eBACS:
ECRYPT Benchmarking
of Cryptographic Systems

http://bench.cr.yp.to

D. J. Bernstein
University of Illinois at Chicago

Joint work with:
Tanja Lange
Technische Universiteit Eindhoven

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

eBACS:
ECRYPT Benchmarking
of Cryptographic Systems

http://bench.cr.yp.to

D. J. Bernstein
University of Illinois at Chicago

Joint work with:
Tanja Lange
Technische Universiteit Eindhoven

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

eBACS:
ECRYPT Benchmarking
of Cryptographic Systems

http://bench.cr.yp.to

D. J. Bernstein
University of Illinois at Chicago

Joint work with:
Tanja Lange
Technische Universiteit Eindhoven

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

Maybe more measurements:
"Oops, lots of overhead
in hashing 256 bytes.
I'll try 4096 bytes."

:

PT Benchmarking

ptographic Systems

//bench.cr.yp.to

Bernstein

sity of Illinois at Chicago

work with:

Lange

sche Universiteit Eindhoven

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

Maybe more measurements:
"Oops, lots of overhead
in hashing 256 bytes.
I'll try 4096 bytes."

Implem
favorite
Adds "
to impl
repeati

Summa
Crypto
is the b
and the

marking
Systems

cr.yp.to

ois at Chicago

ersiteit Eindhoven

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

Maybe more measurements:
"Oops, lots of overhead
in hashing 256 bytes.
I'll try 4096 bytes."

Implementor runs
favorite hardware
Adds "Results" s
to implementatio
repeating what t

Summary:
Cryptographic im
is the benchmark
and the benchma

cago

dhoven

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

Maybe more measurements:
"Oops, lots of overhead
in hashing 256 bytes.
I'll try 4096 bytes."

Implementor runs tool on h
favorite hardware (or emula
Adds "Results" section
to implementation paper
repeating what the tool say

Summary:
Cryptographic implementor
is the benchmark implemen
and the benchmark operate

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

Maybe more measurements:
"Oops, lots of overhead
in hashing 256 bytes.
I'll try 4096 bytes."

Implementor runs tool on his
favorite hardware (or emulator).
Adds "Results" section
to implementation paper
repeating what the tool says.

Summary:
Cryptographic implementor
is the benchmark implementor
and the benchmark operator.

"I've finally finished
my Skein implementation!
Hmmm, how fast is it?"

Traditional answer:
"I'll write a timing tool!
I'll check the clock,
10000× hash 256 bytes,
check the clock again,
subtract, divide by 10000."

Maybe more measurements:
"Oops, lots of overhead
in hashing 256 bytes.
I'll try 4096 bytes."

Implementor runs tool on his
favorite hardware (or emulator).
Adds "Results" section
to implementation paper
repeating what the tool says.

Summary:
Cryptographic implementor
is the benchmark implementor
and the benchmark operator.

This pattern repeats for
every cryptographic implementor.
Hundreds (thousands?) of
separate ad-hoc timing tools
run on various hardware.

nally finished
ein implementation!
n, how fast is it?"

onal answer:
rite a timing tool!
ck the clock,
× hash 256 bytes,
the clock again,
ct, divide by 10000."

more measurements:
lots of overhead
hing 256 bytes.
4096 bytes."

Implementor runs tool on his
favorite hardware (or emulator).
Adds "Results" section
to implementation paper
repeating what the tool says.

Summary:
Cryptographic implementor
is the benchmark implementor
and the benchmark operator.

This pattern repeats for
every cryptographic implementor.
Hundreds (thousands?) of
separate ad-hoc timing tools
run on various hardware.

Europe
NESSI
ECRYP
ECRYP

NESSI
tuned
of man
all supp
wrote a
ran the

Many s
e.g., 24
for 128

ned
nentation!
t is it?"

er:
ng tool!
ck,
6 bytes,
again,
by 10000."

asurements:
verhead
ytes.
s."

Implementor runs tool on his
favorite hardware (or emulator).
Adds "Results" section
to implementation paper
repeating what the tool says.

