On the Security of RC4 in TLS

Nadhem AlFardan, Royal Holloway, University of London;
Daniel J. Bernstein, University of Illinois at Chicago and Technische Universiteit Eindhoven;
Kenneth G. Paterson, Bertram Poettering, and Jacob C.N. Schuldt,
Royal Holloway, University of London
Outline

- Background: TLS and RC4
- Weaknesses in RC4
- Attacks on RC4 in TLS
- Conclusions and Mitigations
Outline

• **Background: TLS and RC4**

• Weaknesses in RC4

• Attacks on RC4 in TLS

• Conclusions and Mitigations
TLS Background

- Started as SSL at Netscape
- Later became TLS
- Transit security
TLS Background

**Handshake**
- Choose public key algorithm (RSA, DHE, ECDHE, etc.)
- Choose symmetric key algorithm (AES-CBC, RC4, AES-GCM, etc.)
- HMAC (MD5, SHA-1, SHA-256)
- Exchange keys
- Authentication (maybe)

**Record**
- Use symmetric algorithm to encrypt data
- Use HMAC to authenticate records
RC4 Background

• Stream cypher

• Old (generated in 1987, leaked in 1994)

• Dead simple, very fast

• Recently popular because of Lucky13, BEAST, etc.
RC4 Background

Key-scheduling stage

- Use key to generate permutation of 256 bytes
- Key is usually between 40 and 256 bits
- TLS uses 128 bit keys

Key-stream Generator

- Use bit vector to generate pseudo random bits
- XOR generated bits against plaintext to generate cypher text
Outline

• Background: TLS and RC4

• **Weaknesses in RC4**

• Attacks on RC4 in TLS

• Conclusions and Mitigations
Weaknesses in RC4

- Stream ciphers are approximations of one time pads
- Reusing one time pads harms security
- Non-random PRNG is similar to reusing pads
Weaknesses in RC4

- RC4 has well known biases
- Single and multiple byte
- Some long established, some unique to paper
RC4: First Byte
RC4: Second Byte
RC4: Third Byte
## RC4: Multi-byte Biases

<table>
<thead>
<tr>
<th>Byte pair</th>
<th>Condition on $i$</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 0)</td>
<td>$i = 1$</td>
<td>$2^{-16}(1 + 2^{-9})$</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>$i \neq 1, 255$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>$i \neq 0, 1$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>($i + 1, 255$)</td>
<td>$i \neq 254$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(255, $i + 1$)</td>
<td>$i \neq 1, 254$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(255, $i + 2$)</td>
<td>$i \neq 0, 253, 254, 255$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(255, 0)</td>
<td>$i = 254$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(255, 1)</td>
<td>$i = 255$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(255, 2)</td>
<td>$i = 0, 1$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(129, 129)</td>
<td>$i = 2$</td>
<td>$2^{-16}(1 + 2^{-8})$</td>
</tr>
<tr>
<td>(255, 255)</td>
<td>$i \neq 254$</td>
<td>$2^{-16}(1 - 2^{-8})$</td>
</tr>
<tr>
<td>(0, $i + 1$)</td>
<td>$i \neq 0, 255$</td>
<td>$2^{-16}(1 - 2^{-8})$</td>
</tr>
</tbody>
</table>
Outline

• Background: TLS and RC4
• Weaknesses in RC4
• **Attacks on RC4 in TLS**
• Conclusions and Mitigations
Elements of the Attack

Known biases in pads +

Many cypher texts, different keys ->

Best guess estimates of plain text
Elements of the Attack

Known biases in pads +

Many cypher texts, different keys ->

Best guess estimates of plain text
Elements of the Attack

Known biases in pads +

Many cypher texts, different keys ->

Best guess estimates of plain text
Elements of the Attack

Known biases in pads +

Many cypher texts, different keys ->

Best guess estimates of plain text

cipher text ⊕ byte bias = “vote” for plain text
$2^{20}$ Sessions

[Graph showing recovery rate against byte position]
$2^{30}$ Sessions
Outline

• Background: TLS and RC4
• Weaknesses in RC4
• Attacks on RC4 in TLS

• Conclusions and Mitigations
Threat Level

- Crazy number of cypher texts needed
- Need to force TLS renegotiations (ie online attack)
Mitigations

• RC4-Drop[N], with \( N \in \{256, 768, 3072\ldots\} \)

• Move back to AES-CBC and incorporate BEAST, Lucky13 mitigations

• Upgrade to TLS-1.2 and use AES-GMC