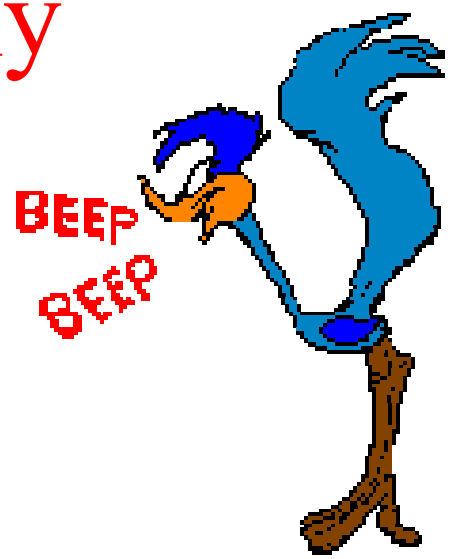
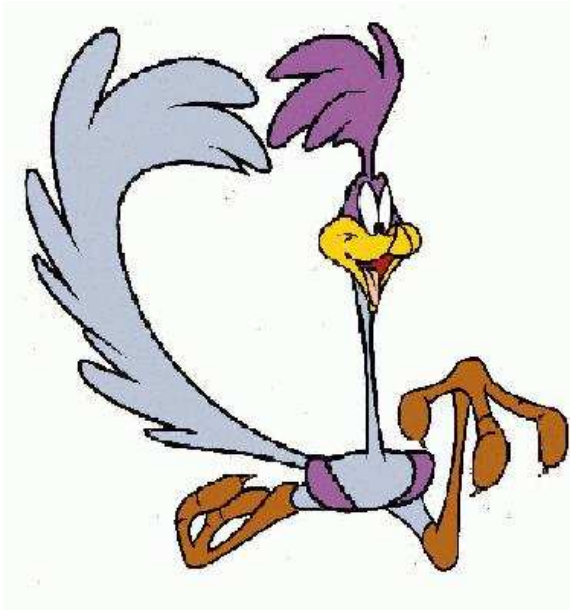
Real Time Cryptography