Summary:
Cryptographic implementor
is the benchmark implementor
and the benchmark operator.

This pattern repeats for
every cryptographic implementor.
Hundreds (thousands?) of
separate ad-hoc timing tools
run on various hardware.

European Union
NESSIE project (
ECRYPT I netwo
ECRYPT II netw

NESSIE's perform
tuned C impleme
of many cryptogr
all supporting the
wrote a benchma
ran the toolkit on

Many specific pe
e.g., 24 cycles/by
for 128-bit AES

Implementor runs tool on his
favorite hardware (or emulator).
Adds "Results" section
to implementation paper
repeating what the tool says.

Summary:
Cryptographic implementor
is the benchmark implementor
and the benchmark operator.

This pattern repeats for
every cryptographic implementor.
Hundreds (thousands?) of
separate ad-hoc timing tools
run on various hardware.

European Union has funde
NESSIE project (2000–200
ECRYPT I network (2004–
ECRYPT II network (2008

NESSIE's performance eva
tuned C implementations
of many cryptographic syst
all supporting the same AP
wrote a benchmarking tool
ran the toolkit on 25 comp

Many specific performance
e.g., 24 cycles/byte on P4
for 128-bit AES encryption

Implementor runs tool on his
favorite hardware (or emulator).
Adds "Results" section
to implementation paper
repeating what the tool says.

Summary:
Cryptographic implementor
is the benchmark implementor
and the benchmark operator.

This pattern repeats for
every cryptographic implementor.
Hundreds (thousands?) of
separate ad-hoc timing tools
run on various hardware.

European Union has funded
NESSIE project (2000–2003),
ECRYPT I network (2004–2008),
ECRYPT II network (2008–2012).

NESSIE's performance evaluators
tuned C implementations
of many cryptographic systems,
all supporting the same API;
wrote a benchmarking toolkit;
ran the toolkit on 25 computers.

Many specific performance results:
e.g., 24 cycles/byte on P4
for 128-bit AES encryption.

nentor runs tool on his
e hardware (or emulator).
"Results" section
lementation paper
ng what the tool says.

ary:
graphic implementor
benchmark implementor
e benchmark operator.

attern repeats for
cryptographic implementor.
eds (thousands?) of
te ad-hoc timing tools
various hardware.

European Union has funded
NESSIE project (2000–2003),
ECRYPT I network (2004–2008),
ECRYPT II network (2008–2012).

NESSIE's performance evaluators
tuned C implementations
of many cryptographic systems,
all supporting the same API;
wrote a benchmarking toolkit;
ran the toolkit on 25 computers.

Many specific performance results:
e.g., 24 cycles/byte on P4
for 128-bit AES encryption.

ECRYP
STVL,
include
STVL
ran eS

De Ca
eSTRE

Stream
matchi
were c
publish
benchm

e.g. 18
third-p

s tool on his
e (or emulator).
section
on paper
he tool says.

nplementor
k implementor
ark operator.

eats for
hic implementor.
ands?) of
timing tools
ardware.

European Union has funded
NESSIE project (2000–2003),
ECRYPT I network (2004–2008),
ECRYPT II network (2008–2012).

NESSIE's performance evaluators
tuned C implementations
of many cryptographic systems,
all supporting the same API;
wrote a benchmarking toolkit;
ran the toolkit on 25 computers.

Many specific performance results:
e.g., 24 cycles/byte on P4
for 128-bit AES encryption.

ECRYPT I had f
STVL, symmetri
included four wo
STVL WG 1, str
ran eSTREAM (2

De Cannière *pub*
eSTREAM bench

Stream-cipher im
matching the be
were contributed
published, often
benchmarked on

e.g. 18 cycles/by
third-party asm A

nis
ator).

ys.

r
ntor
or.

nentor.

ols

European Union has funded
NESSIE project (2000–2003),
ECRYPT I network (2004–2008),
ECRYPT II network (2008–2012).

NESSIE's performance evaluators
tuned C implementations
of many cryptographic systems,
all supporting the same API;
wrote a benchmarking toolkit;
ran the toolkit on 25 computers.

