Making sure
crypto stays insecure

Daniel J. Bernstein

University of Illinois at Chicago &
Technische Universiteit Eindhoven

Terrorist in Hong Kong
prepares to throw deadly weapon
at Chinese government workers.

Image credit: Reuters.

sure

tays insecure

. Bernstein

ty of Illinois at Chicago &

che Universiteit Eindhoven

Terrorist in Hong Kong
prepares to throw deadly weapon
at Chinese government workers.

Image credit: Reuters.


Drug-de
invades
begins s

Image c

ure

n

is at Chicago &

siteit Eindhoven

Terrorist in Hong Kong
prepares to throw deadly weapon
at Chinese government workers.

Image credit: Reuters.


Drug-dealing carte
invades city in Mo
begins selling addi

Image credit: Wik

ago &
hoven

Terrorist in Hong Kong
prepares to throw deadly weapon
at Chinese government workers.

Image credit: Reuters.


Drug-dealing cartel "Starbuc
invades city in Morocco;
begins selling addictive liqui

Image credit: Wikipedia.

Terrorist in Hong Kong
prepares to throw deadly weapon
at Chinese government workers.

Image credit: Reuters.


Drug-dealing cartel "Starbucks"
invades city in Morocco;
begins selling addictive liquid.

Image credit: Wikipedia.

t in Hong Kong
to throw deadly weapon
ese government workers.

redit: Reuters.


Drug-dealing cartel "Starbucks"
invades city in Morocco;
begins selling addictive liquid.

Image credit: Wikipedia.

Pedophi
to remov
sexually

Image cr

Kong
deadly weapon
ment workers.

ters.


Drug-dealing cartel "Starbucks"
invades city in Morocco;
begins selling addictive liquid.

Image credit: Wikipedia.

Pedophile convinc
to remove most of
sexually abuses ch

Image credit: Chil

apon
kers.


Drug-dealing cartel "Starbucks"
invades city in Morocco;
begins selling addictive liquid.

Image credit: Wikipedia.

Pedophile convinces helpless
to remove most of her cloth
sexually abuses child in publ

Image credit: Child pornogr


Drug-dealing cartel “Starbucks”
invades city in Morocco;
begins selling addictive liquid.

Image credit: Wikipedia.

Pedophile convinces helpless child
to remove most of her clothing;
sexually abuses child in public.

Image credit: Child pornographer.


aling cartel "Starbucks"
city in Morocco;
elling addictive liquid.

redit: Wikipedia.

Pedophile convinces helpless child
to remove most of her clothing;
sexually abuses child in public.

Image credit: Child pornographer.


Criminal
calling it
sells clas

Image cr


el "Starbucks"
rocco;
ctive liquid.

ipedia.

Pedophile convinces helpless child to remove most of her clothing; sexually abuses child in public.

Image credit: Child pornographer.


Criminal organizat
calling itself "The
sells classified gov

Image credit: The

cks"

d.

Pedophile convinces helpless child to remove most of her clothing; sexually abuses child in public.

Image credit: Child pornographer.


Criminal organization calling itself "The Guardian" sells classified government se

Image credit: The Guardian

Pedophile convinces helpless child
to remove most of her clothing;
sexually abuses child in public.

Image credit: Child pornographer.


Criminal organization
calling itself “The Guardian”
sells classified government secrets.

Image credit: The Guardian.

le convinces helpless child
ve most of her clothing;
abuses child in public.

redit: Child pornographer.

theguardian

News | Sport | Comment | Culture | Business | Money | Life & Offers | Jobs

News › US news › US national security

Series: Glenn Greenwald on security and liberty

## NSA collecting phone records of millions of Verizon customers daily

**Exclusive:** Top secret court order requiring Verizon to hand over all call data shows scale of domestic surveillance under Obama

• Read the Verizon court order in full here

Criminal organization
calling itself "The Guardian"
sells classified government secrets.

Image credit: The Guardian.

We have
everythi
so that
drug dea
pedophil

es helpless child
f her clothing;
ild in public.

d pornographer.

theguardian

News | Sport | Comment | Culture | Business | Money | Life & Offers | Jobs

News › US news › US national security

Series: Glenn Greenwald on security and liberty

# NSA collecting phone records of millions of Verizon customers daily

**Exclusive:** Top secret court order requiring Verizon to hand over all call data shows scale of domestic surveillance under Obama

• Read the Verizon court order in full here

Criminal organization
calling itself "The Guardian"
sells classified government secrets.

Image credit: The Guardian.

We have to watch
everything that pe
so that we can cat
drug dealers, orga
pedophiles, murde

s child
ing;
ic.

apher.

theguardian

News | Sport | Comment | Culture | Business | Money | Life & Offers | Jobs

News › US news › US national security

Series: Glenn Greenwald on security and liberty

## NSA collecting phone records of millions of Verizon customers daily

**Exclusive:** Top secret court order requiring Verizon to hand over all call data shows scale of domestic surveillance under Obama

• Read the Verizon court order in full here

Criminal organization
calling itself "The Guardian"
sells classified government secrets.

Image credit: The Guardian.

We have to watch and listen
everything that people are d
so that we can catch terroris
drug dealers, organized crim
pedophiles, murderers, etc.


theguardian

News | Sport | Comment | Culture | Business | Money | Life & Offers | Jobs

News › US news › US national security

Series: Glenn Greenwald on security and liberty

NSA collecting phone records of millions of Verizon customers daily

**Exclusive:** Top secret court order requiring Verizon to hand over all call data shows scale of domestic surveillance under Obama

• Read the Verizon court order in full here

Criminal organization
calling itself “The Guardian”
sells classified government secrets.

Image credit: The Guardian.

We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.


theguardian

News | Sport | Comment | Culture | Business | Money | Life & Offers | Jobs

News › US news › US national security

Series: Glenn Greenwald on security and liberty

NSA collecting phone records of millions of Verizon customers daily

**Exclusive:** Top secret court order requiring Verizon to hand over all call data shows scale of domestic surveillance under Obama

• Read the Verizon court order in full here

Criminal organization
calling itself “The Guardian”
sells classified government secrets.

Image credit: The Guardian.

We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.

We try to systematically monitor and record all Internet traffic.

**But what if it’s encrypted?**

theguardian

News | Sport | Comment | Culture | Business | Money | Life & Offers | Jobs

News › US news › US national security

Series: Glenn Greenwald on security and liberty

NSA collecting phone records of millions of Verizon customers daily

**Exclusive:** Top secret court order requiring Verizon to hand over all call data shows scale of domestic surveillance under Obama

• Read the Verizon court order in full here

Criminal organization
calling itself "The Guardian"
sells classified government secrets.

Image credit: The Guardian.

We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.

We try to systematically monitor and record all Internet traffic.
**But what if it's encrypted?**

This talk gives some examples of how we've manipulated the world's crypto ecosystem so that we can understand almost all of this traffic.


n

t | Comment | Culture | Business | Money | Life &

news 〉 US national security

enwald on security and liberty

llecting phone records of
s of Verizon customers daily

p secret court order requiring Verizon to hand over
ows scale of domestic surveillance under Obama

rizon court order in full here

organization

tself "The Guardian"

ssified government secrets.

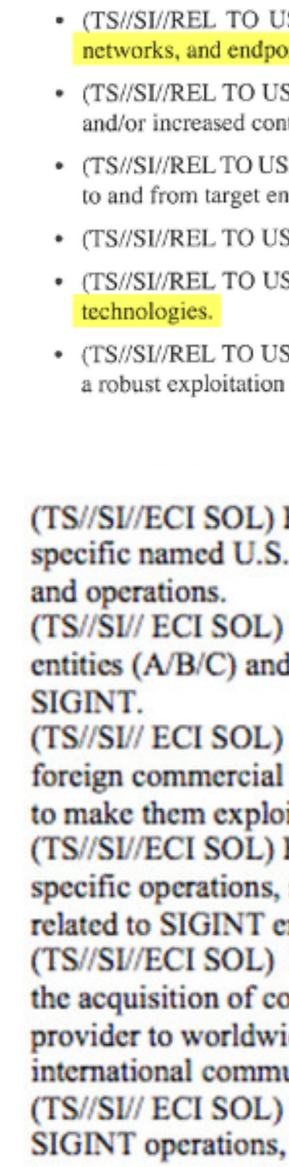
redit: The Guardian.

We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.

We try to systematically monitor and record all Internet traffic.

**But what if it's encrypted?**

This talk gives some examples of how we've manipulated the world's crypto ecosystem so that we can understand almost all of this traffic.


• (TS//SI//REL TO US
networks, and endpo
• (TS//SI//REL TO US
and/or increased con
• (TS//SI//REL TO US
to and from target en
• (TS//SI//REL TO US
• (TS//SI//REL TO US
technologies.
• (TS//SI//REL TO US
a robust exploitation

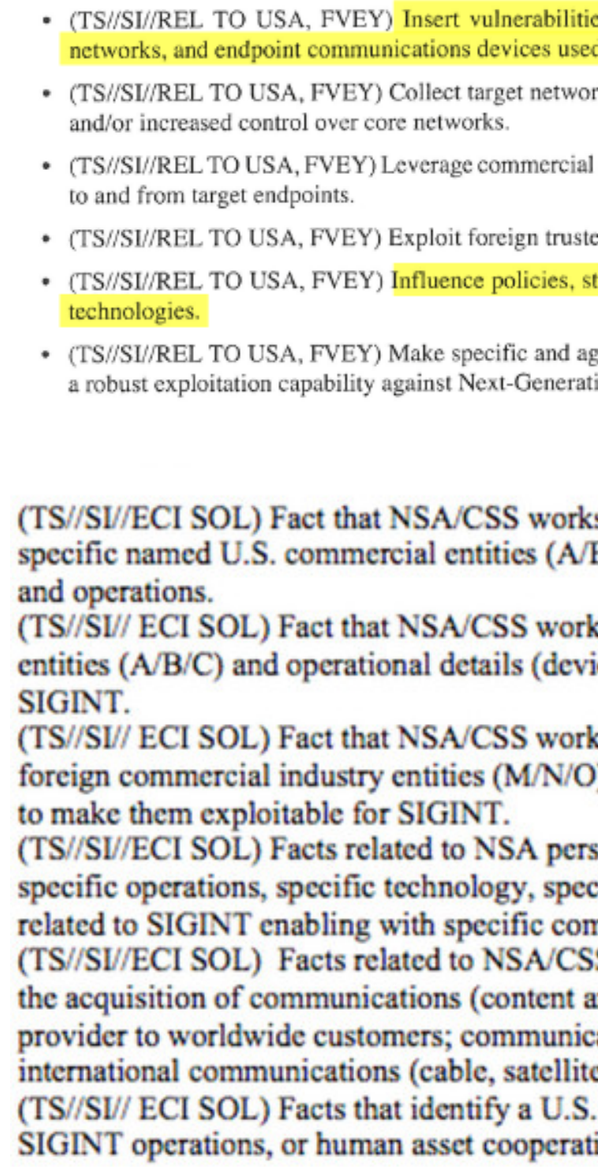
(TS//SI//ECI SOL)
specific named U.S.
and operations.
(TS//SI// ECI SOL)
entities (A/B/C) and
SIGINT.
(TS//SI// ECI SOL)
foreign commercial
to make them explo
(TS//SI//ECI SOL)
specific operations,
related to SIGINT e
(TS//SI//ECI SOL)
the acquisition of co
provider to worldwi
international commu
(TS//SI// ECI SOL)
SIGINT operations,


We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.

