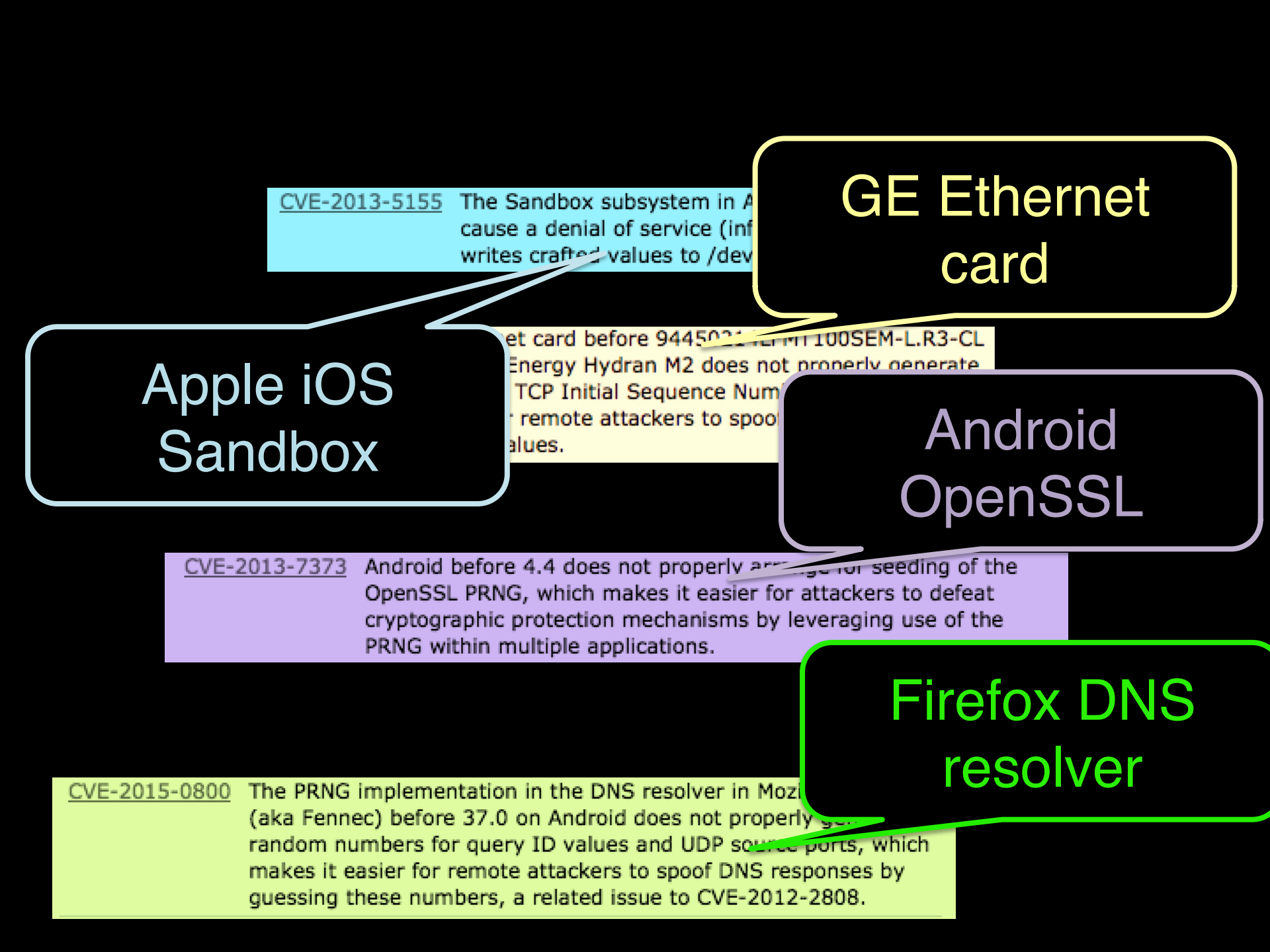
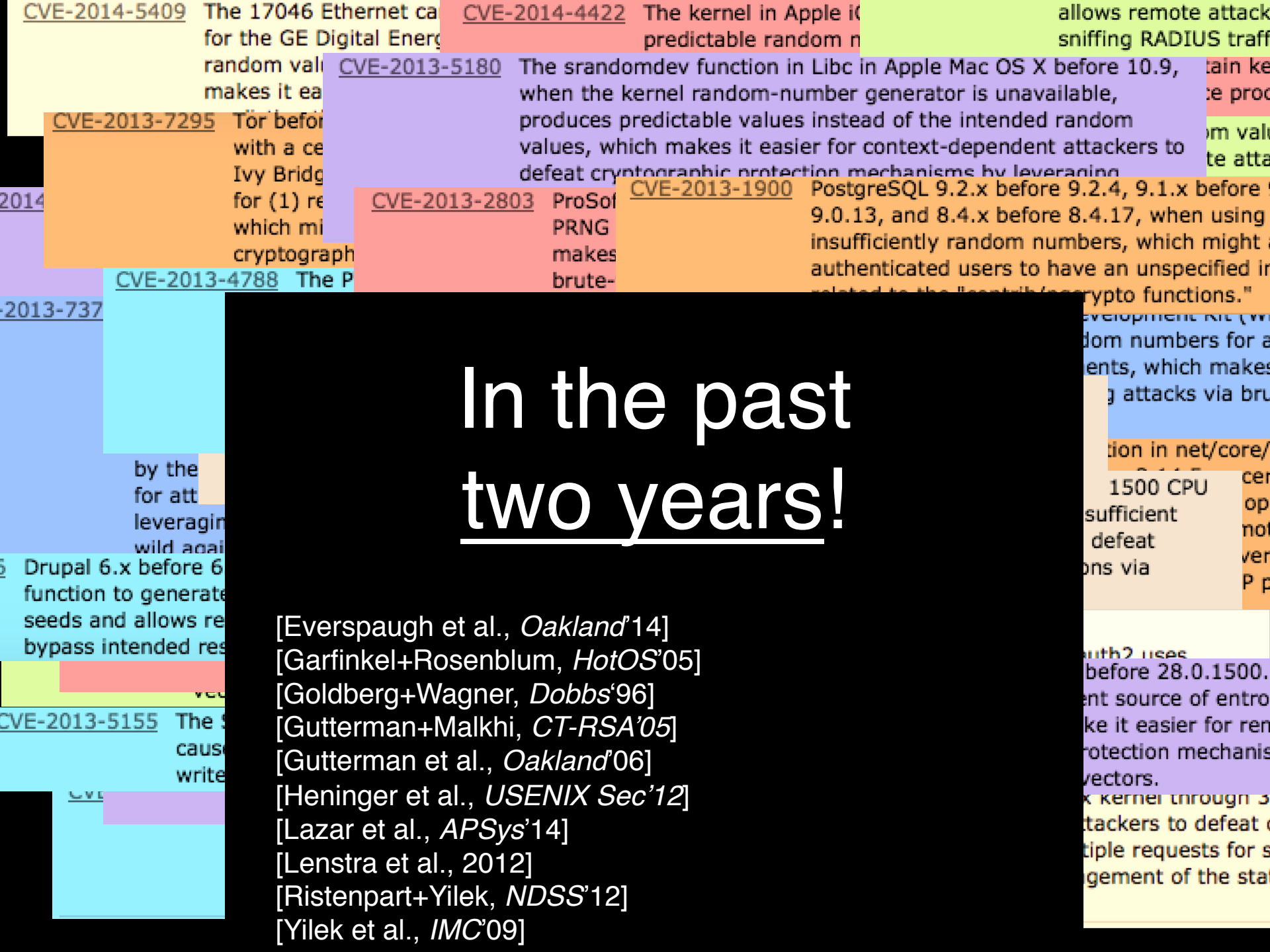
Recommendations for Randomness in the Operating System