Many specific performance results:
e.g., 24 cycles/byte on P4
for 128-bit AES encryption.

ECRYPT I had five "virtua
STVL, symmetric-techniqu
included four working grou
STVL WG 1, stream-ciphe
ran eSTREAM (2004–2008

De Cannière *published*
eSTREAM benchmarking t

Stream-cipher implementat
matching the benchmarkin
were contributed by design
published, often tuned;
benchmarked on many com

e.g. 18 cycles/byte on P4 f
third-party asm AES in toc

European Union has funded
NESSIE project (2000–2003),
ECRYPT I network (2004–2008),
ECRYPT II network (2008–2012).

NESSIE's performance evaluators
tuned C implementations
of many cryptographic systems,
all supporting the same API;
wrote a benchmarking toolkit;
ran the toolkit on 25 computers.

Many specific performance results:
e.g., 24 cycles/byte on P4
for 128-bit AES encryption.

ECRYPT I had five "virtual labs."
STVL, symmetric-techniques lab,
included four working groups.
STVL WG 1, stream-cipher group,
ran eSTREAM (2004–2008).

De Cannière *published*
eSTREAM benchmarking toolkit.

Stream-cipher implementations
matching the benchmarking API
were contributed by designers,
published, often tuned;
benchmarked on many computers.

e.g. 18 cycles/byte on P4 for
third-party asm AES in toolkit.

ean Union has funded
E project (2000–2003),
PT I network (2004–2008),
PT II network (2008–2012).

E's performance evaluators
C implementations
y cryptographic systems,
porting the same API;
a benchmarking toolkit;
e toolkit on 25 computers.

specific performance results:
4 cycles/byte on P4
8-bit AES encryption.

ECRYPT I had five "virtual labs."
STVL, symmetric-techniques lab,
included four working groups.
STVL WG 1, stream-cipher group,
ran eSTREAM (2004–2008).

De Cannière *published*
eSTREAM benchmarking toolkit.

Stream-cipher implementations
matching the benchmarking API
were contributed by designers,
published, often tuned;
benchmarked on many computers.

e.g. 18 cycles/byte on P4 for
third-party asm AES in toolkit.

2006:
Applica
Lab," s
("ECR
of Asy
measur
encrypt

Publish

Project
Has wr
55 pub
matchi
Benchm

has funded
(2000–2003),
ork (2004–2008),
work (2008–2012).

mance evaluators
entations
raphic systems,
e same API;
arking toolkit;
n 25 computers.

rformance results:
yte on P4
encryption.

ECRYPT I had five "virtual labs."
STVL, symmetric-techniques lab,
included four working groups.
STVL WG 1, stream-cipher group,
ran eSTREAM (2004–2008).

De Cannière *published*
eSTREAM benchmarking toolkit.

Stream-cipher implementations
matching the benchmarking API
were contributed by designers,
published, often tuned;
benchmarked on many computers.

e.g. 18 cycles/byte on P4 for
third-party asm AES in toolkit.

2006: VAMPIRE
Application and
Lab," started eB
("ECRYPT Benc
of Asymmetric S
measuring efficie
encryption, signa

Published a new

Project is contin
Has written, coll
55 public-key im
matching the be
Benchmarked on

d
3),
-2008),
–2012).

luators

tems,
PI;
kit;
puters.

results:

.

ECRYPT I had five "virtual labs."
STVL, symmetric-techniques lab,
included four working groups.
STVL WG 1, stream-cipher group,
ran eSTREAM (2004–2008).

De Cannière *published*
eSTREAM benchmarking toolkit.

Stream-cipher implementations
matching the benchmarking API
were contributed by designers,
published, often tuned;
benchmarked on many computers.

e.g. 18 cycles/byte on P4 for
third-party asm AES in toolkit.

2006: VAMPIRE, "Virtual
Application and Implement
Lab," started eBATS
("ECRYPT Benchmarking
of Asymmetric Systems"),
measuring efficiency of pub
encryption, signatures, DH

Published a new toolkit.