We try to systematically monitor and record all Internet traffic.

**But what if it's encrypted?**

This talk gives some examples of how we've manipulated the world's crypto ecosystem so that we can understand almost all of this traffic.



- (TS//SI//REL TO USA, FVEY) Insert vulnerabilitie networks, and endpoint communications devices used
- (TS//SI//REL TO USA, FVEY) Collect target networ and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign truste
- (TS//SI//REL TO USA, FVEY) Influence policies, st technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and ag a robust exploitation capability against Next-Generati

(TS//SI//ECI SOL) Fact that NSA/CSS works specific named U.S. commercial entities (A/B and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS work entities (A/B/C) and operational details (devi SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS work foreign commercial industry entities (M/N/O to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA pers specific operations, specific technology, spec related to SIGINT enabling with specific com
(TS//SI//ECI SOL) Facts related to NSA/CSS the acquisition of communications (content a provider to worldwide customers; communica international communications (cable, satellite
(TS//SI// ECI SOL) Facts that identify a U.S. SIGINT operations, or human asset cooperati

Money | Life &

rds of
ers daily
to hand over
nder Obama

,

ecrets.

.

We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.

We try to systematically monitor and record all Internet traffic.

**But what if it's encrypted?**

This talk gives some examples of how we've manipulated the world's crypto ecosystem so that we can understand almost all of this traffic.

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption s networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperat and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver o to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and techn
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for co technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate a robust exploitation capability against Next-Generation Wireless (NGW) communica

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractua
specific named U.S. commercial entities (A/B/C) to conduct SIGINT
and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.
entities (A/B/C) and operational details (devices/products) to make the
SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign pa
foreign commercial industry entities (M/N/O) and operational details
to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), oper
specific operations, specific technology, specific locations and covert
related to SIGINT enabling with specific commercial entities (A/B/C)
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. com
the acquisition of communications (content and metadata) provided by
provider to worldwide customers; communications transiting the U.S.;
international communications (cable, satellite, etc.) mediums provided
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial p
SIGINT operations, or human asset cooperating with NSA/CSS.

We have to watch and listen to everything that people are doing so that we can catch terrorists, drug dealers, organized criminals, pedophiles, murderers, etc.

We try to systematically monitor and record all Internet traffic.

**But what if it's encrypted?**

This talk gives some examples of how we've manipulated the world's crypto ecosystem so that we can understand almost all of this traffic.

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractual relationships with specific named U.S. commercial entities (A/B/C) to conduct SIGINT enabling programs and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.S. commercial entities (A/B/C) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and foreign commercial industry entities (M/N/O) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), operational meetings, specific operations, specific technology, specific locations and covert communications related to SIGINT enabling with specific commercial entities (A/B/C).
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. commercial entities on the acquisition of communications (content and metadata) provided by the U.S. service provider to worldwide customers; communications transiting the U.S.; or access to international communications (cable, satellite, etc.) mediums provided by the U.S. entity.
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial platform conducting SIGINT operations, or human asset cooperating with NSA/CSS.

e to watch and listen to
ng that people are doing
we can catch terrorists,
alers, organized criminals,
les, murderers, etc.

to systematically monitor
ord all Internet traffic.
**at if it's encrypted?**

k gives some examples
we've manipulated
d's crypto ecosystem
we can understand
ll of this traffic.

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractual relationships with specific named U.S. commercial entities (A/B/C) to conduct SIGINT enabling programs and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.S. commercial entities (A/B/C) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and foreign commercial industry entities (M/N/O) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), operational meetings, specific operations, specific technology, specific locations and covert communications related to SIGINT enabling with specific commercial entities (A/B/C).
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. commercial entities on the acquisition of communications (content and metadata) provided by the U.S. service provider to worldwide customers; communications transiting the U.S.; or access to international communications (cable, satellite, etc.) mediums provided by the U.S. entity.
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial platform conducting SIGINT operations, or human asset cooperating with NSA/CSS.

Other us
not cove

Manipul
so that s
Break in
hundreds
screens,

and listen to
eople are doing
tch terrorists,
nized criminals,
rers, etc.

tically monitor
rnet traffic.
**encrypted?**

me examples
ipulated
ecosystem
derstand
traffic.

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractual relationships with specific named U.S. commercial entities (A/B/C) to conduct SIGINT enabling programs and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.S. commercial entities (A/B/C) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and foreign commercial industry entities (M/N/O) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), operational meetings, specific operations, specific technology, specific locations and covert communications related to SIGINT enabling with specific commercial entities (A/B/C).
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. commercial entities on the acquisition of communications (content and metadata) provided by the U.S. service provider to worldwide customers; communications transiting the U.S.; or access to international communications (cable, satellite, etc.) mediums provided by the U.S. entity.
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial platform conducting SIGINT operations, or human asset cooperating with NSA/CSS.

Other useful strate
not covered in this

Manipulate *softwa*
so that software st
Break into comput
hundreds of millio
screens, microphon

n to
oing
sts,
inals,

nitor
c.
?

es

n

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractual relationships with specific named U.S. commercial entities (A/B/C) to conduct SIGINT enabling programs and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.S. commercial entities (A/B/C) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and foreign commercial industry entities (M/N/O) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), operational meetings, specific operations, specific technology, specific locations and covert communications related to SIGINT enabling with specific commercial entities (A/B/C).
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. commercial entities on the acquisition of communications (content and metadata) provided by the U.S. service provider to worldwide customers; communications transiting the U.S.; or access to international communications (cable, satellite, etc.) mediums provided by the U.S. entity.
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial platform conducting SIGINT operations, or human asset cooperating with NSA/CSS.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosyst
so that software stays insecu
Break into computers; acces
hundreds of millions of disks
screens, microphones, camer

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractual relationships with specific named U.S. commercial entities (A/B/C) to conduct SIGINT enabling programs and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.S. commercial entities (A/B/C) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and foreign commercial industry entities (M/N/O) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), operational meetings, specific operations, specific technology, specific locations and covert communications related to SIGINT enabling with specific commercial entities (A/B/C).
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. commercial entities on the acquisition of communications (content and metadata) provided by the U.S. service provider to worldwide customers; communications transiting the U.S.; or access to international communications (cable, satellite, etc.) mediums provided by the U.S. entity.
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial platform conducting SIGINT operations, or human asset cooperating with NSA/CSS.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.

(TS//SI//ECI SOL) Fact that NSA/CSS works with and has contractual relationships with specific named U.S. commercial entities (A/B/C) to conduct SIGINT enabling programs and operations.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific named U.S. commercial entities (A/B/C) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI// ECI SOL) Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and foreign commercial industry entities (M/N/O) and operational details (devices/products) to make them exploitable for SIGINT.
(TS//SI//ECI SOL) Facts related to NSA personnel (under cover), operational meetings, specific operations, specific technology, specific locations and covert communications related to SIGINT enabling with specific commercial entities (A/B/C).
(TS//SI//ECI SOL) Facts related to NSA/CSS working with U.S. commercial entities on the acquisition of communications (content and metadata) provided by the U.S. service provider to worldwide customers; communications transiting the U.S.; or access to international communications (cable, satellite, etc.) mediums provided by the U.S. entity.
(TS//SI// ECI SOL) Facts that identify a U.S. or foreign commercial platform conducting SIGINT operations, or human asset cooperating with NSA/CSS.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem so that software stays insecure. Break into computers; access hundreds of millions of disks, screens, microphones, cameras.

Add back doors to *hardware*. e.g. 2012 U.S. government report says that Chinese-manufactured routers provide “Chinese intelligence services access to telecommunication networks”.

SA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems,
int communications devices used by targets.

SA, FVEY) Collect target network data and metadata via cooperative network carriers
trol over core networks.

A, FVEY) Leverage commercial capabilities to remotely deliver or receive information
dpoints.

SA, FVEY) Exploit foreign trusted computing platforms and technologies.

SA, FVEY) Influence policies, standards and specification for commercial public key

SA, FVEY) Make specific and aggressive investments to facilitate the development of
capability against Next-Generation Wireless (NGW) communications.

Fact that NSA/CSS works with and has contractual relationships with
commercial entities (A/B/C) to conduct SIGINT enabling programs

Fact that NSA/CSS works with specific named U.S. commercial
operational details (devices/products) to make them exploitable for

Fact that NSA/CSS works with specific foreign partners (X/Y/Z) and
industry entities (M/N/O) and operational details (devices/products)
itable for SIGINT.
Facts related to NSA personnel (under cover), operational meetings,
specific technology, specific locations and covert communications
nabling with specific commercial entities (A/B/C).
Facts related to NSA/CSS working with U.S. commercial entities on
mmunications (content and metadata) provided by the U.S. service
de customers; communications transiting the U.S.; or access to
unications (cable, satellite, etc.) mediums provided by the U.S. entity.
Facts that identify a U.S. or foreign commercial platform conducting
or human asset cooperating with NSA/CSS.

Other useful strategies,
not covered in this talk:

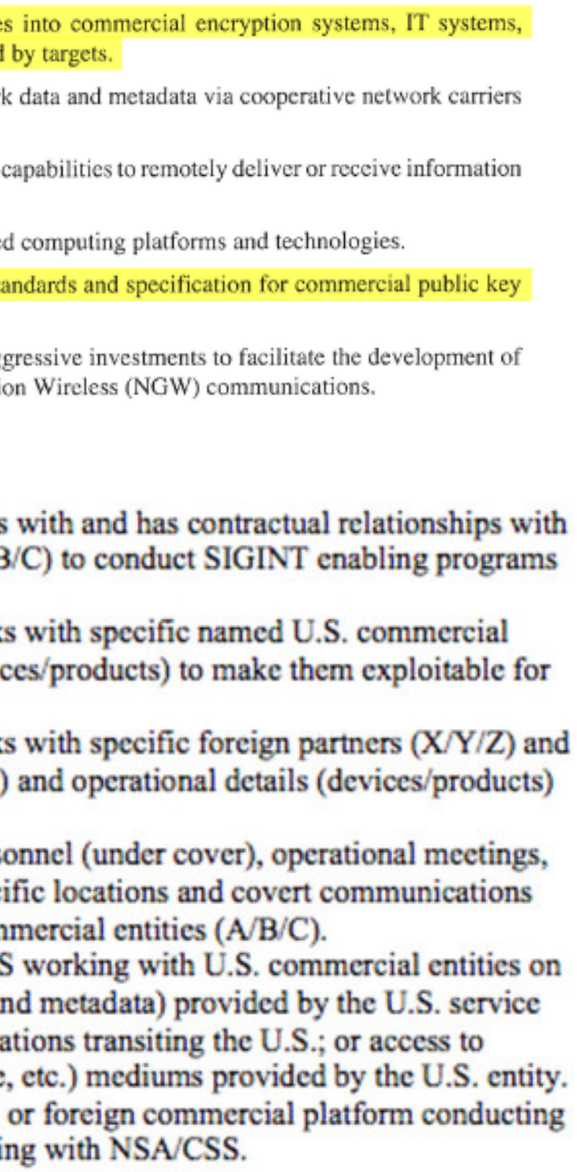
Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide “Chinese
intelligence services access to
telecommunication networks”.

