Stream Ciphers: Past, Present and Future

Prof. Bart Preneel
Katholieke Universiteit Leuven, COSIC
http://www.ecrypt.eu.org/

ISCISC 2010
15 September 2010

Outline

• Stream ciphers
• Attacks on stream ciphers
• eSTREAM Project
• Perspectives

Stream ciphers: history

- 1917 one-time pad
- 1930’s Hagelin
- 1940’s Enigma
- 1960’s LFSRs
- 1977 DES
- 1978 Triple-DES
- 1987 FEAL
- 1990 Khufu/Khafre/IDEA
- 1997 SEAL
- 1998 Panama
- 1999 Eli, SNOW
- 1999 Kasumi
- 2000 AES
- 2004 estream
- 2008 FISH, ISAAC, PIKE, Shrinking generator, VEST, WAKE, ...
- 2001 MUGI
- 2006 SNOW 3G
- 2010 ZUC

Stream ciphers: definitions

• No clear definitions but main characteristics can be identified
  - Operate on small words (bits, bytes, 32-bit words)
  - Internal state
  - Simple
• Continuum between stream ciphers and block ciphers

Synchronous Stream Cipher (SSC)

Self-Synchronising Stream Cipher (SSSC)
Specific properties of an SSC

- Cryptographically secure pseudo-random bit generator (PRBG)
- No error-propagation
- Resynchronization mechanism needed
- Particular model for cryptanalysis
  - Key stream independent of data

Advantages of stream ciphers

- Software:
  - Up to five times faster than AES
  - Useful in high throughput applications
- Hardware
  - Use less area than AES
  - < 2 kGate compared to > 3 kGate for AES
  - Higher throughput/area possible
  - Useful in restricted environments: RFID, ...

Moore's Law: computation/storage

- Storage: Gigabyte/s
- Ethernet: speed in Gbps
- Microprocessor performance: Gflops/s

Types of SSC

- LFSR based stream ciphers
  - good randomness properties
  - mathematical theory
  - maximum period \(2^{\ell-1}\)
  - compact in hardware
  - too linear: easy to predict

Types of SSC (2)

- Destroy linearity of LFSRs with
  - Boolean output function
    - E.g. Sober-t
  - additional non-linear memory
    - E.g. summation generator, SNOW 2.0, E0 (Bluetooth)
  - irregular clocking
    - E.g. A5/1 (GSM), shrinking generator, self-shrinking generator

Example: A5/1 (GSM)

- Clock control: registers agreeing with majority are clocked (2 or 3)
Stream ciphers: Past, Present and Future
Bart Preneel

Types of SSC (3)

- Problem with LFSR-based designs:
  - Algebraic attack and fast algebraic attack
    - [Courtois03] and [Courtois-Meier03]
  - ...
- Precluding algebraic attacks
  - irregular clocking
  - nonlinear update functions
    - Rationale: keep nonlinearity in the state, don’t just introduce it in the output

Types of SSC (4)

- Huge variety of design strategies for nonlinear update mechanisms
  - Shift registers with non-linear feedback: Grain, Trivium
  - random shuffles: RC4, PY, HC-128
  - T-functions
  - FCSR (feedback with carry)
- Security model of most of these designs not well understood
  - more research needed
  - need for wild and creative ideas: think out of the box
- Block cipher based techniques:
  - SEAL, Scream, MUGI, Phelix, Salsa,...

RC4: weaknesses

- was frequently used with 40-bit key
  - US export restrictions until Q4/2000
- best known general shortcut attack: $2^{241}$
- weak keys and key setup (shuffle theory)
- some statistical deviations
  - 2nd output byte is biased (solution for this bias: drop first 256 bytes of output)
- problem with resynchronization modes (WEP)
Stream ciphers: Past, Present and Future
Bart Preneel

Overview of the Presentation
- Stream ciphers
- Attacks on stream ciphers
- eSTREAM Project
- Perspectives