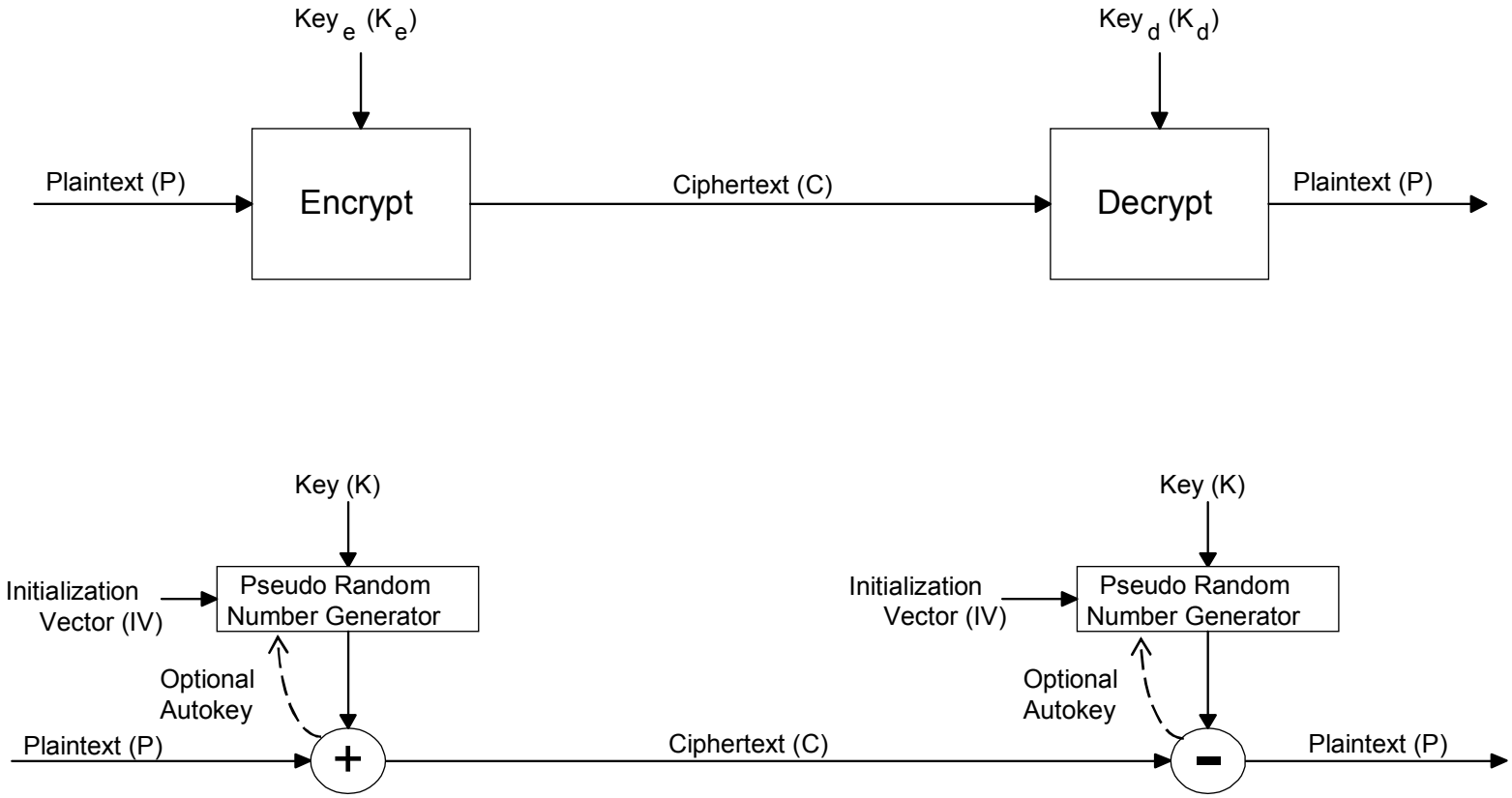
- **The application area**
  - Cryptography optimized for embedded, real-time, control systems
- **The development**
  - A new algorithm, called BeepBeep, overcomes the problems with using existing cryptography for real time systems
- **Contact:** Kevin Driscoll Kevin.Driscoll@Honeywell.com
  612-951-7263 (phone) 612-951-7438 (FAX)

Grid Security



- "need at least 1 Gbps encryption"
- BeepBeep can do that ...
  … in *software* on a 1 GHz Pentium
- No encryption hardware
  - Cheaper
  - More flexible and easier to manage
  - Allows ad hoc grids (i.e., SETI@home model)
- If no physical security at nodes
  - Must assume some nodes will be compromised
    - Portion of data and algorithm will exposed
    - Node's keys will be exposed
      - Need unique key(s) for each node
      - Need crypto algorithm with good key agility

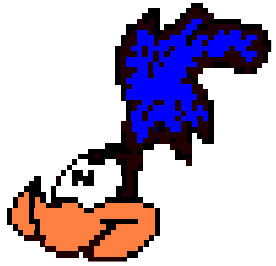
Encryption Basics



Problems with Using Existing Software Cryptography for Embedded Real-Time Systems

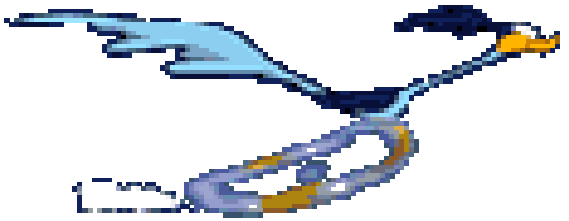
- **Is relatively slow, particularly on start-up**
  - Messages (and sessions) are small, less text to amortize start-up cost
  - Latency (lag) is more important than throughput
  - Only worst case timing counts, average is unimportant
    - One missed deadline is not helped by finishing early at all other times
  - Systems typically use repeating execution time slots of fixed size
  - Central control changes key for each message (high key agility)
- **Uses too much data memory (cache thrashing)**
  - Real time systems are multitasking with many context switches / sec
  - Must assume cache is flushed, S-box accesses are mostly misses
- **Consumes additional communication bandwidth**
  - Ciphertext must be no bigger than plaintext
- **Uses separate secrecy and integrity algorithms (or modes)**
  - Makes execution even slower
  - Prevents “lump in the cable” retrofits
- **Most real-time cryptography will be retrofits, which exacerbates the above problems**