Some im

1. “We”

I want s



es into commercial encryption systems, IT systems,
d by targets.

k data and metadata via cooperative network carriers

capabilities to remotely deliver or receive information

ed computing platforms and technologies.

andards and specification for commercial public key

gressive investments to facilitate the development of
ion Wireless (NGW) communications.

s with and has contractual relationships with
B/C) to conduct SIGINT enabling programs

ts with specific named U.S. commercial
ces/products) to make them exploitable for

ts with specific foreign partners (X/Y/Z) and
) and operational details (devices/products)

sonnel (under cover), operational meetings,
ific locations and covert communications
mmercial entities (A/B/C).
S working with U.S. commercial entities on
nd metadata) provided by the U.S. service
ations transiting the U.S.; or access to
e, etc.) mediums provided by the U.S. entity.
or foreign commercial platform conducting
ing with NSA/CSS.

Other useful strategies,
not covered in this talk:

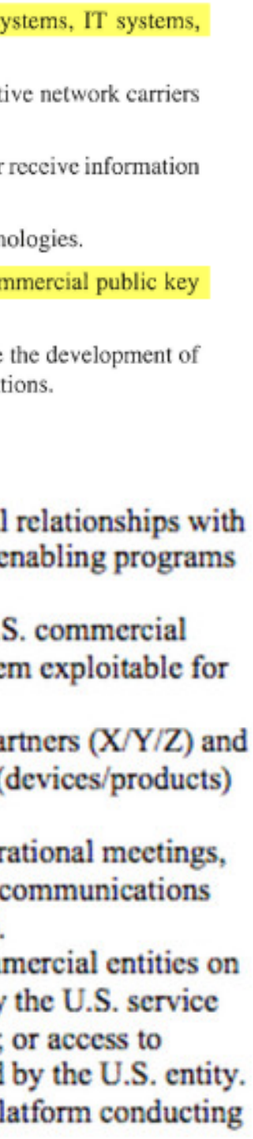
Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide “Chinese
intelligence services access to
telecommunication networks”.

Some important c

1. “We” doesn’t i

I want secure cryp


ystems, IT systems,
tive network carriers
r receive information
nologies.
mmercial public key
e the development of
tions.
l relationships with
enabling programs
S. commercial
em exploitable for
artners (X/Y/Z) and
(devices/products)
rational meetings,
communications
mercial entities on
y the U.S. service
; or access to
l by the U.S. entity.
latform conducting

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide "Chinese
intelligence services access to
telecommunication networks".

## Some important clarification

1. "We" doesn't include me
I want secure crypto.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide "Chinese
intelligence services access to
telecommunication networks".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide "Chinese
intelligence services access to
telecommunication networks".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide "Chinese
intelligence services access to
telecommunication networks".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

3. I don't know how much
of today's crypto ecosystem
was deliberately manipulated.

Other useful strategies,
not covered in this talk:

Manipulate *software* ecosystem
so that software stays insecure.
Break into computers; access
hundreds of millions of disks,
screens, microphones, cameras.

Add back doors to *hardware*.
e.g. 2012 U.S. government report
says that Chinese-manufactured
routers provide "Chinese
intelligence services access to
telecommunication networks".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

3. I don't know how much
of today's crypto ecosystem
was deliberately manipulated.

This talk is actually
a thought experiment:
how *could* an attacker manipulate
the ecosystem for insecurity?

seful strategies,
ered in this talk:

ate *software* ecosystem
software stays insecure.
to computers; access
s of millions of disks,
microphones, cameras.

k doors to *hardware*.
2 U.S. government report
t Chinese-manufactured
provide "Chinese
nce services access to
munication networks".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

3. I don't know how much
of today's crypto ecosystem
was deliberately manipulated.

This talk is actually
a thought experiment:
how *could* an attacker manipulate
the ecosystem for insecurity?

## Timing a

2005 Os
65ms to
used for
Attack p
but with

Almost a
use fast
Kernel's
influence
influenci
influenci
of the at
65ms: c

egies,
s talk:

*re* ecosystem
tays insecure.
ters; access
ns of disks,
nes, cameras.

*hardware*.
vernment report
manufactured
Chinese
es access to
n networks".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

3. I don't know how much
of today's crypto ecosystem
was deliberately manipulated.

This talk is actually
a thought experiment:
how *could* an attacker manipulate
the ecosystem for insecurity?

## Timing attacks

2005 Osvik–Shami
65ms to steal Linu
used for hard-disk
Attack process on
but without privile

Almost all AES im
use fast lookup ta
Kernel's secret AE
influences table-lo
influencing CPU c
influencing measur
of the attack proc
65ms: compute ke

tem
ure.
ss
s,
ras.
e.
report
ured
o
".

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

3. I don't know how much
of today's crypto ecosystem
was deliberately manipulated.

This talk is actually
a thought experiment:
how *could* an attacker manipulate
the ecosystem for insecurity?

## Timing attacks

2005 Osvik–Shamir–Tromer:
65ms to steal Linux AES ke
used for hard-disk encryptio
Attack process on same CP
but without privileges.

Almost all AES implementat
use fast lookup tables.
Kernel's secret AES key
influences table-load address
influencing CPU cache state
influencing measurable timi
of the attack process.
65ms: compute key from ti

## Some important clarifications

1. "We" doesn't include me.
I want secure crypto.

2. Their actions violate
fundamental human rights.

3. I don't know how much
of today's crypto ecosystem
was deliberately manipulated.

This talk is actually
a thought experiment:
how *could* an attacker manipulate
the ecosystem for insecurity?

## Timing attacks

2005 Osvik–Shamir–Tromer:
65ms to steal Linux AES key
used for hard-disk encryption.
Attack process on same CPU
but without privileges.

Almost all AES implementations
use fast lookup tables.
Kernel's secret AES key
influences table-load addresses,
influencing CPU cache state,
influencing measurable timings
of the attack process.
65ms: compute key from timings.

## nportant clarifications

' doesn't include me.
ecure crypto.

r actions violate
ental human rights.

't know how much
's crypto ecosystem
berately manipulated.

k is actually
nt experiment:
*ld* an attacker manipulate
ystem for insecurity?

## Timing attacks

2005 Osvik–Shamir–Tromer:
65ms to steal Linux AES key
used for hard-disk encryption.
Attack process on same CPU
but without privileges.

Almost all AES implementations
use fast lookup tables.
Kernel's secret AES key
influences table-load addresses,
influencing CPU cache state,
influencing measurable timings
of the attack process.
65ms: compute key from timings.

2011 Br
minutes
machine
Secret b
influence

Most cry
has man
variation
e.g., `mem`

Many m
2014 va
extracte
from 25

larifications

nclude me.
to.

iolate
an rights.

ow much
ecosystem
anipulated.

ly
ent:
cker manipulate
insecurity?

Timing attacks

2005 Osvik–Shamir–Tromer:
65ms to steal Linux AES key
used for hard-disk encryption.
Attack process on same CPU
but without privileges.

Almost all AES implementations
use fast lookup tables.
Kernel's secret AES key
influences table-load addresses,
influencing CPU cache state,
influencing measurable timings
of the attack process.
65ms: compute key from timings.

2011 Brumley–Tuv
minutes to steal a
machine's OpenSS
Secret branch con
influence timings.

Most cryptographi
has many more sm
variations in timin
e.g., `memcmp` for IP

Many more timing
2014 van de Pol–S
extracted Bitcoin
from 25 OpenSSL

ns

e.

d.

pulate

?

## Timing attacks

2005 Osvik–Shamir–Tromer:
65ms to steal Linux AES key
used for hard-disk encryption.
Attack process on same CPU
but without privileges.

Almost all AES implementations
use fast lookup tables.
Kernel's secret AES key
influences table-load addresses,
influencing CPU cache state,
influencing measurable timings
of the attack process.
65ms: compute key from timings.

2011 Brumley–Tuveri:
minutes to steal another
machine's OpenSSL ECDSA
Secret branch conditions
influence timings.

Most cryptographic software
has many more small-scale
variations in timing:
e.g., `memcmp` for IPsec MAC

Many more timing attacks:
2014 van de Pol–Smart–Yar
extracted Bitcoin secret keys
from 25 OpenSSL signatures

## Timing attacks

2005 Osvik–Shamir–Tromer:
65ms to steal Linux AES key
used for hard-disk encryption.
Attack process on same CPU
but without privileges.

Almost all AES implementations
use fast lookup tables.
Kernel's secret AES key
influences table-load addresses,
influencing CPU cache state,
influencing measurable timings
of the attack process.
65ms: compute key from timings.

2011 Brumley–Tuveri:
minutes to steal another
machine's OpenSSL ECDSA key.
Secret branch conditions
influence timings.

Most cryptographic software
has many more small-scale
variations in timing:
e.g., `memcmp` for IPsec MACs.

Many more timing attacks: e.g.
2014 van de Pol–Smart–Yarom
extracted Bitcoin secret keys
from 25 OpenSSL signatures.

attacks

vik–Shamir–Tromer:
steal Linux AES key
hard-disk encryption.
process on same CPU
hout privileges.

all AES implementations
lookup tables.
secret AES key
es table-load addresses,
ng CPU cache state,
ng measurable timings
ttack process.
ompute key from timings.

2011 Brumley–Tuveri:
minutes to steal another
machine's OpenSSL ECDSA key.
Secret branch conditions
influence timings.

Most cryptographic software
has many more small-scale
variations in timing:
e.g., `memcmp` for IPsec MACs.

Many more timing attacks: e.g.
2014 van de Pol–Smart–Yarom
extracted Bitcoin secret keys
from 25 OpenSSL signatures.

Manufac
that suc

Maybe t
won't tr

2001 NI
developr
Encrypti
"A gene
timing a
each enc
operatio
amount
not vuln

ir–Tromer:
ux AES key
encryption.
same CPU
eges.

nplementations
bles.
ES key
ad addresses,
ache state,
rable timings
ess.
ey from timings.

2011 Brumley–Tuveri:
minutes to steal another
machine's OpenSSL ECDSA key.
Secret branch conditions
influence timings.

Most cryptographic software
has many more small-scale
variations in timing:
e.g., `memcmp` for IPsec MACs.

Many more timing attacks: e.g.
2014 van de Pol–Smart–Yarom
extracted Bitcoin secret keys
from 25 OpenSSL signatures.

