative,
  – Require browser extensions, or
  – Do not take structure into account
• Is it feasible to have all these properties?
SECRET

- First **Secure**, **Efficient**, and **Collaborative** Real-time **Editor**
- SECRET is the first collaboration tool with
  1. encryption of whole documents or arbitrary sub-parts,
  2. novel combination of tree-based OT and structure preserving encryption,
  3. only a modern browser without any extra software installation or browser extension required
Building Blocks

• ShareJS
  – JavaScript Middleware with OT algorithms
• State-of-the-Art Web Technologies
  – WebSockets for Asynchronous Messaging
  – W3C Web Cryptography API for AES-128 in Galois Counter Mode (GCM)
  – PostMessage API
• XML Encryption
  – Structure Preserving Encryption
XML Encryption

<PaymentInfo>
  <Name>John Smith</Name>
  <CreditCard Limit="5,000" Currency="USD">
    <Number>1234 5678 2580 1595</Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/17</Expiration>
  </CreditCard>
</PaymentInfo>

<PaymentInfo>
  <Name>John Smith</Name>
  <EncryptedData
    Type="http://www.w3.org/2001/04/xmlenc#Element"
    xmlns="http://www.w3.org/2001/04/xmlenc#">
    <CipherData>
      <CipherValue>184797A8C2FE977DEFA10A7FE540A0D0</CipherValue>
    </CipherData>
  </EncryptedData>
</PaymentInfo>
Implementation Challenges

• ShareJS does not support XML
  – Solution: Implemented OT for XML documents as an extension of ShareJS

• Browsers do not support XML Encryption
  – Solution: Implemented a JavaScript library to encrypt, decrypt, sign, or verify documents

• WebCrypto API does not handle long-lived, persistent keys
  – Solution: Store them on an key-server or derive them from a password
SECRET: Secure, Efficient, and Collaborative Real-Time Web Editor

Split Size: 4

Part 1 - Encrypted

abcdef

Part 2 - Encrypted

xyzikopq

Part 3 - Plaintext

123

Please use the following credentials:

- **Username**: john.doe
- **Password**: 12345

Username:
Password:
Log on

Show me the ciphertext
Documents

Diagram showing a tree structure with nodes labeled as follows:
- Document
  - Encrypted Block (EB)
    - Encrypted Block (EB)
      - Encrypted Keys
        - Wrapped BK for group A
    - Unencrypted Block (UB)
      - Encrypted Block (EB)
      - Encrypted Data (ED)
        - Wrapped BK for group B
  - Unencrypted Block (UB)
    - Encrypted Block (EB)
      - Encrypted Data (ED)
Screenshot Ciphertext
Splitting into Encrypted Data Chunks

• Large encrypted blocks
  ⇒ Updates are inefficient
  ⇒ Splitting is necessary

• Small encrypted blocks
  ⇒ Large XML overhead

Q: What is the optimal split size?

<div>
  <span>Hello</span>
  <span>World</span>
</div>
Evaluation

- Google Chrome 50 with Selenium
- Simulated typing at 200 key strokes / min
- Measured storage and network overhead
Evaluation Storage

![Graph showing encrypted document size vs content text size for different split sizes.]
Evaluation Storage

<table>
<thead>
<tr>
<th>Split size</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage expansion</td>
<td>3.50</td>
<td>2.46</td>
<td>1.92</td>
<td>1.66</td>
<td>1.53</td>
</tr>
</tbody>
</table>

Table 2: Ciphertext expansion of a 4096 byte document.

- Numbers look high
- In fact, they are far below the numbers in related work
- Best results before ours: 3.75 – 4.82
Evaluation Network

![Chart showing network traffic against appended text block size with different split sizes.]

- Red line: Split size = 32
- Green line: Split size = 64
- Blue line: Split size = 128
- Purple line: Split size = 256
- Orange line: Split size = 512
- Dashed black line: Google Docs

**Network Traffic (Byte) vs. Appended Text (Byte)**
Evaluation Network

Network Traffic (Byte) vs. Encrypted Block

- Red: Split size = 32
- Green: Split size = 64
- Blue: Split size = 128
- Purple: Split size = 256
- Orange: Split size = 512

Google Docs

Appended Text (Byte)
Bandwidth Requirements

<table>
<thead>
<tr>
<th>Protokoll</th>
<th>Bits/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame</td>
<td>14 k</td>
</tr>
<tr>
<td>Ethernet</td>
<td>14 k</td>
</tr>
<tr>
<td>Internet Protocol Version 4</td>
<td>14 k</td>
</tr>
<tr>
<td>Transmission Control Protocol</td>
<td>14 k</td>
</tr>
<tr>
<td>Hypertext Transfer Protocol</td>
<td>12 k</td>
</tr>
<tr>
<td>Line-based text data</td>
<td>3557</td>
</tr>
<tr>
<td>HTML Form URL Encoded</td>
<td>8802</td>
</tr>
</tbody>
</table>

Figure 7: Screenshot from *Wireshark* measuring the required bandwidth for SECRET at 200 key strokes per minute with a split size of 128 bytes.
Conclusion & Outlook

• A Secure, Efficient, and Collaborative Real-Time Web Editor is feasible
• No need for large overheads when using Structure Preserving Encryption
• GUI and editing features can be improved
• How about full-fledged office documents?
• SECRET’s code is on GitHub: https://github.com/RUB-NDS/SECRET/
Questions?

Dennis Felsch
Horst Görtz Institute for IT-Security
Chair for Network and Data Security
Ruhr-University Bochum

dennis.felsch@rub.de

Twitter: @dfelsch

NDS-Blog:
http://web-in-security.blogspot.de