Attacks on stream ciphers
- Brute force attacks \((k,v,s)\) size of (key, IV, state)
  - Exhaustive key search
  - Exhaustive state search
  - Time-Memory Tradeoffs
- Shortcut attacks
- Side channel attacks: active and passive

Shortcut attacks
- Statistical attacks:
  - (fast) correlation attacks [Meier-Staffelbach89]
  - linear attacks [Golic94]
  - \(\chi^2\) attacks
- Guess and determine attacks
- Algebraic attacks [Courtois03,Meier89]

Side channel attacks
- Very powerful
- Designs to the edge
- Not surprising that these attacks are very effective in particular against unprotected implementations
- Realistic? Plaintext is available in encrypter and decrypter
- Specific scenarios: RFID cloning, DRM, remote (electromagnetic) attacks
Brute force attacks

- **Traditional key-recovery attack**
  - Time \( T = 2^k \) on average
  - Data \( D = \text{negligible} \)
  - Memory \( M = \text{negligible} \)
  - Precomputation \( P = \text{negligible} \)

- **Table-based key-recovery attack**
  - Time \( T = 1 \) step
  - Data \( D = \text{negligible} \)
  - Memory \( M = 2^k \) k-bit values
  - Precomputation \( P = 2^k \)

- **Trade-off attack**
  - Trade some \( T \) complexity for \( D, M \) and/or \( P \) complexity

Classical TMD trade-offs against any one-way function (including any stream cipher)

- Aim: given one key stream, recover the secret key (\( k \) bits)
- \([\text{Hellman80]}\]
  - \( T = 2^{2k/3}, M = 2^{2k/3}, D=1 \) (\( k \) bits) and \( P = 2^k \)
  - Note that \( T \cdot M = 2^{4k/3} \)

Classical TMD trade-offs against stream ciphers with unkeyed output transformation

Aim: given one key stream, recover \( s \) bits of internal state

- Birthday \([\text{Babbage-Golic}]\)
  - \( T \cdot M = 2^s, D=T \) and \( P=M \)
  - Not very flexible

- Extended birthday \([\text{Biryukov-Shamir00}]\)
  - \( T \cdot M^2 \cdot D^2 = 2^{2s}, T>D^2 \) and \( P=2^s/D \)
  - More flexible trade-off between time, memory and data

Classical TMD trade-offs (2)

- Classical TMD on a stream cipher can be easily prevented by making the state size \( s \) larger than twice the key size \( k \)

  - Note: eSTREAM candidates not vulnerable to classical TMD trade-offs

TMK trade-offs

- **time-memory-key tradeoffs** \([\text{Biham96-02}]\), \([\text{Biryukov+05}]\), \([\text{Hong-Sarkar05}]\)

- **Attack scenario**
  - Attacker intercepts a large number \( D_k \) of short key streams, coming from different keys and same IV
  - Attacker wants to recover only one of those keys
  - Realistic scenario in e.g. broadcasting one message to many users

TMK trade-offs (2)

- Same trade-offs as in classical TMD, but we attack the key directly, instead of the state

- Birthday \([\text{Babbage-Golic}]\)
  - \( T \cdot M = 2^k, T=D_k \) and \( P=M \)

- Extended birthday \([\text{Biryukov-Shamir}]\)
  - \( T \cdot M^2 \cdot D_k^2 = 2^{2k}, T>D_k^2 \) and \( P=2^k/D_k \)

- Impact on the eSTREAM key sizes (128 bit, 80 bit)
TMK trade-offs (3): 128 bit key

Example: $D_k = 2^{40}$, then $T = 2^{20}$, $M = 2^{48}$, $P = 2^{88}$

TMK trade-offs (4): 80 bit key

Example: $D_k = 2^{25}$, then $T = 2^{10}$, $M = 2^{20}$, $P = 2^{55}$

TMK trade-offs (5)
- This can be a problem in some cases
- Solution?
  - TMK trade-offs work because the only entropy we introduce into the state comes from the secret key
  - Solution is therefore simple: add more entropy to the initialization, either by:
    - Using full-entropy IVs [Biryukov+05]
    - Using longer keys [Bernstein06]