Henry Corrigan-Gibbs and Suman Jana
Stanford University

HotOS XV – 20 May 2015


The New York Times
Flaw Found in an Online Encryption Method
By JOHN MARKOFF FEB. 14, 2012
LibreSSL's PRNG is
[Update: LibreSSL
Debian Security
DSA-1571
random number generator
Android random number flaw implicated in Bitcoin thefts


CVE-2013-5155 The Sandbox subsystem in A
cause a denial of service (inf
writes craft values to /dev
GE Ethernet card
et card before 9445021 MT100SEM-L.R3-CL
Energy Hydran M2 does not properly generate
TCP Initial Sequence Num
remote attackers to spoo
alues.
Apple iOS Sandbox
Android OpenSSL
CVE-2013-7373 Android before 4.4 does not properly ar seeding of the OpenSSL PRNG, which makes it easier for attackers to defeat cryptographic protection mechanisms by leveraging use of the PRNG within multiple applications.
Firefox DNS resolver
CVE-2015-0800 The PRNG implementation in the DNS resolver in Moz
(aka Fennec) before 37.0 on Android does not properly
random numbers for query ID values and UDP source ports, which makes it easier for remote attackers to spoof DNS responses by guessing these numbers, a related issue to CVE-2012-2808.


CVE-2014-5409
CVE-2014-4422
CVE-2013-5180
The srandomdev function in Libc in Apple Mac OS X before 10.9, when the kernel random-number generator is unavailable, produces predictable values instead of the intended random values, which makes it easier for context-dependent attackers to
CVE-2013-7295
CVE-2013-2803
CVE-2013-1900
PostgreSQL 9.2.x before 9.2.4, 9.1.x before
9.0.13, and 8.4.x before 8.4.17, when using
insufficiently random numbers, which might
authenticated users to have an unspecified
CVE-2013-4788
In the past
two years!
[Everspaugh et al., *Oakland*'14]
[Garfinkel+Rosenblum, *HotOS*'05]
[Goldberg+Wagner, *Dobbs*'96]
[Gutterman+Malkhi, *CT-RSA'05*]
[Gutterman et al., *Oakland*'06]
[Heninger et al., *USENIX Sec'12*]
[Lazar et al., *APSys*'14]
[Lenstra et al., 2012]
[Ristenpart+Yilek, *NDSS*'12]
[Yilek et al., *IMC*'09]
CVE-2013-5155

Why so many bugs?

Bad news: OS is a big part of the problem.

Randomness subsystems have:

- Buggy design
- Error-prone APIs
- Misleading documentation

} See paper

Good news: The OS can be part of the solution!

What is entropy?

Password has *k* bits of [guessing] entropy w.r.t. an adversary *A*

It takes *A* around 2^k guesses to guess your password

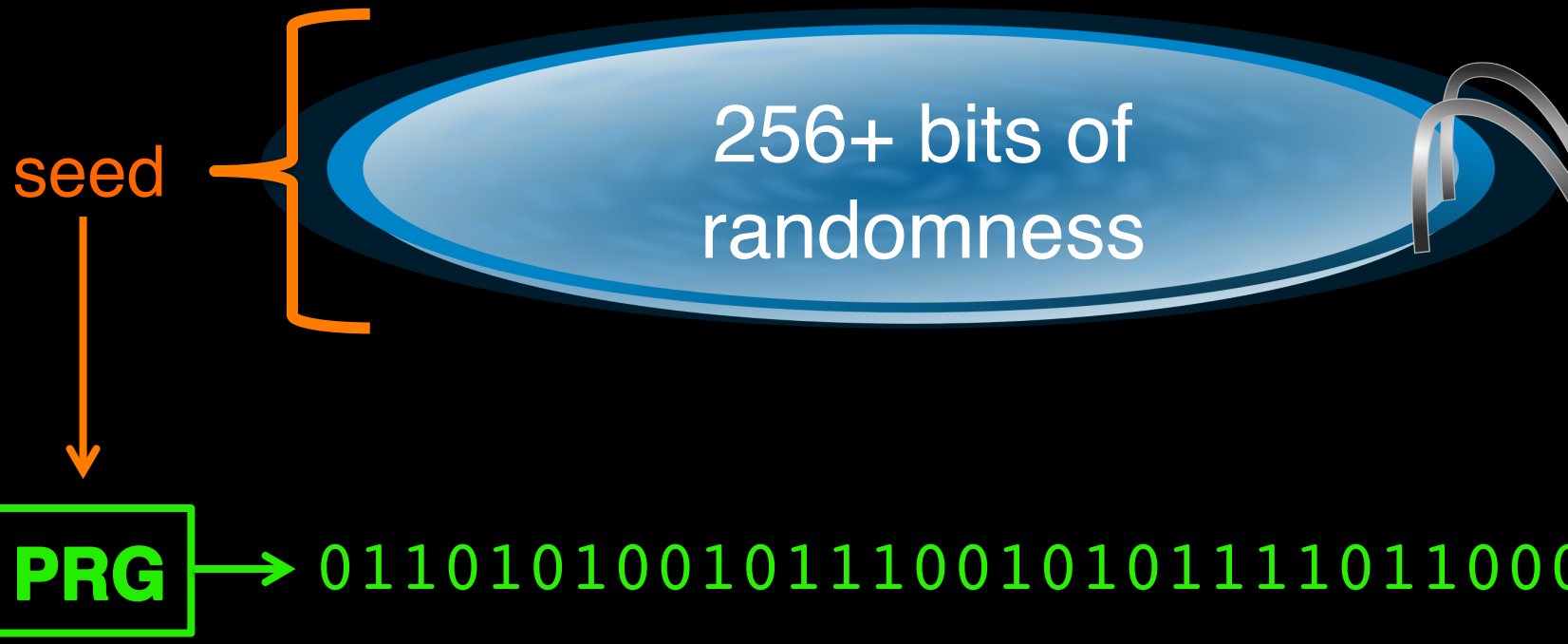
Why do we need it?