Manufacture publi
that such attacks

Maybe terrorists A
won't try to stop t

2001 NIST "Repo
development of th
Encryption Standa
"A general defense
timing attacks is t
each encryption ar
operation runs in t
amount of time. .
not vulnerable to t

y
n.
U

tions

ses,
e,
ngs

mings.

2011 Brumley–Tuveri:
minutes to steal another
machine's OpenSSL ECDSA key.
Secret branch conditions
influence timings.

Most cryptographic software
has many more small-scale
variations in timing:
e.g., `memcmp` for IPsec MACs.

Many more timing attacks: e.g.
2014 van de Pol–Smart–Yarom
extracted Bitcoin secret keys
from 25 OpenSSL signatures.

Manufacture public denials
that such attacks exist.

Maybe terrorists Alice and B
won't try to stop the attack

2001 NIST "Report on the
development of the Advance
Encryption Standard (AES)"
"A general defense against
timing attacks is to ensure t
each encryption and decrypt
operation runs in the same
amount of time. ... Table l
not vulnerable to timing atta

2011 Brumley–Tuveri:
minutes to steal another
machine's OpenSSL ECDSA key.
Secret branch conditions
influence timings.

Most cryptographic software
has many more small-scale
variations in timing:
e.g., `memcmp` for IPsec MACs.

Many more timing attacks: e.g.
2014 van de Pol–Smart–Yarom
extracted Bitcoin secret keys
from 25 OpenSSL signatures.

Manufacture public denials
that such attacks exist.

Maybe terrorists Alice and Bob
won't try to stop the attacks.

2001 NIST "Report on the
development of the Advanced
Encryption Standard (AES)":
"A general defense against
timing attacks is to ensure that
each encryption and decryption
operation runs in the same
amount of time. ... Table lookup:
not vulnerable to timing attacks."

umley–Tuveri:
to steal another
's OpenSSL ECDSA key.
ranch conditions
e timings.

yptographic software
y more small-scale
ns in timing:
`mcmp` for IPsec MACs.

ore timing attacks: e.g.
n de Pol–Smart–Yarom
d Bitcoin secret keys
OpenSSL signatures.

Manufacture public denials that such attacks exist.

Maybe terrorists Alice and Bob won't try to stop the attacks.

2001 NIST "Report on the development of the Advanced Encryption Standard (AES)": "A general defense against timing attacks is to ensure that each encryption and decryption operation runs in the same amount of time. . . . Table lookup: not vulnerable to timing attacks."

2008 RF
Layer Se
Version
small tim
performa
extent o
fragmen
be large
due to t
existing
of the ti

veri:
nother
SL ECDSA key.
ditions

ic software
nall-scale
g:
Psec MACs.

g attacks: e.g.
Smart–Yarom
secret keys
signatures.

Manufacture public denials that such attacks exist.

Maybe terrorists Alice and Bob won't try to stop the attacks.

2001 NIST "Report on the development of the Advanced Encryption Standard (AES)": "A general defense against timing attacks is to ensure that each encryption and decryption operation runs in the same amount of time. . . . Table lookup: not vulnerable to timing attacks."

2008 RFC 5246 "T
Layer Security (TL
Version 1.2": "Th
small timing chann
performance depen
extent on the size
fragment, but it is
be large enough to
due to the large bl
existing MACs and
of the timing signa

key.

s.

e.g.

om

s

s.

Manufacture public denials that such attacks exist.

Maybe terrorists Alice and Bob won't try to stop the attacks.

2001 NIST "Report on the development of the Advanced Encryption Standard (AES)": "A general defense against timing attacks is to ensure that each encryption and decryption operation runs in the same amount of time. . . . Table lookup: not vulnerable to timing attacks."

2008 RFC 5246 "The Transp
Layer Security (TLS) Protoc
Version 1.2": "This leaves a
small timing channel, since
performance depends to som
extent on the size of the dat
fragment, but it is not believ
be large enough to be exploi
due to the large block size o
existing MACs and the smal
of the timing signal."

Manufacture public denials that such attacks exist.

Maybe terrorists Alice and Bob won't try to stop the attacks.

2001 NIST "Report on the development of the Advanced Encryption Standard (AES)": "A general defense against timing attacks is to ensure that each encryption and decryption operation runs in the same amount of time. . . . Table lookup: not vulnerable to timing attacks."

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

Manufacture public denials that such attacks exist.

Maybe terrorists Alice and Bob won't try to stop the attacks.

2001 NIST "Report on the development of the Advanced Encryption Standard (AES)": "A general defense against timing attacks is to ensure that each encryption and decryption operation runs in the same amount of time. . . . Table lookup: not vulnerable to timing attacks."

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

2013 AlFardan–Paterson "Lucky Thirteen: breaking the TLS and DTLS record protocols": exploit these timings; steal plaintext.

cture public denials
h attacks exist.

errorists Alice and Bob
y to stop the attacks.

ST "Report on the
ment of the Advanced
on Standard (AES)":
ral defense against
ttacks is to ensure that
cryption and decryption
n runs in the same
of time. . . . Table lookup:
erable to timing attacks."

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

2013 AlFardan–Paterson "Lucky Thirteen: breaking the TLS and DTLS record protocols": exploit these timings; steal plaintext.

Some in
flow fro
timings:
constant
(on mos

What if
software
instructi
see anyt

c denials
exist.

Alice and Bob
the attacks.

rt on the
e Advanced
ard (AES)":
e against
o ensure that
nd decryption
the same
... Table lookup:
timing attacks."

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

2013 AlFardan–Paterson "Lucky Thirteen: breaking the TLS and DTLS record protocols": exploit these timings; steal plaintext.

Some instructions
flow from their inp
timings: e.g., logic
constant-distance
(on most CPUs), a

What if Alice and
software built sole
instructions? Yike
see anything from

Bob
s.

ed
":

that
tion

ookup:
acks."

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

2013 AlFardan–Paterson "Lucky Thirteen: breaking the TLS and DTLS record protocols": exploit these timings; steal plaintext.

Some instructions have no d
flow from their inputs to CP
timings: e.g., logic instructio
constant-distance shifts, mu
(on most CPUs), add, subtr

What if Alice and Bob use c
software built solely from th
instructions? Yikes: we won
see anything from timings!

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

2013 AlFardan–Paterson "Lucky Thirteen: breaking the TLS and DTLS record protocols": exploit these timings; steal plaintext.

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

2008 RFC 5246 "The Transport Layer Security (TLS) Protocol, Version 1.2": "This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal."

2013 AlFardan–Paterson "Lucky Thirteen: breaking the TLS and DTLS record protocols": exploit these timings; steal plaintext.

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

Try to scare implementors away from constant-time software.
e.g. "It will be too slow."
"It's too hard to write."

FC 5246 "The Transport
ecurity (TLS) Protocol,
1.2": "This leaves a
ming channel, since MAC
ance depends to some
n the size of the data
t, but it is not believed to
enough to be exploitable,
he large block size of
MACs and the small size
ming signal."

Fardan–Paterson "Lucky
: breaking the TLS and
ecord protocols": exploit
mings; steal plaintext.

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

Try to scare implementors away from constant-time software.
e.g. "It will be too slow."
"It's too hard to write."

Fund va
maybe w
that ma
for resea
but that
with our

The Transport
LS) Protocol,
is leaves a
nel, since MAC
nds to some
of the data
s not believed to
o be exploitable,
lock size of
d the small size
al."

terson "Lucky
g the TLS and
ocols": exploit
al plaintext.

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

Try to scare implementors away from constant-time software.
e.g. "It will be too slow."
"It's too hard to write."

Fund variable-time
maybe with "coun
that make the tim
for researchers to
but that are still b
with our computer

port
col,
n
MAC
ne
ta
ved to
itable,
of
l size

ucky
and
ploit
t.

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

Try to scare implementors away from constant-time software.
e.g. "It will be too slow."
"It's too hard to write."

Fund variable-time software, maybe with "countermeasur
that make the timings diffic
for researchers to analyze
but that are still breakable
with our computer resources

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

Try to scare implementors away from constant-time software.
e.g. "It will be too slow."
"It's too hard to write."

Fund variable-time software, maybe with "countermeasures" that make the timings difficult for researchers to analyze but that are still breakable with our computer resources.

Some instructions have no data flow from their inputs to CPU timings: e.g., logic instructions, constant-distance shifts, multiply (on most CPUs), add, subtract.

What if Alice and Bob use crypto software built solely from these instructions? Yikes: we won't see anything from timings!

Try to scare implementors away from constant-time software.
e.g. "It will be too slow."
"It's too hard to write."

Fund variable-time software, maybe with "countermeasures" that make the timings difficult for researchers to analyze but that are still breakable with our computer resources.

Continue expressing skepticism that constant time is needed. e.g. 2012 Mowery–Keelveedhi–Shacham "Are AES x86 cache timing attacks still feasible?", unfortunately shredded by 2014 Irazoqui–Inci–Eisenbarth–Sunar "Wait a minute! A fast, cross-VM attack on AES".

structions have no data
m their inputs to CPU
e.g., logic instructions,
t-distance shifts, multiply
t CPUs), add, subtract.

Alice and Bob use crypto
built solely from these
ons? Yikes: we won't
hing from timings!

care implementors away
nstant-time software.
will be too slow."
o hard to write."

Fund variable-time software,
maybe with "countermeasures"
that make the timings difficult
for researchers to analyze
but that are still breakable
with our computer resources.

Continue expressing skepticism
that constant time is needed.
e.g. 2012 Mowery–Keelveedhi–
Shacham "Are AES x86 cache
timing attacks still feasible?",
unfortunately shredded by 2014
Irazoqui–Inci–Eisenbarth–Sunar
"Wait a minute! A fast,
cross-VM attack on AES".

What if
use a dif
constant
are simp

Don't st
e.g. choo
not high
Watch o
standard

Discoura
Pretend
is a guar
while an
has ques

have no data
puts to CPU
c instructions,
shifts, multiply
add, subtract.

Bob use crypto
ly from these
s: we won't
timings!

mentors away
e software.
o slow."
write."

Fund variable-time software,
maybe with "countermeasures"
that make the timings difficult
for researchers to analyze
but that are still breakable
with our computer resources.

Continue expressing skepticism
that constant time is needed.
e.g. 2012 Mowery–Keelveedhi–
Shacham "Are AES x86 cache
timing attacks still feasible?",
unfortunately shredded by 2014
Irazoqui–Inci–Eisenbarth–Sunar
"Wait a minute! A fast,
cross-VM attack on AES".

What if terrorists
use a different cip
constant-time imp
are simple and fas

Don't standardize
e.g. choose Rijnda
not higher-security
Watch out for any
standardization eff

Discourage use of
Pretend that stand
is a guarantee of s
while anything nor
has questionable s

lata
PU
ons,
ltiply
act.

crypto
ese
n't

way
.

Fund variable-time software,
maybe with "countermeasures"
that make the timings difficult
for researchers to analyze
but that are still breakable
with our computer resources.

