Spritz—a spongy RC4-like stream cipher and hash function

Ronald L. Rivest\textsuperscript{1}  Jacob C. N. Schuldt\textsuperscript{2}

\textsuperscript{1}Vannevar Bush Professor of EECS
MIT CSAIL
Cambridge, MA 02139
rivest@mit.edu

\textsuperscript{2}Information Security Group
Royal Holloway, University of London
jacob.schuldt@rhul.ac.uk

CRYPTO rump session
August 19, 2014
Widely used (50% of all TLS connections).
Simple, fast.

(Wikipedia)
RC4 attacks

RC4 has numerous vulnerabilities and “soft spots” [see paper for citations]:

- Key-dependent biases of initial output
- Key collisions (producing same internal state)
- Key recovery possible from known internal state
- Related-key attacks (WEP)
- State recovery from known output (feasible?)
- Output biases; distinguishers
We started design after CRYPTO 2013.

Principles:
- Drop-in replacement for RC4
- Retain “RC4 style” (e.g. state is a few registers plus a permutation $S$ of $\{0, 1, \ldots, N - 1\}$)
- Minimize statistical vulnerabilities
- Redo key-setup entirely

Automatically examined many thousands of candidates; filtered by syntactic, statistical, and cryptographic criteria. Approximately 5 “core-months” of CPU time used!
Code comparison (Output routines)

Winner is #4933 (postfix for \(i, j, k, z\)):

\[
\text{i1+, kjiS+S+, ikjS++, jikz+S+S+S}
\]

<table>
<thead>
<tr>
<th>RC4()</th>
<th>SPRITZ()</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (i = i + 1)</td>
<td>1 (i = i + 1)</td>
</tr>
<tr>
<td>2 (j = j + S[i])</td>
<td>2 (j = k + S[j + S[i]])</td>
</tr>
<tr>
<td>3 Swap((S[i], S[j]))</td>
<td>3 (k = i + k + S[j])</td>
</tr>
<tr>
<td>4 (z = S[S[i] + S[j]])</td>
<td>4 Swap((S[i], S[j]))</td>
</tr>
<tr>
<td>5 return (z)</td>
<td>5 (z = S[j + S[i + S[z + k]]])</td>
</tr>
<tr>
<td></td>
<td>6 return (z)</td>
</tr>
</tbody>
</table>

- About 50% longer
- Uses new register \(k\) as well RC4 registers \(i, j\); output register \(z\) also used in feedback.
Spritz statistical biases are much fainter than for RC4.
We estimate the biases for $N = 256$ by extrapolating from Spritz with $N = 16, 24, 32$.
For $N = 256$:
- Can distinguish RC4-256 from random with $2^{41}$ samples.
- Our tests suggest that $2^{81}$ samples are required to distinguish SPRITZ-256 from random.
But wait, there’s more! SPRITZ is spongy!

- SPRITZ is also a (modified) sponge function, and usable as a hash function:

1. **INITIALIZE**\textsc{STATE}(N)
2. \textsc{ABSORB}("abc") — ACCEPT INPUT PIECEMEAL.
3. \textsc{ABSORB}("def")
4. \textsc{SQUEEZE}(32) — OUTPUT 32 BYTE HASH.
5. \textsc{ABSORB}("ghi") — KEEP GOING...
6. \textsc{SQUEEZE}(1000)

- Large state space (like KECCAK), but also has built-in protection against inference of key from knowledge of internal state (which KECCAK does not).
- (But very much slower than Keccak...)
Our paper on SPRITZ can be found on my web site:
people.csail.mit.edu/rivest/pubs.html#RS14

More security review needed; comments and analysis appreciated!