- Cryptographic secrets
- ASLR
- DNS source ports
- Password salts
- Etc.

1000100

Application reads
/dev/random

Once the OS has accumulated enough entropy, it will never “run out” of entropy



Great! But what should the OS do before it has 256 bits in the pool?

After first boot…

State of the Art:

Entropy Estimation

State of the Art

0111

"0111"

27 bits

Ye Olde Entropy Estimator

"2.3 bits"

State of the Art

1000

“1000”

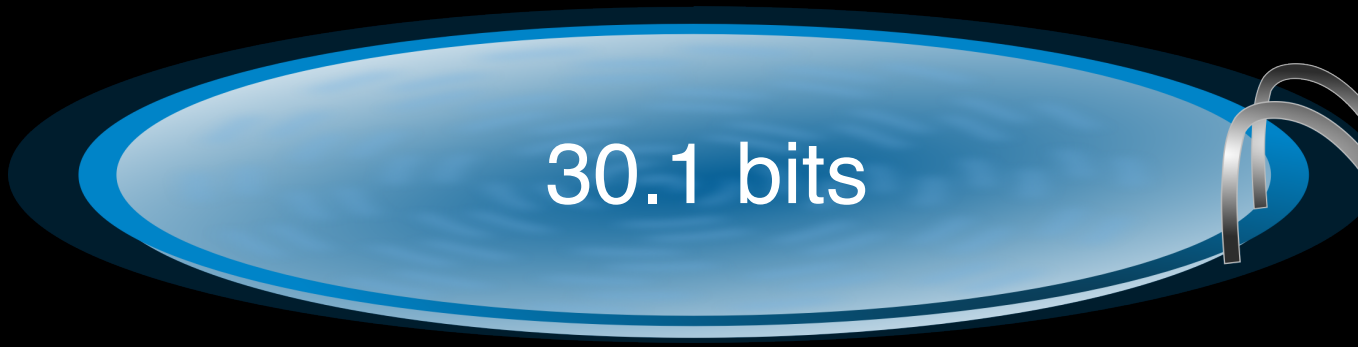
29.3 bits

Ye Olde Entropy Estimator

“0.8 bits”

