Entropy Lessons Learned

Edward Morris

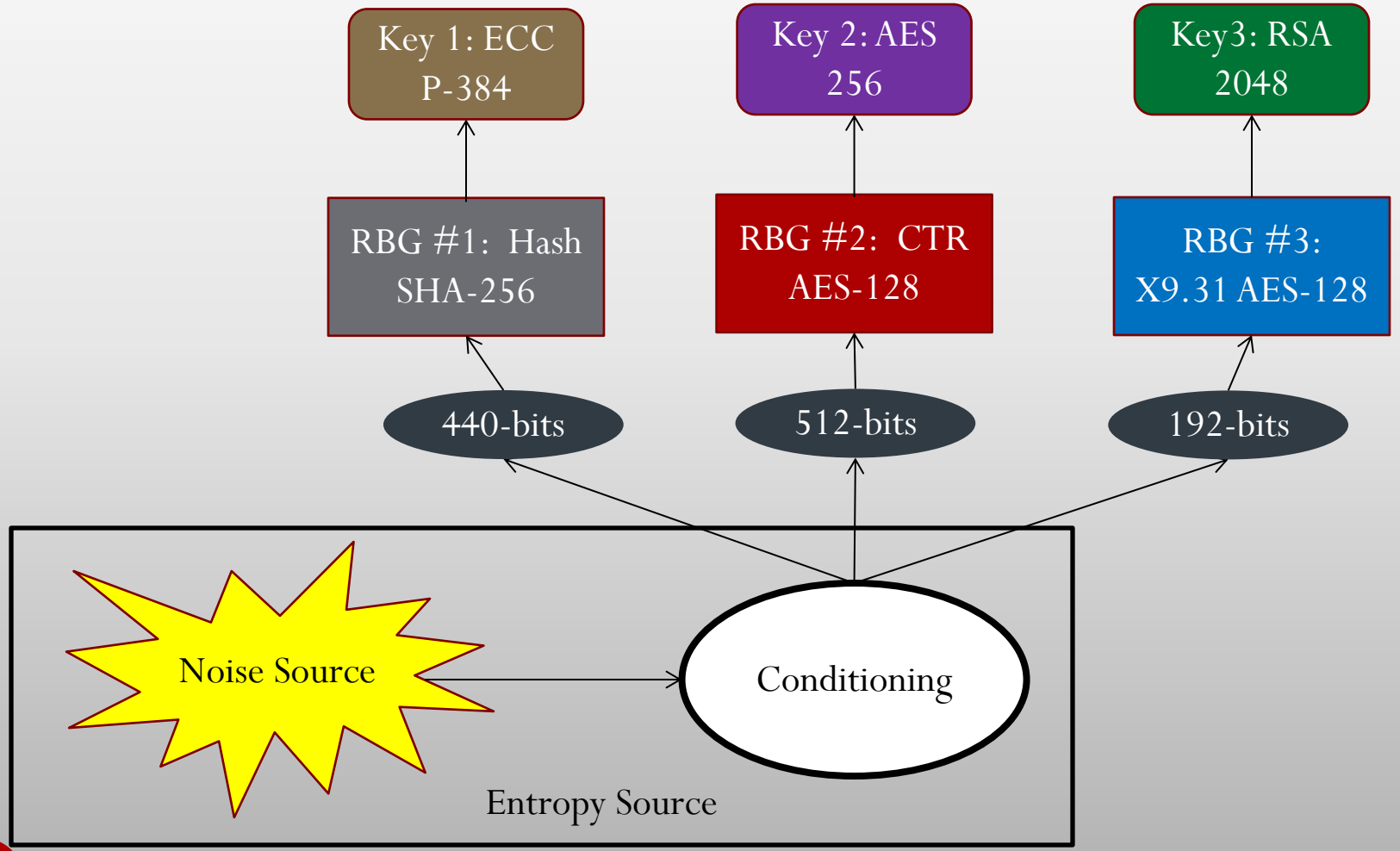
April 15, 2015

Gossamer Laboratories

Reference

- **NIAP**
  - Entropy Annex
  - Entropy Documentation and Assessment Clarification
  - Entropy Documentation Assessment Worksheet
  - Entropy FAQ
  - Entropy Process
- **NIST**
  - SP 800-90A
  - SP 800-90B Draft
  - Python90b

Diagram (Entropy Architecture)



Noise Sources & Conditioning

- **Source type (hardware or software)**
  - Ring oscillators, voltage oscillators
  - Timing variations in high-precision clock
- **Obtain raw entropy before any whitening**
  - Unbiasing (Von Neumann)
  - Condensing (XORing, folding)
  - Pool "stirring" (Hashing)
- **Assess min-entropy of raw entropy**
  - NIST SP800-90
  - IAD tool

RBG Strengths

Standard	RBG	Bits of security	Seed Size
X9.31	AES-128	128	256
800-90A	AES-128 CTR	128	192
X9.31	AES-256	256	384
800-90A	AES-256 CTR	256	384
800-90A	SHA-256 Hash	256	440
800-90A	SHA-256 HMAC	256	440

Key Strengths

Bits of security	Symmetric key algs	FFC (e.g, DSA, DH)	IFC (e.g, RSA)	ECC (e.g., ECDSA)
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

Seeding of RBGs

- **Seed RBGs with sufficient entropy**
  - Use either security strength of the RBG or
  - Use the maximum key size generated by the RBG (ensure max key size < RBG sec strength).
  - Multiply seed size by post-conditioned min-entropy
- **Be wary of older ANSI X9.31 RBGs**
  - Will self-destruct in 2016
  - IAD cryptanalysis states seed and seed key entropy not additive
- **Be mindful of pseudo-keys**
  - Many protocols use salts, nonces, or random values to derive/generate keying matieral.

Lessons Learned (1 of 3)

- **Vendors with hardware sources typically better**
  - Security designs in silicon requires more planning
  - Systems-level vendors need more coaching
  - Vendors using third-party entropy sources generally less knowledgeable
- **Know your common implementations**
  - Linux Kernel Random Number Generator (/dev/random)
  - Microsoft Windows CryptGenRandom
  - RedHat Linux
  - Sun/Oracle Java JVM

Lessons Learned(2 of 3)

- **Allow sufficient time for Review**
  - Frontload all such discussions
  - 3-way NDA roulette
  - Language and time zone issues
- **Distributed (lack of?) knowledge**
  - Existing information often only a partial picture or misses the mark (AIS-20/31, Shannon entropy, etc.)
  - "it's possible" versus "this is what's done"
- **Nailing down the details**
  - Proprietary cloak of obfuscation (force a conf. call for details)
  - Lack of documentation and proper nomenclature
  - Conference calls and the need for written documentation
  - Request an entropy architectural diagram

Lessons Learned(3 of 3)

- **Engineering a solution**
  - If consulting, know the common pitfalls and work around's
  - Advise on risks and preemptively suggest risk mitigation strategies
  - Ask for updated documentation to track progress
- **Assembling a coherent report**
  - Starting fresh versus amending an existing report
  - Less is more (LKRNG source, 800-90A DRBG details, etc.)
  - Vendor review (segregating proprietary information)
- **Shepherding the report home**
  - Be ready to respond quickly to entropy review comments
  - Minimize vendor delays when responding

Questions?

Contacts:

- Ed Morris
  - EdMorris@gossamersec.com

www.gossamersec.com
www.facebook.com/gossamersec
@gossamersec