Continue expressing skepticism
that constant time is needed.
e.g. 2012 Mowery–Keelveedhi–
Shacham "Are AES x86 cache
timing attacks still feasible?",
unfortunately shredded by 2014
Irazoqui–Inci–Eisenbarth–Sunar
"Wait a minute! A fast,
cross-VM attack on AES".

What if terrorists Alice and
use a different cipher for wh
constant-time implementatio
are simple and fast? Yikes!

Don't standardize that ciphe
e.g. choose Rijndael as AES
not higher-security Serpent.
Watch out for any subseque
standardization efforts.

Discourage use of the cipher
Pretend that standardization
is a guarantee of security
while anything non-standard
has questionable security.

Fund variable-time software,
maybe with "countermeasures"
that make the timings difficult
for researchers to analyze
but that are still breakable
with our computer resources.

Continue expressing skepticism
that constant time is needed.
e.g. 2012 Mowery–Keelveedhi–
Shacham "Are AES x86 cache
timing attacks still feasible?",
unfortunately shredded by 2014
Irazoqui–Inci–Eisenbarth–Sunar
"Wait a minute! A fast,
cross-VM attack on AES".

What if terrorists Alice and Bob
use a different cipher for which
constant-time implementations
are simple and fast? Yikes!

Don't standardize that cipher.
e.g. choose Rijndael as AES,
not higher-security Serpent.
Watch out for any subsequent
standardization efforts.

Discourage use of the cipher.
Pretend that standardization
is a guarantee of security
while anything non-standard
has questionable security.

riable-time software,
with "countermeasures"
ke the timings difficult
rchers to analyze
are still breakable
computer resources.

e expressing skepticism
stant time is needed.
2 Mowery–Keelveedhi–
n "Are AES x86 cache
ttacks still feasible?",
ately shredded by 2014
–Inci–Eisenbarth–Sunar
minute! A fast,
M attack on AES".

What if terrorists Alice and Bob use a different cipher for which constant-time implementations are simple and fast? Yikes!

Don't standardize that cipher. e.g. choose Rijndael as AES, not higher-security Serpent. Watch out for any subsequent standardization efforts.

Discourage use of the cipher. Pretend that standardization is a guarantee of security while anything non-standard has questionable security.

Padding

1998 Bl
Decrypt
by obser
to $\approx 10^6$

SSL first
then che
(which r
Subsequ
more ser

Server re
pattern
pattern

e software,
termeasures"
ings difficult
analyze
reakable
r resources.

ng skepticism
e is needed.
–Keelveedhi–
ES x86 cache
l feasible?",
dded by 2014
nbarth–Sunar
A fast,
on AES".

What if terrorists Alice and Bob use a different cipher for which constant-time implementations are simple and fast? Yikes!

Don't standardize that cipher. e.g. choose Rijndael as AES, not higher-security Serpent. Watch out for any subsequent standardization efforts.

Discourage use of the cipher. Pretend that standardization is a guarantee of security while anything non-standard has questionable security.

Padding oracles

1998 Bleichenbach
Decrypt SSL RSA
by observing serve
to $\approx 10^6$ variants o

SSL first inverts R
then checks for "P
(which many forge
Subsequent proces
more serious integ

Server responses r
pattern of PKCS f
pattern reveals pla

es”
ult

s.

sm
d.
hi–
he
”,
’
014
nar

What if terrorists Alice and Bob use a different cipher for which constant-time implementations are simple and fast? Yikes!

Don’t standardize that cipher. e.g. choose Rijndael as AES, not higher-security Serpent. Watch out for any subsequent standardization efforts.

Discourage use of the cipher. Pretend that standardization is a guarantee of security while anything non-standard has questionable security.

## Padding oracles

1998 Bleichenbacher: Decrypt SSL RSA ciphertext by observing server response to $\approx 10^6$ variants of cipherte

SSL first inverts RSA, then checks for “PKCS padd (which many forgeries have) Subsequent processing appli more serious integrity checks

Server responses reveal pattern of PKCS forgeries; pattern reveals plaintext.

What if terrorists Alice and Bob use a different cipher for which constant-time implementations are simple and fast? Yikes!

Don't standardize that cipher.
e.g. choose Rijndael as AES,
not higher-security Serpent.
Watch out for any subsequent standardization efforts.

Discourage use of the cipher.
Pretend that standardization is a guarantee of security while anything non-standard has questionable security.

## Padding oracles

1998 Bleichenbacher:
Decrypt SSL RSA ciphertext
by observing server responses
to $\approx 10^6$ variants of ciphertext.

SSL first inverts RSA,
then checks for "PKCS padding"
(which many forgeries have).
Subsequent processing applies more serious integrity checks.

Server responses reveal
pattern of PKCS forgeries;
pattern reveals plaintext.

terrorists Alice and Bob
fferent cipher for which
t-time implementations
le and fast? Yikes!

andardize that cipher.
ose Rijndael as AES,
er-security Serpent.
ut for any subsequent
lization efforts.

age use of the cipher.
that standardization
rantee of security
ything non-standard
stionable security.

## Padding oracles

1998 Bleichenbacher:
Decrypt SSL RSA ciphertext
by observing server responses
to $\approx 10^6$ variants of ciphertext.

SSL first inverts RSA,
then checks for "PKCS padding"
(which many forgeries have).
Subsequent processing applies
more serious integrity checks.

Server responses reveal
pattern of PKCS forgeries;
pattern reveals plaintext.

Design c
so that f
as much

e.g. Des
and chec
checking
Broken I
such as

e.g. Des
IPsec op
Paterson
Degabrie

Alice and Bob
her for which
lementations
t? Yikes!

that cipher.
el as AES,
y Serpent.
y subsequent
forts.

the cipher.
dardization
security
n-standard
ecurity.

## Padding oracles

1998 Bleichenbacher:
Decrypt SSL RSA ciphertext
by observing server responses
to $\approx 10^6$ variants of ciphertext.

SSL first inverts RSA,
then checks for "PKCS padding"
(which many forgeries have).
Subsequent processing applies
more serious integrity checks.

Server responses reveal
pattern of PKCS forgeries;
pattern reveals plaintext.

Design cryptograp
so that forgeries a
as much processin

e.g. Design SSL to
and check padding
checking a serious
Broken by padding
such as BEAST ar

e.g. Design "encry
IPsec options. Bro
Paterson–Yau for
Degabriele–Paterso

Bob
ich
ons
er.
"
nt
r.
n

## Padding oracles

1998 Bleichenbacher:
Decrypt SSL RSA ciphertext
by observing server responses
to $\approx 10^6$ variants of ciphertext.

SSL first inverts RSA,
then checks for "PKCS padding"
(which many forgeries have).
Subsequent processing applies
more serious integrity checks.

Server responses reveal
pattern of PKCS forgeries;
pattern reveals plaintext.

Design cryptographic system
so that forgeries are sent thr
as much processing as possi

e.g. Design SSL to decrypt
and check padding before
checking a serious MAC.
Broken by padding-oracle at
such as BEAST and POODL

e.g. Design "encrypt-only"
IPsec options. Broken by 20
Paterson–Yau for Linux and
Degabriele–Paterson for RFC

## Padding oracles

1998 Bleichenbacher:
Decrypt SSL RSA ciphertext
by observing server responses
to $\approx 10^6$ variants of ciphertext.

SSL first inverts RSA,
then checks for "PKCS padding"
(which many forgeries have).
Subsequent processing applies
more serious integrity checks.

Server responses reveal
pattern of PKCS forgeries;
pattern reveals plaintext.

Design cryptographic systems
so that forgeries are sent through
as much processing as possible.

e.g. Design SSL to decrypt
and check padding before
checking a serious MAC.
Broken by padding-oracle attacks
such as BEAST and POODLE.

e.g. Design "encrypt-only"
IPsec options. Broken by 2006
Paterson–Yau for Linux and 2007
Degabriele–Paterson for RFCs.

## oracles

eichenbacher:
SSL RSA ciphertext
rving server responses
variants of ciphertext.

t inverts RSA,
ecks for "PKCS padding"
many forgeries have).
ent processing applies
rious integrity checks.

esponses reveal
of PKCS forgeries;
reveals plaintext.

Design cryptographic systems
so that forgeries are sent through
as much processing as possible.

e.g. Design SSL to decrypt
and check padding before
checking a serious MAC.
Broken by padding-oracle attacks
such as BEAST and POODLE.

e.g. Design "encrypt-only"
IPsec options. Broken by 2006
Paterson–Yau for Linux and 2007
Degabriele–Paterson for RFCs.

## Random

1995 Go
SSL key

2008 Be
OpenSS
$<$20 bits

2012 Le
Bos–Kle
Heninge
Halderm
keys for
The prin
randomn

ner:
ciphertext
r responses
of ciphertext.

RSA,
PKCS padding"
eries have).
ssing applies
rity checks.

eveal
forgeries;
hintext.

Design cryptographic systems
so that forgeries are sent through
as much processing as possible.

e.g. Design SSL to decrypt
and check padding before
checking a serious MAC.
Broken by padding-oracle attacks
such as BEAST and POODLE.

e.g. Design "encrypt-only"
IPsec options. Broken by 2006
Paterson–Yau for Linux and 2007
Degabriele–Paterson for RFCs.

<u>Randomness</u>

1995 Goldberg–Wa
SSL keys had <50

2008 Bello: Debia
OpenSSL keys for
<20 bits of entrop

2012 Lenstra–Hug
Bos–Kleinjung–Wa
Heninger–Durume
Halderman broke t
keys for 0.5% of a
The primes had so
randomness that t

t
es
xt.

ding”

.
es
s.

Design cryptographic systems
so that forgeries are sent through
as much processing as possible.

e.g. Design SSL to decrypt
and check padding before
checking a serious MAC.
Broken by padding-oracle attacks
such as BEAST and POODLE.

e.g. Design “encrypt-only”
IPsec options. Broken by 2006
Paterson–Yau for Linux and 2007
Degabriele–Paterson for RFCs.

Randomness

1995 Goldberg–Wagner: Ne
SSL keys had <50 bits of en

2008 Bello: Debian/Ubuntu
OpenSSL keys for years had
<20 bits of entropy.

2012 Lenstra–Hughes–Augie
Bos–Kleinjung–Wachter and
Heninger–Durumeric–Wustro
Halderman broke the RSA p
keys for 0.5% of all SSL serv
The primes had so little
randomness that they collide

Design cryptographic systems
so that forgeries are sent through
as much processing as possible.

e.g. Design SSL to decrypt
and check padding before
checking a serious MAC.
Broken by padding-oracle attacks
such as BEAST and POODLE.

e.g. Design "encrypt-only"
IPsec options. Broken by 2006
Paterson–Yau for Linux and 2007
Degabriele–Paterson for RFCs.