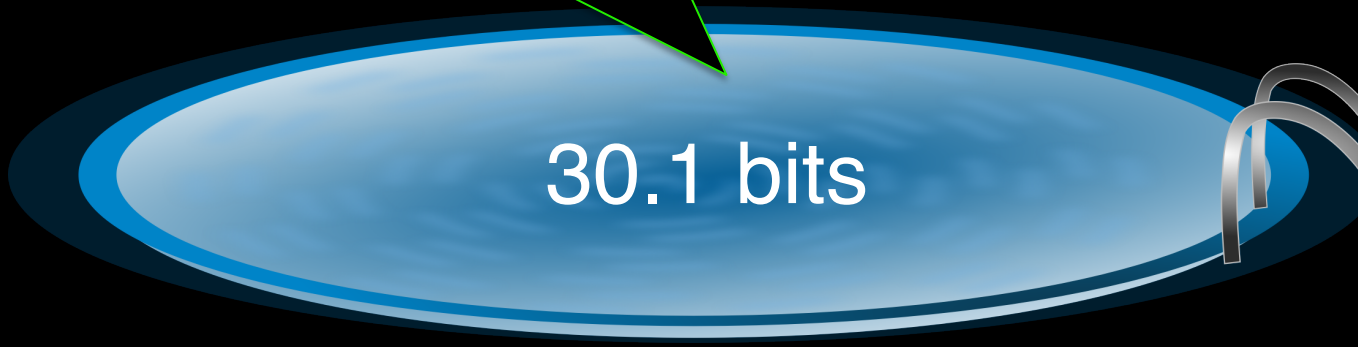
State of the Art

Ye Olde Entropy Estimator



State of the Art

Block `/dev/random` until pool has 256+ bits



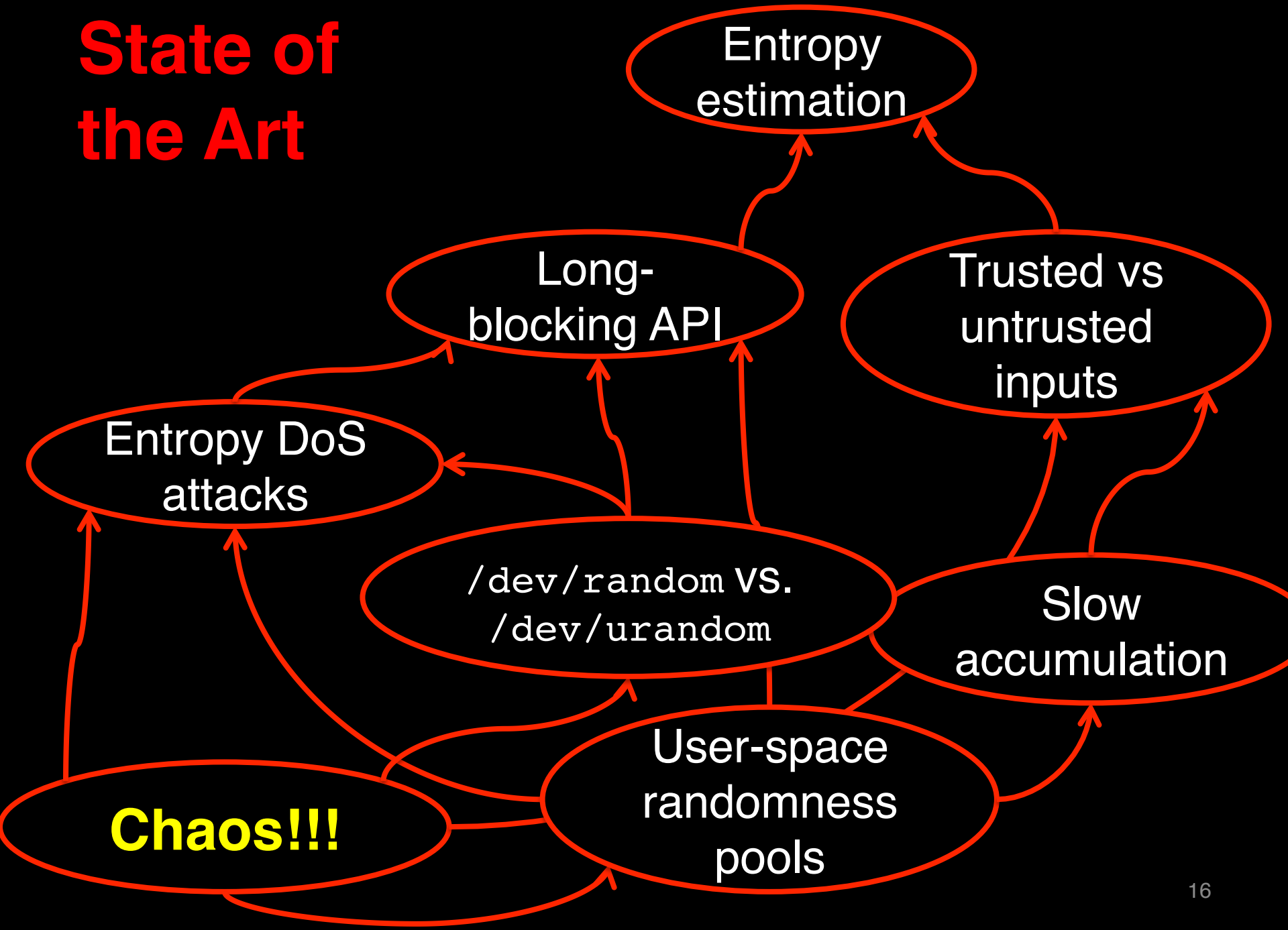
Ye Olde Entropy Estimator

Entropy is a function of **adversary's** knowledge

Estimate could be: 256 bits
Reality could be: **0 bits**

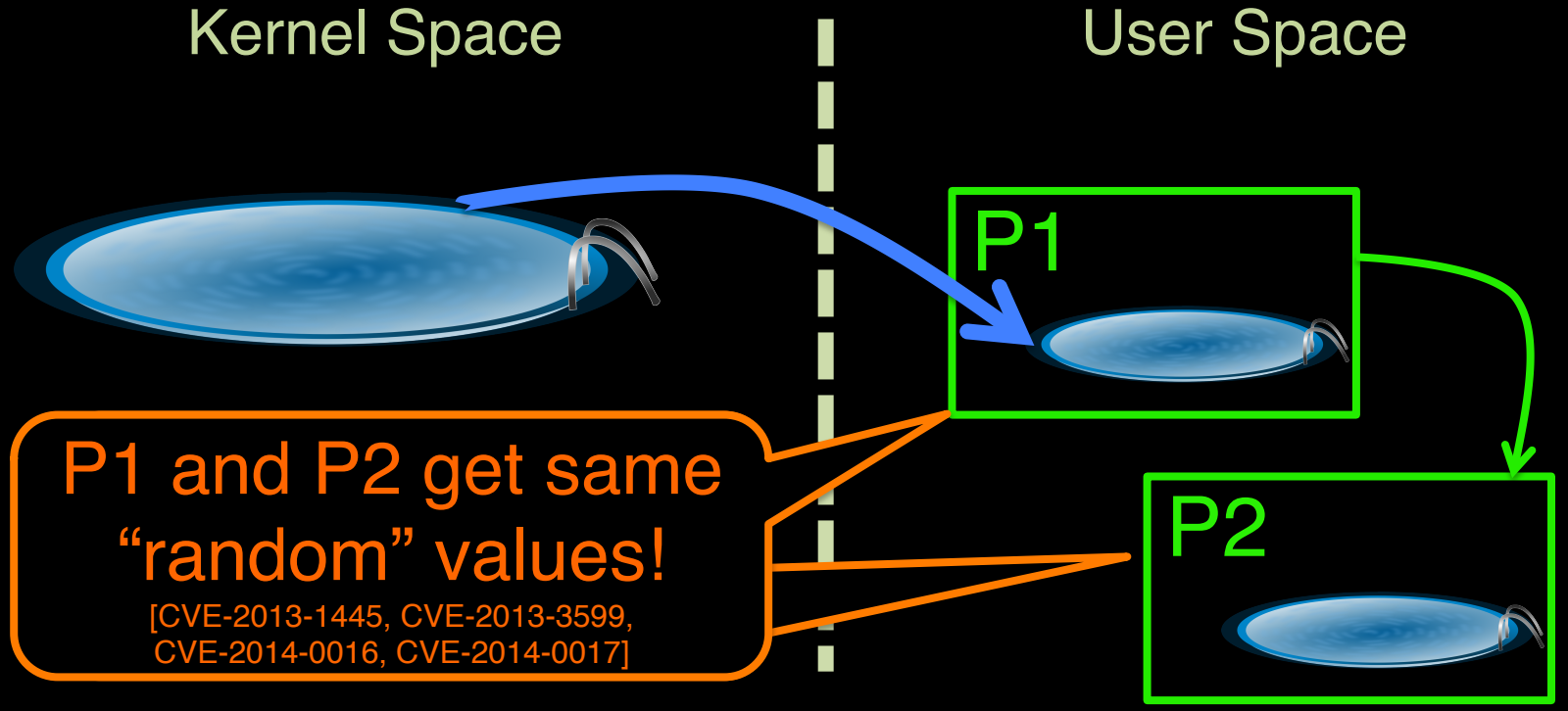
[Barak-Halevi CCS'05]
[Dodis et al. CCS'13]
[Kelsey et al., SAC'00]

State of the Art



One Consequence: User-space Pools

Reading many bytes from `/dev/random` can drive *down* entropy estimate and starve other processes



Our Proposal

Our Proposal

Option 1: Strong Assumptions

- Assume low-order bit of "`RDRAND`" has one bit of entropy
- **Easy!** Just gather 256 samples, use them to seed a PRG

What happens if your assumption is wrong?

Our Proposal

Option 2: “Best-effort” entropy accumulation

- Never estimate
- Never block
  [Barak-Halevi CCS’05] [Dodis et al. CCS’13] [Kelsey et al., SAC‘00]
- Any process can write into OS pool
- Per-process pools

[See paper for details]

→ Honest process’ pools will *eventually* accumulate entropy

Conclusions

- Popular OSes make using randomness more difficult than it needs to be
- Entropy estimation is at the heart of the problem

Our Proposal

- “Best effort” randomness
- Never estimate, never block