TMK trade-offs (6)
- Pros and cons of both approaches

<table>
<thead>
<tr>
<th>Longer keys</th>
<th>Full-entropy IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Secure distribution and storage is needed</td>
<td>- Generation and transmission is necessary</td>
</tr>
<tr>
<td>- Security level differs from key length</td>
<td>- correspondence between key length and security level is maintained</td>
</tr>
<tr>
<td>+ Brute-force attacks are made more expensive</td>
<td>- Security level is maintained</td>
</tr>
<tr>
<td>- Resynchronization may become more delicate</td>
<td>- Only provides protection against trade-off attacks</td>
</tr>
<tr>
<td>(slide attacks)</td>
<td></td>
</tr>
</tbody>
</table>

TMK trade-offs (7)
- How much entropy is needed?
  - Can be easily calculated from formulae
  - Attack scenario 1: $T, M, D, P$ all $< 2^k$
    - Can be avoided by adding entropy equivalent to $k$ of the security level
  - Attack scenario 2: $T, M, D$ all $< 2^k$, but we allow $P > 2^k$
    - Can be avoided by adding entropy equivalent to once the security level

Cube attack [Dinur-Shamir’08]
- exploits low degree equations in stream cipher
- can break certain ciphers which could not be broken before
- ….media hype and controversy
- relation to higher order attacks (Lai) and AIDA (algebraic IV differential attack) (Vielhaber)
- Trivium:
  - key setup can be broken if number of rounds is reduced from 1024 to 753 (Aida) or 767 (cube)
  - attack may be further improved
  - solution: increase number of rounds of key/IV setup to 2048
- Grain: results announced at Crypto rump session (no details yet)
Stream ciphers: Past, Present and Future
Bart Preneel

Outline

• Stream ciphers
• Attacks on stream ciphers
• eSTREAM Project
• Perspectives

ECRYPT basic facts

• European Network of Excellence in Cryptology
• ECRYPT: Feb 2004 – July 2008
• ECRYPT II: Aug 2008 – July 2010
• Total funding: 8.6 M€
• Very limited effort for person-months – mostly networking activities

Cryptographic competitions

• US
  – two calls for block cipher in 70ies, resulting in DES
  – SHA-3 (2007-2012)
• Japan: CRYPTREC (http://www.ipa.go.jp/security)
• Europe
  – eSTREAM (2004-2008)

eSTREAM Project

• Project within ECRYPT (2004-2008)
• Goals:
  – Identify a set of promising stream ciphers
  – Promote a greater understanding of stream cipher design and analysis
• Differences compared to AES/SHA-3 competition
  – eSTREAM is not a standardisation effort
  – We offered the designers more flexibility with regards to “tweaking” because we did not want to miss out on good design ideas because of an easily avoided flaw

eSTREAM timeline

SASC (2004) 04/05 07/06 09/07
SASC (2006) 01/05 01/06 01/07 01/08
CIP 01/08
Phase 1 07/06
Phase 2 01/07
Phase 3
Final Report

NESSIE Stream ciphers

• BMGL* (Ericsson, KTH Stockholm)
• Leviathan (Cisco Systems Inc.)
• LILI-128 (Queensland Univ. of Technology, Univ. of Belgrade)
• SNOW* (Lund Univ.)
• Sobert-t16 (Qualcomm Int.)
• Sobert-t32 (Qualcomm Int.)
Call for Primitives

- eSTREAM solicits stream ciphers for two types of environments
  - PROFILE 1:
    - Stream ciphers for high throughput SW applications
      - 128-bit key, 64 and 128-bit IVs
  - PROFILE 2:
    - Stream ciphers for HW with restricted resources
      - 80-bit key, 32-bit and 64-bit IVs

- Associated authentication mechanisms will also be studied (tag lengths from 32 to 128 bits)

CfP: Evaluation criteria

- Security
- Performance
  - when compared to the AES
  - when compared to other submissions
- Justification and supporting analysis
- Simplicity and flexibility
- Completeness and clarity of submission

CfP: results

- 34 submissions
  - 31 SSC and 2 SSSC
  - 20 countries: Australia, Belgium, France, Sweden, Germany, USA, Canada, Russia,...
  - 10 LFSR, 6 NLFSR, 3 T-functions, 2 SPN
  - several combine multiple ideas
  - 7 offer an authentication method
  - many wild ideas...