Project is continuing.
Has written, collected, pub
55 public-key implementati
matching the benchmarkin
Benchmarked on many com

ECRYPT I had five "virtual labs."
STVL, symmetric-techniques lab,
included four working groups.
STVL WG 1, stream-cipher group,
ran eSTREAM (2004–2008).

De Cannière *published*
eSTREAM benchmarking toolkit.

Stream-cipher implementations
matching the benchmarking API
were contributed by designers,
published, often tuned;
benchmarked on many computers.

e.g. 18 cycles/byte on P4 for
third-party asm AES in toolkit.

2006: VAMPIRE, "Virtual
Application and Implementation
Lab," started eBATS
("ECRYPT Benchmarking
of Asymmetric Systems"),
measuring efficiency of public-key
encryption, signatures, DH.

Published a new toolkit.

Project is continuing.
Has written, collected, published
55 public-key implementations
matching the benchmarking API.
Benchmarked on many computers.

PT I had five "virtual labs."
symmetric-techniques lab,
ed four working groups.
WG 1, stream-cipher group,
TREAM (2004–2008).

nnière *published*
AM benchmarking toolkit.

-cipher implementations
ng the benchmarking API
ontributed by designers,
ed, often tuned;
marked on many computers.

cycles/byte on P4 for
arty asm AES in toolkit.

2006: VAMPIRE, "Virtual Application and Implementation Lab," started eBATS ("ECRYPT Benchmarking of Asymmetric Systems"), measuring efficiency of public-key encryption, signatures, DH.

Published a new toolkit.

Project is continuing.
Has written, collected, published
55 public-key implementations
matching the benchmarking API.
Benchmarked on many computers.

2008:
("ECR
of Stre
post-eS

VAMP
("ECR
of All S

eBACS
of Cryp
include
Contin

New to
with C
AES no

ive "virtual labs."
c-techniques lab,
rking groups.
eam-cipher group,
2004–2008).

lished
nmarking toolkit.

nplementations
nchmarking API
by designers,
tuned;
many computers.

te on P4 for
AES in toolkit.

2006: VAMPIRE, "Virtual Application and Implementation Lab," started eBATS ("ECRYPT Benchmarking of Asymmetric Systems"), measuring efficiency of public-key encryption, signatures, DH.

Published a new toolkit.

Project is continuing.
Has written, collected, published 55 public-key implementations matching the benchmarking API.
Benchmarked on many computers.

2008: VAMPIRE
("ECRYPT Benc
of Stream Ciphe
post-eSTREAM

VAMPIRE also s
("ECRYPT Benc
of All Submitted

eBACS ("ECRYP
of Cryptographic
includes eBATS,
Continues under

New toolkit, API
with CACE librar
AES now 14 cycl

al labs."
es lab,
ps.
r group,
3).

toolkit.

tions
g API
ers,

nputers.

for
olkit.

2006: VAMPIRE, "Virtual Application and Implementation Lab," started eBATS ("ECRYPT Benchmarking of Asymmetric Systems"), measuring efficiency of public-key encryption, signatures, DH.

Published a new toolkit.

Project is continuing.
Has written, collected, published 55 public-key implementations matching the benchmarking API.
Benchmarked on many computers.

2008: VAMPIRE started e
("ECRYPT Benchmarking
of Stream Ciphers") for
post-eSTREAM benchmar

VAMPIRE also started eBA
("ECRYPT Benchmarking
of All Submitted Hashes")

eBACS ("ECRYPT Benchm
of Cryptographic Systems"
includes eBATS, eBASH, e
Continues under ECRYPT

New toolkit, API; coordina
with CACE library (NaCl).
AES now 14 cycles/byte o

2006: VAMPIRE, "Virtual Application and Implementation Lab," started eBATS ("ECRYPT Benchmarking of Asymmetric Systems"), measuring efficiency of public-key encryption, signatures, DH.

Published a new toolkit.

Project is continuing.
Has written, collected, published 55 public-key implementations matching the benchmarking API.
Benchmarked on many computers.

2008: VAMPIRE started eBASC ("ECRYPT Benchmarking of Stream Ciphers") for post-eSTREAM benchmarks.