Randomness

1995 Goldberg–Wagner: Netscape
SSL keys had $<50$ bits of entropy.

2008 Bello: Debian/Ubuntu
OpenSSL keys for years had
$<20$ bits of entropy.

2012 Lenstra–Hughes–Augier–
Bos–Kleinjung–Wachter and 2012
Heninger–Durumeric–Wustrow–
Halderman broke the RSA public
keys for 0.5% of all SSL servers.
The primes had so little
randomness that they collided.

cryptographic systems
forgeries are sent through
processing as possible.

ign SSL to decrypt
ck padding before
g a serious MAC.
by padding-oracle attacks
BEAST and POODLE.

ign "encrypt-only"
tions. Broken by 2006
n–Yau for Linux and 2007
ele–Paterson for RFCs.

Randomness

1995 Goldberg–Wagner: Netscape SSL keys had <50 bits of entropy.

2008 Bello: Debian/Ubuntu OpenSSL keys for years had <20 bits of entropy.

2012 Lenstra–Hughes–Augier–Bos–Kleinjung–Wachter and 2012 Heninger–Durumeric–Wustrow–Halderman broke the RSA public keys for 0.5% of all SSL servers. The primes had so little randomness that they collided.

Make ra
extremel

Have ea
its own

Maintain
each app
build thi
from the
available

Pay peo
RNGs su
Claim "p

hic systems
re sent through
g as possible.

o decrypt
g before
MAC.
g-oracle attacks
nd POODLE.

ypt-only"
oken by 2006
Linux and 2007
on for RFCs.

Randomness

1995 Goldberg–Wagner: Netscape SSL keys had $<$50 bits of entropy.

2008 Bello: Debian/Ubuntu OpenSSL keys for years had $<$20 bits of entropy.

2012 Lenstra–Hughes–Augier–Bos–Kleinjung–Wachter and 2012 Heninger–Durumeric–Wustrow–Halderman broke the RSA public keys for 0.5% of all SSL servers. The primes had so little randomness that they collided.

Make randomness-
extremely difficult

Have each applica
its own RNG "for

Maintain separate
each application.
build this RNG in
from the inputs co
available to that a

Pay people to use
RNGs such as Dua

Claim "provable se

ns
rough
ble.

tacks
LE.

006
2007
Cs.

Randomness

1995 Goldberg–Wagner: Netscape SSL keys had <50 bits of entropy.

2008 Bello: Debian/Ubuntu OpenSSL keys for years had <20 bits of entropy.

2012 Lenstra–Hughes–Augier–Bos–Kleinjung–Wachter and 2012 Heninger–Durumeric–Wustrow–Halderman broke the RSA public keys for 0.5% of all SSL servers. The primes had so little randomness that they collided.

Make randomness-generation extremely difficult to audit.

Have each application maint its own RNG "for speed".

Maintain separate RNG *cod* each application. "For simpl build this RNG in ad-hoc wa from the inputs conveniently available to that application

Pay people to use backdoore RNGs such as Dual EC.

Claim "provable security".

## Randomness

1995 Goldberg–Wagner: Netscape SSL keys had $<50$ bits of entropy.

2008 Bello: Debian/Ubuntu OpenSSL keys for years had $<20$ bits of entropy.

2012 Lenstra–Hughes–Augier–Bos–Kleinjung–Wachter and 2012 Heninger–Durumeric–Wustrow–Halderman broke the RSA public keys for 0.5% of all SSL servers. The primes had so little randomness that they collided.

Make randomness-generation code extremely difficult to audit.

Have each application maintain its own RNG "for speed".

Maintain separate RNG *code* for each application. "For simplicity" build this RNG in ad-hoc ways from the inputs conveniently available to that application.

Pay people to use backdoored RNGs such as Dual EC.
Claim "provable security".

ness

oldberg–Wagner: Netscape
s had $<$50 bits of entropy.

llo: Debian/Ubuntu
L keys for years had
s of entropy.

nstra–Hughes–Augier–
injung–Wachter and 2012
r–Durumeric–Wustrow–
an broke the RSA public
0.5% of all SSL servers.
mes had so little
ness that they collided.

Make randomness-generation code extremely difficult to audit.

Have each application maintain its own RNG "for speed".

Maintain separate RNG *code* for each application. "For simplicity" build this RNG in ad-hoc ways from the inputs conveniently available to that application.

Pay people to use backdoored RNGs such as Dual EC.

Claim "provable security".

What if
merge a
into a ce

This poo
bad/faili
if there i
Merging
Yikes!

agner: Netscape
bits of entropy.

n/Ubuntu
years had
by.

ghes–Augier–
achter and 2012
ric–Wustrow–
the RSA public
ll SSL servers.
o little
hey collided.

Make randomness-generation code extremely difficult to audit.

Have each application maintain its own RNG "for speed".

Maintain separate RNG *code* for each application. "For simplicity" build this RNG in ad-hoc ways from the inputs conveniently available to that application.

Pay people to use backdoored RNGs such as Dual EC.

Claim "provable security".

What if the terror
merge all available
into a central entr

This pool can surv
bad/failing/malici
if there is one goo
Merging process is
Yikes!

tscape
ntropy.

er–
l 2012
ow–
ublic
vers.

ed.

Make randomness-generation code extremely difficult to audit.

Have each application maintain its own RNG "for speed".

Maintain separate RNG *code* for each application. "For simplicity" build this RNG in ad-hoc ways from the inputs conveniently available to that application.

Pay people to use backdoored RNGs such as Dual EC.
Claim "provable security".

What if the terrorists merge all available inputs into a central entropy pool?

This pool can survive many bad/failing/malicious inputs if there is one good input.
Merging process is auditable
Yikes!

Make randomness-generation code extremely difficult to audit.

Have each application maintain its own RNG "for speed".

Maintain separate RNG *code* for each application. "For simplicity" build this RNG in ad-hoc ways from the inputs conveniently available to that application.

Pay people to use backdoored RNGs such as Dual EC.
Claim "provable security".

What if the terrorists merge all available inputs into a central entropy pool?

This pool can survive many bad/failing/malicious inputs if there is one good input.
Merging process is auditable.
Yikes!

Make randomness-generation code extremely difficult to audit.

Have each application maintain its own RNG "for speed".

Maintain separate RNG *code* for each application. "For simplicity" build this RNG in ad-hoc ways from the inputs conveniently available to that application.

Pay people to use backdoored RNGs such as Dual EC.
Claim "provable security".

What if the terrorists merge all available inputs into a central entropy pool?

This pool can survive many bad/failing/malicious inputs if there is one good input. Merging process is auditable. Yikes!

Claim performance problems in writing to a central pool, reading from a central pool. Modify pool to make it unusable (`random`) or scary (`urandom`).

ndomness-generation code
ly difficult to audit.

ch application maintain
RNG "for speed".

n separate RNG *code* for
plication. "For simplicity"
s RNG in ad-hoc ways
e inputs conveniently
e to that application.

ple to use backdoored
uch as Dual EC.
provable security".

What if the terrorists
merge all available inputs
into a central entropy pool?

This pool can survive many
bad/failing/malicious inputs
if there is one good input.
Merging process is auditable.
Yikes!

Claim performance problems in
writing to a central pool,
reading from a central pool.
Modify pool to make it unusable
(`random`) or scary (`urandom`).

What if
RNG spe

Make it
to use ra
possible.
tests, en

e.g. DSA
new ran
$m$; could
$H(s, m)$.
user is g
which to
Bushing-
"PS3 ep

-generation code
to audit.

tion maintain
speed".

RNG *code* for
"For simplicity"
ad-hoc ways
onveniently
pplication.

backdoored
al EC.
ecurity".

What if the terrorists
merge all available inputs
into a central entropy pool?

This pool can survive many
bad/failing/malicious inputs
if there is one good input.
Merging process is auditable.
Yikes!

Claim performance problems in
writing to a central pool,
reading from a central pool.
Modify pool to make it unusable
(`random`) or scary (`urandom`).

What if the terrori
RNG speed isn't a

Make it an issue!
to use randomness
possible. This also
tests, encouraging

e.g. DSA and ECD
new random numb
$m$; could have rep
$H(s, m)$. 1992 Riv
user is given enou
which to hang him
Bushing–Marcan–S
"PS3 epic fail": P

n code

tain

*e* for
licity”
ays
y
.
ed

What if the terrorists
merge all available inputs
into a central entropy pool?

This pool can survive many
bad/failing/malicious inputs
if there is one good input.
Merging process is auditable.
Yikes!

Claim performance problems in
writing to a central pool,
reading from a central pool.
Modify pool to make it unusable
(`random`) or scary (`urandom`).

What if the terrorists realize
RNG speed isn't an issue?

Make it an issue! Design cry
to use randomness as often
possible. This also complica
tests, encouraging bugs.

e.g. DSA and ECDSA use a
new random number $k$ to si
$m$; could have replaced $k$ wi
$H(s, m)$. 1992 Rivest: “the
user is given enough rope wi
which to hang himself”. 201
Bushing–Marcan–Segher–Sv
“PS3 epic fail”: PS3 forgeri

What if the terrorists
merge all available inputs
into a central entropy pool?

This pool can survive many
bad/failing/malicious inputs
if there is one good input.
Merging process is auditable.
Yikes!

Claim performance problems in
writing to a central pool,
reading from a central pool.
Modify pool to make it unusable
(`random`) or scary (`urandom`).

What if the terrorists realize that
RNG speed isn't an issue?

Make it an issue! Design crypto
to use randomness as often as
possible. This also complicates
tests, encouraging bugs.

e.g. DSA and ECDSA use a
new random number $k$ to sign
$m$; could have replaced $k$ with
$H(s, m)$. 1992 Rivest: "the poor
user is given enough rope with
which to hang himself". 2010
Bushing–Marcan–Segher–Sven
"PS3 epic fail": PS3 forgeries.

the terrorists
ll available inputs
entral entropy pool?

ol can survive many
ing/malicious inputs
is one good input.
process is auditable.

erformance problems in
to a central pool,
from a central pool.
pool to make it unusable
n) or scary (`urandom`).

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure cry

2008 Ste
Appelba
Osvik–d
MD5 ⇒

ists
e inputs
opy pool?

vive many
ous inputs
d input.
s auditable.

e problems in
al pool,
ntral pool.
ake it unusable
(`urandom`).

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure crypto failure

2008 Stevens–Soti
Appelbaum–Lenstr
Osvik–de Weger e
MD5 $\Rightarrow$ rogue CA

e.

s in

sable

n).

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 $\Rightarrow$ rogue CA for TLS.

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 $\Rightarrow$ rogue CA for TLS.

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 $\Rightarrow$ rogue CA for TLS.

2012 Flame: new MD5 attack.

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 $\Rightarrow$ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years after the introduction of MD5, Preneel and Dobbertin were calling for MD5 to be scrapped.

What if the terrorists realize that RNG speed isn't an issue?

Make it an issue! Design crypto to use randomness as often as possible. This also complicates tests, encouraging bugs.

e.g. DSA and ECDSA use a new random number $k$ to sign $m$; could have replaced $k$ with $H(s, m)$. 1992 Rivest: "the poor user is given enough rope with which to hang himself". 2010 Bushing–Marcan–Segher–Sven "PS3 epic fail": PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 $\Rightarrow$ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years after the introduction of MD5, Preneel and Dobbertin were calling for MD5 to be scrapped.