Benefits (vs AES, on Pentium)



- **About 2 times faster for very large messages**
- **About 40 times faster for small messages**
- **About half the memory size**
- **25 to 200 times faster than 3DES**
- **Includes integrity with secrecy (increases the above ratios)**
  - Allows “lump in the cable” (or “dongle”) implementations (with possible sub-bit-time latency)
- **Several thousand times faster and smaller than public key**
- **1:1 byte replacement (to fit existing message sizes)**
  - Can eliminate need for the addition of an explicit IV
  - Can incorporate existing CRC or checksum into integrity
- **Optimized for CPUs typically found in embedded, real time, control, and communication systems**

Achieving Speed



- **Use an efficient stream cipher**
- **State stays in CPU registers (no RAM used)**
- **Ignore or circumvent conventional wisdom fears**
  - Feedback shift registers are slow in software
    - Invention to improve speed by almost 100 times
  - Multiply is slow
    - Becoming faster (from 42 clocks to 1//4 clocks)
    - Invention to use multiply in a powerful new way
  - Conditional jumps are slow on pipelined CPUs
    - Use multiplexor logic instead of conditional jump
      - Instead of: `if C then Z = A else Z = B`
      - Use this: `Z = ((A xor B) and C) xor B`
    - Use unrolled loop to eliminate other jumps
- **Speed on Pentium is better than 1 bit per clock**
  - Actual speed is 1.19 vs theoretical 1.83 bits per clock

Simple and Small

- BeepBeep's executable code
  - One page of C code
    (half of which is declarations and comments)
  - Pentium* without explicit IV 419 bytes
  - Pentium* with explicit IV 484 bytes
  - Pentium* main loop 185 bytes
- BeepBeep's data memory
  - Pentium* MMX (data stays in registers) 0 bytes

* with MMX registers

Minimizing Message Size

- **No block padding (BeepBeep isn't a block cipher)**
- **Minimize or eliminate Initialization Vector (IV)**
  - Use existing data for IV (e.g. unencrypted header fields)
  - Use explicit or implicit message IDs (e.g. time / sequence) (most real time systems use such IDs)
  - Use Block IV Mode to eliminate IVs (see next slide)
  - Eliminate some IVs by chaining messages together
    - Can be used with reliable message delivery
    - Crypto-state is carried over between messages
- **Use existing CRC or other check data for integrity** (may need to add bits if existing check bits aren't enough)

Security

Basis: 127 bit Linear Feedback Shift Register (LFSR)

Benefits: Good statistics, period that won't repeat

Old Attack: Berlekamp-Massey

Old Fixes: clock control, nonlinear filter, nonlinear combination of multiple LFSRs

Newer Attacks: embedding, probabilistic correlation, linear consistency, best affine approximation, …