Submission results

- 22/34 broken in Phase 1

Phase 2 algorithms (many tweaked)

Phase 3 algorithms

<table>
<thead>
<tr>
<th>SW Phase 3</th>
<th>HW Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CryptMT*</td>
<td>DECIM v2*</td>
</tr>
<tr>
<td>DRAGON</td>
<td>Edon-80*</td>
</tr>
<tr>
<td>HC-128 (-256)</td>
<td>F-FCFSR-H v2*</td>
</tr>
<tr>
<td>LEX</td>
<td>Grain v1</td>
</tr>
<tr>
<td>NLS (encrypt only)</td>
<td>MICRO128 v2</td>
</tr>
<tr>
<td>Rabbit*</td>
<td>MOUSTIQUE*</td>
</tr>
<tr>
<td>Salza20</td>
<td>POMARANCH v3*</td>
</tr>
<tr>
<td>SOSEMANUK</td>
<td>Trivium</td>
</tr>
</tbody>
</table>

* not focus in Phase 2

All algorithms are the most recent version
Phase 3

- No algorithms with authenticated encryption left
- Limited cryptanalytic results
  - EDON-80
  - LEX
  - MOUSTIQUE
  - POMARANCH
- Performance played an important role in final choice

The eSTREAM Portfolio
(http://www.ecrypt.eu.org/stream)

<table>
<thead>
<tr>
<th>Software</th>
<th>Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>HC-128</td>
<td>F-ECFSR-H v2</td>
</tr>
<tr>
<td>Rabbit</td>
<td>Grain v1</td>
</tr>
<tr>
<td>Salsa20/12</td>
<td>MICKEY v2</td>
</tr>
<tr>
<td>Sosemanuk</td>
<td>Trivium</td>
</tr>
</tbody>
</table>

(In alphabetical order) Goal: promote research, not one standard

STVL Tools and Resources

- Stream Cipher SW Performance Framework
  - www.ecrypt.eu.org/stream/perf/

Software performance: Pentium M
(cycles/byte)

<table>
<thead>
<tr>
<th></th>
<th>40 bytes</th>
<th>576 bytes</th>
<th>1500 bytes</th>
<th>IMIX 576 bytes</th>
<th>Long IMIX streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copy</td>
<td>2.72</td>
<td>0.61</td>
<td>0.34</td>
<td>0.66</td>
<td>0.22</td>
</tr>
<tr>
<td>AES (CTR)</td>
<td>22.7</td>
<td>16.2</td>
<td>16.1</td>
<td>16.6</td>
<td>16.0</td>
</tr>
<tr>
<td>SNOW2.0</td>
<td>24.3</td>
<td>5.80</td>
<td>5.28</td>
<td>6.88</td>
<td>4.75</td>
</tr>
<tr>
<td>HC-128</td>
<td>768</td>
<td>56.3</td>
<td>23.8</td>
<td>93.1</td>
<td>16.5</td>
</tr>
<tr>
<td>Rabbit</td>
<td>22.7</td>
<td>4.85</td>
<td>4.46</td>
<td>5.93</td>
<td>9.94</td>
</tr>
<tr>
<td>Salsa20/12</td>
<td>22.1</td>
<td>7.65</td>
<td>7.83</td>
<td>8.71</td>
<td>7.43</td>
</tr>
<tr>
<td>Sosemanuk</td>
<td>36.0</td>
<td>10.1</td>
<td>8.60</td>
<td>11.4</td>
<td>5.60</td>
</tr>
</tbody>
</table>