VAMPIRE also started eBASH ("ECRYPT Benchmarking of All Submitted Hashes").

eBACS ("ECRYPT Benchmarking of Cryptographic Systems") includes eBATS, eBASH, eBASC.
Continues under ECRYPT II.

New toolkit, API; coordinated with CACE library (NaCl).
AES now 14 cycles/byte on P4.

VAMPIRE, “Virtual
ation and Implementation
started eBATS
YPT Benchmarking
mmetric Systems”),
ring efficiency of public-key
tion, signatures, DH.

ned a new toolkit.

t is continuing.
ritten, collected, published
lic-key implementations
ng the benchmarking API.
marked on many computers.

2008: VAMPIRE started eBASC
(“ECRYPT Benchmarking
of Stream Ciphers”) for
post-eSTREAM benchmarks.

VAMPIRE also started eBASH
(“ECRYPT Benchmarking
of All Submitted Hashes”).

eBACS (“ECRYPT Benchmarking
of Cryptographic Systems”)
includes eBATS, eBASH, eBASC.
Continues under ECRYPT II.

New toolkit, API; coordinated
with CACE library (NaCl).
AES now 14 cycles/byte on P4.

Many a
over ac
- >100
- >100
- Many
- Very
- Publi
- Real
- Easy

Today

Biggest
Report
hard to
. . . but

, "Virtual
Implementation
ATS
chmarking
ystems"),
ncy of public-key
tures, DH.

toolkit.

uing.
ected, published
plementations
nchmarking API.
many computers.

2008: VAMPIRE started eBASC
("ECRYPT Benchmarking
of Stream Ciphers") for
post-eSTREAM benchmarks.

VAMPIRE also started eBASH
("ECRYPT Benchmarking
of All Submitted Hashes").

eBACS ("ECRYPT Benchmarking
of Cryptographic Systems")
includes eBATS, eBASH, eBASC.
Continues under ECRYPT II.

New toolkit, API; coordinated
with CACE library (NaCl).
AES now 14 cycles/byte on P4.

Many advantages
over ad-hoc benc
- >1000 compile
- >100 machine-
- Many message
- Very high relia
- Public verifiabi
- Real API, not
- Easy for implen

Today have 431

Biggest disadvan
Report latency is
hard to use durin
... but we're wo

tation

blic-key
.

blished
ions
g API.
mputers.

2008: VAMPIRE started eBASC
("ECRYPT Benchmarking
of Stream Ciphers") for
post-eSTREAM benchmarks.

VAMPIRE also started eBASH
("ECRYPT Benchmarking
of All Submitted Hashes").

eBACS ("ECRYPT Benchmarking
of Cryptographic Systems")
includes eBATS, eBASH, eBASC.
Continues under ECRYPT II.

New toolkit, API; coordinated
with CACE library (NaCl).
AES now 14 cycles/byte on P4.

Many advantages of eBAC
over ad-hoc benchmarking:

- >1000 compiler options.
- >100 machine-ABI pairs
- Many message lengths.
- Very high reliability.
- Public verifiability.
- Real API, not only timin
- Easy for implementor!

Today have 431 implement

Biggest disadvantage:
Report latency is high;
hard to use during develop

... but we're working on t

2008: VAMPIRE started eBASC
("ECRYPT Benchmarking
of Stream Ciphers") for
post-eSTREAM benchmarks.

VAMPIRE also started eBASH
("ECRYPT Benchmarking
of All Submitted Hashes").

eBACS ("ECRYPT Benchmarking
of Cryptographic Systems")
includes eBATS, eBASH, eBASC.
Continues under ECRYPT II.

New toolkit, API; coordinated
with CACE library (NaCl).
AES now 14 cycles/byte on P4.

Many advantages of eBACS
over ad-hoc benchmarking:

- $>$1000 compiler options.
- $>$100 machine-ABI pairs.
- Many message lengths.
- Very high reliability.
- Public verifiability.
- Real API, not only timing.
- Easy for implementor!

Today have 431 implementations.

Biggest disadvantage:
Report latency is high;
hard to use during development.
... but we're working on this.