We managed to keep MD5. How? Speed; standards; compatibility.

the terrorists realize that
eed isn't an issue?

an issue! Design crypto
andomness as often as
This also complicates
couraging bugs.

A and ECDSA use a
dom number $k$ to sign
d have replaced $k$ with
. 1992 Rivest: "the poor
given enough rope with
o hang himself". 2010
–Marcan–Segher–Sven
ic fail": PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–
Appelbaum–Lenstra–Molnar–
Osvik–de Weger exploited
MD5 $\Rightarrow$ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years
after the introduction of MD5,
Preneel and Dobbertin were
calling for MD5 to be scrapped.

We managed to keep MD5. How?
Speed; standards; compatibility.

2014: D
to "secu
e.g. `dns`
address

ists realize that
n issue?

Design crypto
s as often as
o complicates
bugs.

DSA use a
ber $k$ to sign
laced $k$ with
vest: "the poor
gh rope with
nself". 2010
Segher–Sven
PS3 forgeries.

Pure crypto failures

2008 Stevens–Sotirov–
Appelbaum–Lenstra–Molnar–
Osvik–de Weger exploited
MD5 $\Rightarrow$ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years
after the introduction of MD5,
Preneel and Dobbertin were
calling for MD5 to be scrapped.

We managed to keep MD5. How?
Speed; standards; compatibility.

2014: DNSSEC us
to "secure" IP add
e.g. `dnssec-depl`
address is signed b

e that

ypto
as
tes

gn
th
poor
ith
10
en
es.

## Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 ⇒ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years after the introduction of MD5, Preneel and Dobbertin were calling for MD5 to be scrapped.

We managed to keep MD5. How? Speed; standards; compatibility.

2014: DNSSEC uses RSA-1
to "secure" IP addresses.
e.g. `dnssec-deployment.o`
address is signed by RSA-10

## Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 $\Rightarrow$ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years after the introduction of MD5, Preneel and Dobbertin were calling for MD5 to be scrapped.

We managed to keep MD5. How? Speed; standards; compatibility.

2014: DNSSEC uses RSA-1024 to "secure" IP addresses.
e.g. `dnssec-deployment.org` address is signed by RSA-1024.

## Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 ⇒ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years after the introduction of MD5, Preneel and Dobbertin were calling for MD5 to be scrapped.

We managed to keep MD5. How? Speed; standards; compatibility.

2014: DNSSEC uses RSA-1024 to "secure" IP addresses. e.g. `dnssec-deployment.org` address is signed by RSA-1024.

Fact: Analyses in 2003 concluded that RSA-1024 was breakable; e.g., 2003 Shamir–Tromer estimated 1 year, $\approx 10^7$ USD.

## Pure crypto failures

2008 Stevens–Sotirov–Appelbaum–Lenstra–Molnar–Osvik–de Weger exploited MD5 ⇒ rogue CA for TLS.

2012 Flame: new MD5 attack.

Fact: By 1996, a few years after the introduction of MD5, Preneel and Dobbertin were calling for MD5 to be scrapped.

We managed to keep MD5. How? Speed; standards; compatibility.

2014: DNSSEC uses RSA-1024 to "secure" IP addresses. e.g. `dnssec-deployment.org` address is signed by RSA-1024.

Fact: Analyses in 2003 concluded that RSA-1024 was breakable; e.g., 2003 Shamir–Tromer estimated 1 year, $\approx 10^7$ USD.

DNSSEC's main excuse for sticking to RSA-1024: speed. "Tradeoff between the risk of key compromise and performance."

ypto failures

evens–Sotirov–
um–Lenstra–Molnar–
e Weger exploited
rogue CA for TLS.

ame: new MD5 attack.

y 1996, a few years
e introduction of MD5,
and Dobbertin were
or MD5 to be scrapped.

aged to keep MD5. How?
standards; compatibility.

2014: DNSSEC uses RSA-1024
to "secure" IP addresses.
e.g. `dnssec-deployment.org`
address is signed by RSA-1024.

Fact: Analyses in 2003 concluded
that RSA-1024 was breakable;
e.g., 2003 Shamir–Tromer
estimated 1 year, $\approx 10^7$ USD.

DNSSEC's main excuse
for sticking to RSA-1024: speed.
"Tradeoff between the risk of key
compromise and performance."

How to
that sec

Many te
incompe

es

rov–
ra–Molnar–
xploited
A for TLS.

MD5 attack.

few years
tion of MD5,
ertin were
o be scrapped.

eep MD5. How?
compatibility.

2014: DNSSEC uses RSA-1024
to "secure" IP addresses.
e.g. `dnssec-deployment.org`
address is signed by RSA-1024.

Fact: Analyses in 2003 concluded
that RSA-1024 was breakable;
e.g., 2003 Shamir–Tromer
estimated 1 year, $\approx 10^7$ USD.

DNSSEC's main excuse
for sticking to RSA-1024: speed.
"Tradeoff between the risk of key
compromise and performance."

How to convince t
that secure crypto

Many techniques:
incompetent bench

–

ck.

05,

ped.

How?

lity.

2014: DNSSEC uses RSA-1024
to "secure" IP addresses.
e.g. `dnssec-deployment.org`
address is signed by RSA-1024.

Fact: Analyses in 2003 concluded
that RSA-1024 was breakable;
e.g., 2003 Shamir–Tromer
estimated 1 year, $\approx 10^7$ USD.

DNSSEC's main excuse
for sticking to RSA-1024: speed.
"Tradeoff between the risk of key
compromise and performance."

How to convince terrorists
that secure crypto is too slo

Many techniques: obsolete
incompetent benchmarks, fr

2014: DNSSEC uses RSA-1024
to "secure" IP addresses.
e.g. `dnssec-deployment.org`
address is signed by RSA-1024.

Fact: Analyses in 2003 concluded
that RSA-1024 was breakable;
e.g., 2003 Shamir–Tromer
estimated 1 year, $\approx 10^7$ USD.

DNSSEC's main excuse
for sticking to RSA-1024: speed.
"Tradeoff between the risk of key
compromise and performance."

How to convince terrorists
that secure crypto is too slow?

Many techniques: obsolete data,
incompetent benchmarks, fraud.

2014: DNSSEC uses RSA-1024
to "secure" IP addresses.
e.g. `dnssec-deployment.org`
address is signed by RSA-1024.

Fact: Analyses in 2003 concluded
that RSA-1024 was breakable;
e.g., 2003 Shamir–Tromer
estimated 1 year, $\approx 10^7$ USD.

DNSSEC's main excuse
for sticking to RSA-1024: speed.
"Tradeoff between the risk of key
compromise and performance."

How to convince terrorists
that secure crypto is too slow?

Many techniques: obsolete data,
incompetent benchmarks, fraud.

Example:
"*PRESERVE contributes to the security and privacy of future vehicle-to-vehicle and vehicle-to-infrastructure communication systems by addressing critical issues like performance, scalability, and deployability of V2X security systems.*"
`preserve-project.eu`

NSSEC uses RSA-1024
re" IP addresses.
sec-deployment.org
is signed by RSA-1024.

nalyses in 2003 concluded
A-1024 was breakable;
03 Shamir–Tromer
ed 1 year, $\approx 10^7$ USD.

C's main excuse
ing to RSA-1024: speed.
ff between the risk of key
mise and performance."

How to convince terrorists
that secure crypto is too slow?

Many techniques: obsolete data,
incompetent benchmarks, fraud.

Example:
"*PRESERVE contributes to the security and privacy of future vehicle-to-vehicle and vehicle-to-infrastructure communication systems by addressing critical issues like performance, scalability, and deployability of V2X security systems.*"
preserve-project.eu

"*[In] mo*
*the pack*
*750 pac*
*maximu*
*goes we*
*(2,265 p*

*Processi*
*second a*
*ms can*
*hardwar*
*a Pentiu*
*needs ab*
*a verific*
*cryptogr*
*likely to*

ses RSA-1024
dresses.
oyment.org
by RSA-1024.

2003 concluded
as breakable;
-Tromer
$\approx 10^7$ USD.

xcuse
A-1024: speed.
the risk of key
erformance."

How to convince terrorists
that secure crypto is too slow?

Many techniques: obsolete data,
incompetent benchmarks, fraud.

Example:
"*PRESERVE contributes to the security and privacy of future vehicle-to-vehicle and vehicle-to-infrastructure communication systems by addressing critical issues like performance, scalability, and deployability of V2X security systems.*"
preserve-project.eu

"*[In] most driving*
*the packet rates d*
*750 packets per se*
*maximum highway*
*goes well beyond t*
*(2,265 packets per*

*Processing 1,000 p*
*second and proces*
*ms can hardly be*
*hardware. As disc*
*a Pentium D 3.4 G*
*needs about 5 tim*
*a verification . . . a*
*cryptographic co-p*
*likely to be necess*

024

rg
24.

cluded
le;

).

peed.
of key
ce."

How to convince terrorists
that secure crypto is too slow?

Many techniques: obsolete data,
incompetent benchmarks, fraud.

Example:
"*PRESERVE contributes to the security and privacy of future vehicle-to-vehicle and vehicle-to-infrastructure communication systems by addressing critical issues like performance, scalability, and deployability of V2X security systems.*"
preserve-project.eu

"*[In] most driving situations the packet rates do not exce 750 packets per second. On maximum highway scenario goes well beyond this value (2,265 packets per second).*

*Processing 1,000 packets pe second and processing each ms can hardly be met by cu hardware. As discussed in [3 a Pentium D 3.4 GHz proce needs about 5 times as long a verification . . . a dedicate cryptographic co-processor i likely to be necessary.*"

How to convince terrorists
that secure crypto is too slow?

Many techniques: obsolete data,
incompetent benchmarks, fraud.

Example:
"*PRESERVE contributes to the security and privacy of future vehicle-to-vehicle and vehicle-to-infrastructure communication systems by addressing critical issues like performance, scalability, and deployability of V2X security systems.*"
preserve-project.eu

"*[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary.*"

convince terrorists
ure crypto is too slow?

chniques: obsolete data,
tent benchmarks, fraud.

:
*ERVE contributes to the*
*and privacy of future*
*o-vehicle and vehicle-*
*structure communication*
*by addressing critical*
*ke performance, scalability,*
*loyability of V2X security*
"

ve-project.eu

"*[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary.*"

Compare
on 1GHz
5.48 cyc
2.30 cyc
for Salsa
498349
624846
for Curv

errorists
is too slow?

obsolete data,
hmarks, fraud.

*ributes to the*
*cy of future*
*and vehicle-*
*ommunication*
*sing critical*
*ance, scalability,*
*of V2X security*

ct.eu

*"[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary."*

Compare to "NEO
on 1GHz Cortex-A
5.48 cycles/byte (
2.30 cycles/byte (
for Salsa20, Poly1
498349 cycles (20
624846 cycles (16
for Curve25519 DH

w?

data,
aud.