Software performance: AMD64
(cycles/byte)

<table>
<thead>
<tr>
<th></th>
<th>40 bytes</th>
<th>576 bytes</th>
<th>1500 bytes</th>
<th>IMIX 576 bytes</th>
<th>Long IMIX streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copy</td>
<td>1.36</td>
<td>0.27</td>
<td>0.28</td>
<td>0.35</td>
<td>0.29</td>
</tr>
<tr>
<td>AES (CTR)</td>
<td>18.1</td>
<td>13.4</td>
<td>13.4</td>
<td>13.7</td>
<td>13.4</td>
</tr>
<tr>
<td>SNOW2.0</td>
<td>23.2</td>
<td>5.77</td>
<td>5.34</td>
<td>6.81</td>
<td>4.83</td>
</tr>
<tr>
<td>HC-128</td>
<td>587</td>
<td>43.2</td>
<td>18.4</td>
<td>71.4</td>
<td>2.86</td>
</tr>
<tr>
<td>Rabbit</td>
<td>14.6</td>
<td>5.55</td>
<td>5.25</td>
<td>6.06</td>
<td>4.98</td>
</tr>
<tr>
<td>Salsa20/12</td>
<td>12.1</td>
<td>5.00</td>
<td>5.07</td>
<td>5.52</td>
<td>4.85</td>
</tr>
<tr>
<td>Sosemanuk</td>
<td>25.3</td>
<td>7.20</td>
<td>6.20</td>
<td>8.03</td>
<td>4.07</td>
</tr>
</tbody>
</table>

Software performance (cycles/byte):
Pentium M

July 2010
Software performance (cycles/byte):
AMD Phenom II X6 1055T 2.8 GHz

Software

- HC-128: very fast; large tables so slow setup
- Rabbit: broad performance, oldest algorithm (2003) but no support for large keys and IP encumbered
- SALSA20/12: simple, clean and scalable
- SOSEMANUK: builds on earlier designs, robust security margin
- (SNOW v2): not submitted

Hardware performance (100 kHz)

<table>
<thead>
<tr>
<th></th>
<th>Throughput (Mbps)</th>
<th>Power (µW)</th>
<th>Area (µm²)</th>
<th>Throughput/area (kbps x µm⁻²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>F-FCSR-H v2</td>
<td>0.8</td>
<td>10.6</td>
<td>25006</td>
<td>0.032</td>
</tr>
<tr>
<td>Grain</td>
<td>0.1</td>
<td>3.3</td>
<td>6667</td>
<td>0.015</td>
</tr>
<tr>
<td>Grain (x4)</td>
<td>0.4</td>
<td>4.5</td>
<td>8696</td>
<td>0.046</td>
</tr>
<tr>
<td>Grain (x8)</td>
<td>0.8</td>
<td>6.1</td>
<td>11429</td>
<td>0.070</td>
</tr>
<tr>
<td>Grain (x16)</td>
<td>1.6</td>
<td>9.3</td>
<td>16842</td>
<td>0.095</td>
</tr>
<tr>
<td>MICKEYv2</td>
<td>0.1</td>
<td>7.1</td>
<td>25000</td>
<td>0.004</td>
</tr>
<tr>
<td>Trivium</td>
<td>0.1</td>
<td>5.5</td>
<td>12500</td>
<td>0.008</td>
</tr>
<tr>
<td>Trivium (x4)</td>
<td>0.4</td>
<td>6.0</td>
<td>13793</td>
<td>0.029</td>
</tr>
<tr>
<td>Trivium (x8)</td>
<td>0.8</td>
<td>7.0</td>
<td>15385</td>
<td>0.052</td>
</tr>
<tr>
<td>Trivium (x16)</td>
<td>1.6</td>
<td>7.4</td>
<td>16326</td>
<td>0.068</td>
</tr>
</tbody>
</table>
Hardware

- F-FCSR-H v2: broken
- Grain v1: simple, effective, nice tradeoffs
- Mickey v2: clarity and simplicity; less tradeoffs but perhaps more robust
- Trivium: simple and clear; exceptional hardware performance