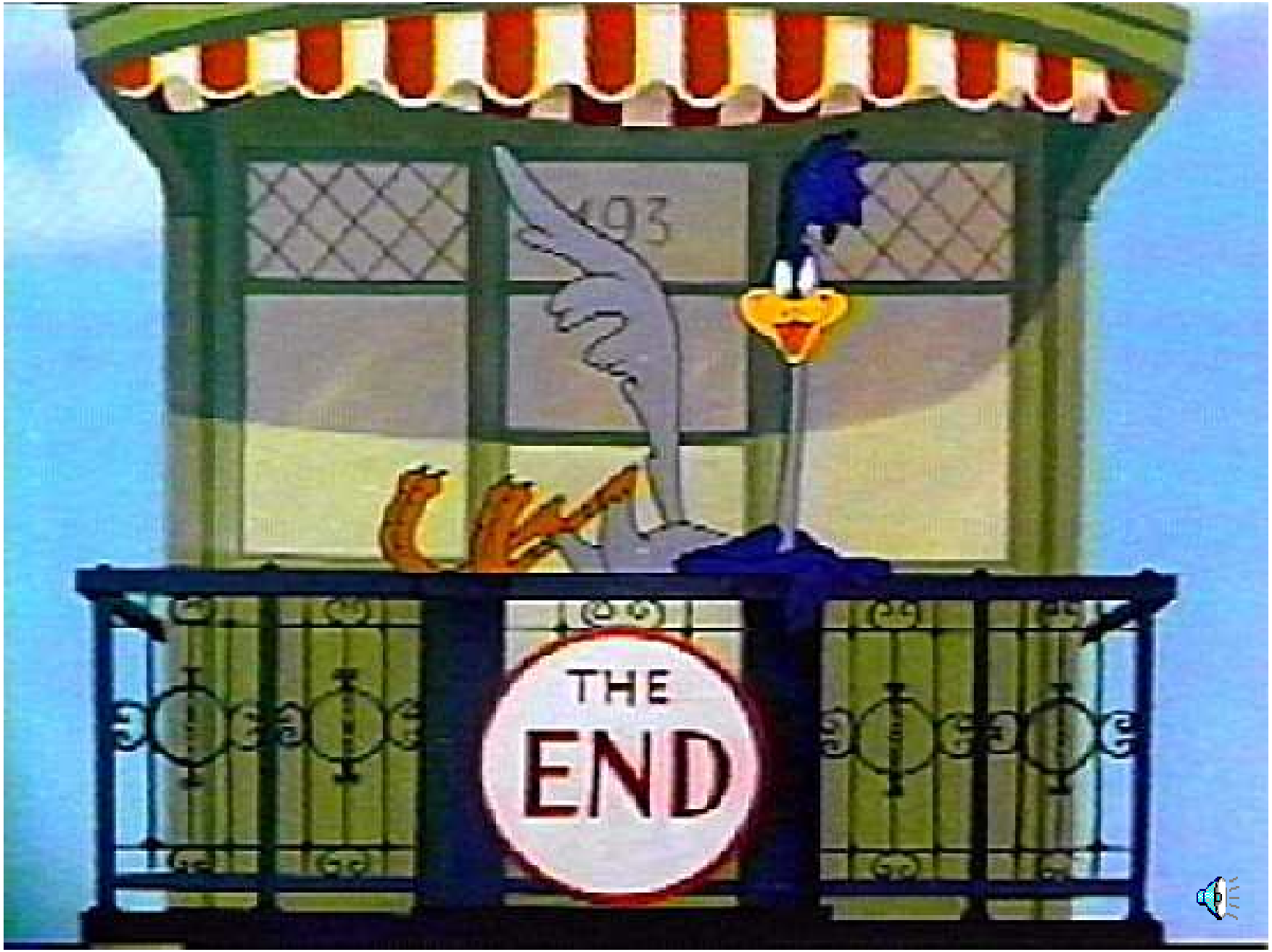
Fixes: use both clock control and nonlinear filter with state, two-stage combiner, and 5 different algebras

Security (continued)

- **Non-linear filter**
  - 32-bit ones complement addition and 32-bit multiply provide 128th order non-linearity
- **Integrity provided by two mechanisms**
  - Two-stage combiner's non-associative operations
    - Ciphertext $=$ (Plaintext + Key_1) XOR Key_2
    - Ciphertext $\neq$ Plaintext + (Key_1 XOR Key_2)
    - Ciphertext $\neq$ Plaintext + (Key_x) [for any possible key]
      - Can't directly recover running key, even with known plaintext
  - Plaintext-based autokey feeds back into the non-linear filter's state and into the clock control's state
    - Propagates any text changes through to the end of message
    - Real-time control system messages usually end with check data that can be used to detect tampering

Postulated Uses

- **Aviation**
  - Encryption of ACARS radio traffic
  - Constraints: bandwidth, real-time multitasking
    - Also memory and execution time for retro-fit applications
  - This application has been cleared for export by BXA
- **Home automation and security** (see Oct 2001 IEEE Computer)
  - Secrecy, integrity, authentication, and key management
  - Between central site and residences
  - Constraints: memory, bandwidth, small (8/16 bit) CPU
    - No other algorithm could meet memory constraints when all four security services were included
      - Limit: 1638 bytes Flash ROM, 50 bytes RAM
      - Used: 1628 28
  - This application has been cleared for export by BXA
- **Commercial buildings and industrial controls**


193
THE
END