*the*
*re*
*e-*
*ation*
*al*
*iability,*
*curity*

"*[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary.*"

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second
624846 cycles (1600/second
for Curve25519 DH, verify.

*"[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary."*

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second),
624846 cycles (1600/second)
for Curve25519 DH, verify.

*"[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary."*

Compare to "NEON crypto" on 1GHz Cortex-A8 core: 5.48 cycles/byte (1.4 Gbps), 2.30 cycles/byte (3.4 Gbps) for Salsa20, Poly1305. 498349 cycles (2000/second), 624846 cycles (1600/second) for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end smartphone core in 2010: e.g., Samsung Exynos 3110 (Galaxy S); TI OMAP3630 (Motorola Droid X); Apple A4 (iPad 1/iPhone 4).

*"[In] most driving situations . . . the packet rates do not exceed 750 packets per second. Only the maximum highway scenario . . . goes well beyond this value (2,265 packets per second). . . .*

*Processing 1,000 packets per second and processing each in 1 ms can hardly be met by current hardware. As discussed in [32], a Pentium D 3.4 GHz processor needs about 5 times as long for a verification . . . a dedicated cryptographic co-processor is likely to be necessary."*

Compare to "NEON crypto" on 1GHz Cortex-A8 core: 5.48 cycles/byte (1.4 Gbps), 2.30 cycles/byte (3.4 Gbps) for Salsa20, Poly1305. 498349 cycles (2000/second), 624846 cycles (1600/second) for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end smartphone core in 2010: e.g., Samsung Exynos 3110 (Galaxy S); TI OMAP3630 (Motorola Droid X); Apple A4 (iPad 1/iPhone 4).

2013: Allwinner A13, $5 in bulk.

*ost driving situations . . .*
*ket rates do not exceed*
*kets per second. Only the*
*m highway scenario . . .*
*ll beyond this value*
*packets per second). . . .*

*ing 1,000 packets per*
*and processing each in 1*
*hardly be met by current*
*e. As discussed in [32],*
*m D 3.4 GHz processor*
*bout 5 times as long for*
*ation . . . a dedicated*
*raphic co-processor is*
*be necessary.*"

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second),
624846 cycles (1600/second)
for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end
smartphone core in 2010: e.g.,
Samsung Exynos 3110 (Galaxy S);
TI OMAP3630 (Motorola Droid
X); Apple A4 (iPad 1/iPhone 4).

2013: Allwinner A13, $5 in bulk.

What if
hear abo
Yikes!

Similar t
Don't st
Discoura

*situations . . .*
*lo not exceed*
*econd. Only the*
*y scenario . . .*
*this value*
*r second). . . .*

*packets per*
*ssing each in 1*
*met by current*
*ussed in [32],*
*GHz processor*
*es as long for*
*a dedicated*
*processor is*
*ary.*"

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second),
624846 cycles (1600/second)
for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end
smartphone core in 2010: e.g.,
Samsung Exynos 3110 (Galaxy S);
TI OMAP3630 (Motorola Droid
X); Apple A4 (iPad 1/iPhone 4).

2013: Allwinner A13, $5 in bulk.

What if the terrori
hear about fast se
Yikes!

Similar to constan
Don't standardize
Discourage use of

. . .
*eed*
*ly the*
. . .
. . .
*er*
*in 1*
*rrent*
*32],*
*ssor*
*for*
*d*
*s*

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second),
624846 cycles (1600/second)
for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end
smartphone core in 2010: e.g.,
Samsung Exynos 3110 (Galaxy S);
TI OMAP3630 (Motorola Droid
X); Apple A4 (iPad 1/iPhone 4).

2013: Allwinner A13, $5 in bulk.

What if the terrorists
hear about fast secure crypt
Yikes!

Similar to constant-time sto
Don't standardize good cryp
Discourage use of good cryp

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second),
624846 cycles (1600/second)
for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end
smartphone core in 2010: e.g.,
Samsung Exynos 3110 (Galaxy S);
TI OMAP3630 (Motorola Droid
X); Apple A4 (iPad 1/iPhone 4).

2013: Allwinner A13, $5 in bulk.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

Compare to "NEON crypto"
on 1GHz Cortex-A8 core:
5.48 cycles/byte (1.4 Gbps),
2.30 cycles/byte (3.4 Gbps)
for Salsa20, Poly1305.
498349 cycles (2000/second),
624846 cycles (1600/second)
for Curve25519 DH, verify.

1GHz Cortex-A8 was high-end
smartphone core in 2010: e.g.,
Samsung Exynos 3110 (Galaxy S);
TI OMAP3630 (Motorola Droid
X); Apple A4 (iPad 1/iPhone 4).

2013: Allwinner A13, $5 in bulk.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise "cryptographic agility";
actually cryptographic fragility.
Pretend that this "agility"
justifies using breakable crypto.

e to “NEON crypto”
z Cortex-A8 core:
les/byte (1.4 Gbps),
les/byte (3.4 Gbps)
n20, Poly1305.
cycles (2000/second),
cycles (1600/second)
e25519 DH, verify.

ortex-A8 was high-end
one core in 2010: e.g.,
g Exynos 3110 (Galaxy S);
AP3630 (Motorola Droid
le A4 (iPad 1/iPhone 4).

llwinner A13, $5 in bulk.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don’t standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise “cryptographic agility”;
actually cryptographic fragility.
Pretend that this “agility”
justifies using breakable crypto.

Precomp

Try to b
into mar

e.g. Com

ON crypto"
A8 core:
1.4 Gbps),
3.4 Gbps)
305.
00/second),
00/second)
H, verify.

was high-end
n 2010: e.g.,
3110 (Galaxy S);
Motorola Droid
d 1/iPhone 4).

13, $5 in bulk.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise "cryptographic agility";
actually cryptographic fragility.
Pretend that this "agility"
justifies using breakable crypto.

Precomputed signa

Try to build crypto
into many layers o

e.g. Complicate th

),
)

nd
g.,
axy S);
roid
ne 4).

bulk.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise "cryptographic agility";
actually cryptographic fragility.
Pretend that this "agility"
justifies using breakable crypto.

Precomputed signatures

Try to build cryptographic fr
into many layers of the syste

e.g. Complicate the protocol

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise "cryptographic agility";
actually cryptographic fragility.
Pretend that this "agility"
justifies using breakable crypto.

## Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise "cryptographic agility";
actually cryptographic fragility.
Pretend that this "agility"
justifies using breakable crypto.

## Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

What if the terrorists
hear about fast secure crypto?
Yikes!

Similar to constant-time story.
Don't standardize good crypto.
Discourage use of good crypto.

If the good crypto persists,
try to bury it behind
a huge menu of bad options.
Advertise "cryptographic agility";
actually cryptographic fragility.
Pretend that this "agility"
justifies using breakable crypto.

## Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

the terrorists

out fast secure crypto?

to constant-time story.

andardize good crypto.

age use of good crypto.

ood crypto persists,

ury it behind

menu of bad options.

e "cryptographic agility";

cryptographic fragility.

that this "agility"

using breakable crypto.

Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

Solve "t

by preco

Insist th

allow pre

e.g. DNS

ists

cure crypto?

t-time story.

good crypto.

good crypto.

persists,

nd

ad options.

graphic agility";

phic fragility.

"agility"

kable crypto.

<u>Precomputed signatures</u>

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

Solve "the easy pr

by precomputing s

Insist that the pro

allow precomputat

e.g. DNSSEC.

o?

ry.
pto.
pto.

.
ility";
ity.

pto.

Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for s
e.g. DNSSEC.

## Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for speed".
e.g. DNSSEC.

## Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for speed".
e.g. DNSSEC.

The protocol has trouble handling
dynamically generated answers,
and unpredictable questions; also,
trouble guaranteeing freshness.
Deployment hits many snags.

## Precomputed signatures

Try to build cryptographic fragility
into many layers of the system.

e.g. Complicate the protocols.

Split cryptographic security
into "the easy problem"
of protecting integrity
and "the hard problem"
of protecting confidentiality.

e.g. argue against encrypted SNI
since DNS is unencrypted,
and argue against encrypted DNS
since SNI is unencrypted.

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for speed".
e.g. DNSSEC.

The protocol has trouble handling
dynamically generated answers,
and unpredictable questions; also,
trouble guaranteeing freshness.
Deployment hits many snags.

Argue that it's too early
to look at "the hard problem"
when most data is still unsigned.

puted signatures

uild cryptographic fragility
ny layers of the system.

nplicate the protocols.

yptographic security
e easy problem"
cting integrity
e hard problem"
cting confidentiality.

ue against encrypted SNI
NS is unencrypted,
ue against encrypted DNS
ll is unencrypted.

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for speed".
e.g. DNSSEC.

The protocol has trouble handling
dynamically generated answers,
and unpredictable questions; also,
trouble guaranteeing freshness.
Deployment hits many snags.

Argue that it's too early
to look at "the hard problem"
when most data is still unsigned.

More str

Divert "
and hum
into acti
threaten

Set up c
encrypti
that coll

More dis
breakabl

Declare
without

atures

ographic fragility
f the system.

e protocols.

c security
blem"
grity
blem"
identiality.

encrypted SNI
crypted,
encrypted DNS
crypted.

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for speed".
e.g. DNSSEC.

The protocol has trouble handling
dynamically generated answers,
and unpredictable questions; also,
trouble guaranteeing freshness.
Deployment hits many snags.

Argue that it's too early
to look at "the hard problem"
when most data is still unsigned.

More strategies

Divert "crypto" fu
and human resour
into activities that
threaten mass surv

Set up centralized
encrypting data to
that collaborate w

More distraction:
breakable by activ

Declare crypto suc
without encrypting

ragility
em.
ls.

SNI

DNS

Solve “the easy problem”
by precomputing signatures.
Insist that the protocol
allow precomputation “for speed”.
e.g. DNSSEC.

The protocol has trouble handling
dynamically generated answers,
and unpredictable questions; also,
trouble guaranteeing freshness.
Deployment hits many snags.

Argue that it’s too early
to look at “the hard problem”
when most data is still unsigned.

More strategies

Divert “crypto” funding
and human resources
into activities that don’t
threaten mass surveillance.

Set up centralized systems
encrypting data to companie
that collaborate with us.

More distraction: build syste
breakable by active attacks.

Declare crypto success
without encrypting the Inter

Solve "the easy problem"
by precomputing signatures.
Insist that the protocol
allow precomputation "for speed".
e.g. DNSSEC.

The protocol has trouble handling
dynamically generated answers,
and unpredictable questions; also,
trouble guaranteeing freshness.
Deployment hits many snags.

Argue that it's too early
to look at "the hard problem"
when most data is still unsigned.

## More strategies

Divert "crypto" funding
and human resources
into activities that don't
threaten mass surveillance.

Set up centralized systems
encrypting data to companies
that collaborate with us.

More distraction: build systems
breakable by active attacks.

Declare crypto success
without encrypting the Internet.