Outline

- Stream ciphers
- Attacks on stream ciphers
- eSTREAM Project
  - Perspectives

Future perspectives

- Further work on eSTREAM portfolio
- Lightweight crypto
- Provable secure stream ciphers
- Need authenticated encryption
  - Stream cipher + GCM?
  - Innovative designs: duplex: sponge-based (e.g. Keccak)

AES implementations: efficient/compact

- software
  - 7.6 cycles/byte on Core 2 or 110 Mbyte/s bitsliced [Käsper-Schwabe'09]
- co-processor in Intel Westmere
  - new AES instruction: 0.75 cycles/byte [09-'10]
- hardware
  - fast 43 Gbit/s in 130 nm CMOS ['05]
  - most compact: 3600 gates

Low cost hw: throughput versus area

<table>
<thead>
<tr>
<th>Gate equivalents</th>
<th>Throughput (Kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Present-80 (18)</td>
<td>0</td>
</tr>
<tr>
<td>Present-128 (18)</td>
<td></td>
</tr>
<tr>
<td>HIGHT (25)</td>
<td></td>
</tr>
<tr>
<td>Grain (12)</td>
<td></td>
</tr>
<tr>
<td>TEA (16)</td>
<td></td>
</tr>
<tr>
<td>AES (35)</td>
<td></td>
</tr>
<tr>
<td>CLEFIA (9)</td>
<td></td>
</tr>
<tr>
<td>Enocoro-80 (9)</td>
<td></td>
</tr>
<tr>
<td>mCRYPTON-96 (13)</td>
<td></td>
</tr>
<tr>
<td>Grain (8) (13)</td>
<td></td>
</tr>
<tr>
<td>Trivium (8) (13)</td>
<td></td>
</tr>
<tr>
<td>Trivium (8) (13)</td>
<td></td>
</tr>
<tr>
<td>AES (35)</td>
<td></td>
</tr>
<tr>
<td>CLEFIA (9)</td>
<td></td>
</tr>
<tr>
<td>Grain (8) (13)</td>
<td></td>
</tr>
<tr>
<td>Trivium (8) (13)</td>
<td></td>
</tr>
<tr>
<td>AES (35)</td>
<td></td>
</tr>
</tbody>
</table>

Are block ciphers the future for lightweight crypto?

- Comparisons are unfair
  - Technologies different
  - Security is different: birthday attacks for 1 key
    - block cipher with block length of 64 bits: $2^{32}$
    - stream cipher with 80-bit key and 160-bit state: $2^{80}$
  - All novel designs have very small nonlinear component, so the cost is the memory
    - hardware: its costs between 2 and 8 gates to store 1 bit
    - software: RAM usage is critical factor
      - 256 bytes on low-end 8-bit processor (such as PIC10-16, RS08TM, HC08TM, COP8, 80C51TM)
- Stream ciphers offer better throughput per area
Quad [Berbain-Gilbert-Patarin'06]

- iteration of a randomly chosen multivariate quadratic system
- Key/IV setup – similar requirements to a block cipher
- $k=v=80; s=160$
- 2915 PIV cycles/byte
- Cryptanalysis [Yang-Chen-Bernstein-Chen'07]
  - Conclusion: need at least $k=v=160; s=320$

Authenticated encryption based on sponge/duplex [Daemen+10]

- Building block of SHA-3 candidate Keccak can be used for authenticated encryption with associated data
- More efficient than GCM/CCM
- Not patented
- Pseudo-random bit generator with reseeding
- More details: see paper of Daemen et al. at Aug 2010 SHA-3 conference
Conclusions

- stream ciphers are not dead
  - even if AES gets faster, they offer interesting trade-offs not reachable by AES
- open issues
  - authenticated encryption
  - stream ciphers based on permutations and state permutations
  - lightweight crypto
  - self-synchronizing stream ciphers
- many designs still very new and close to the edge
- standardization for the coming years