

Sichere Wegweiser im Internet: DNSSEC im praktischen Einsatz

- Einführung/Überblick
- DNSSEC
- Tools
- Mögliche Einsatzszenarien
- Stolpersteine / Praxis
- Links / Hilfen

- Stefan Neufeind
- Aus Neuss
- Tätig für SpeedPartner GmbH, ein Internet-Service-Provider (ISP)
(Consulting, Entwicklung, Administration)
 - Individuelle TYPO3-Entwicklungen
 - Hosting, Housing, Managed Services
 - Domains / Domain-Services
 - xDSL, IP-Upstream, IPv6, DNSSEC
- Aktive Mitarbeit im Community-Umfeld (PHP/PEAR, TYPO3, Linux)
- Freier Autor für z.B. t3n, iX, Internet World, ...

Das „Telefonbuch des Internet“:

- Anfrage stellen
- Antwort(en) erhalten

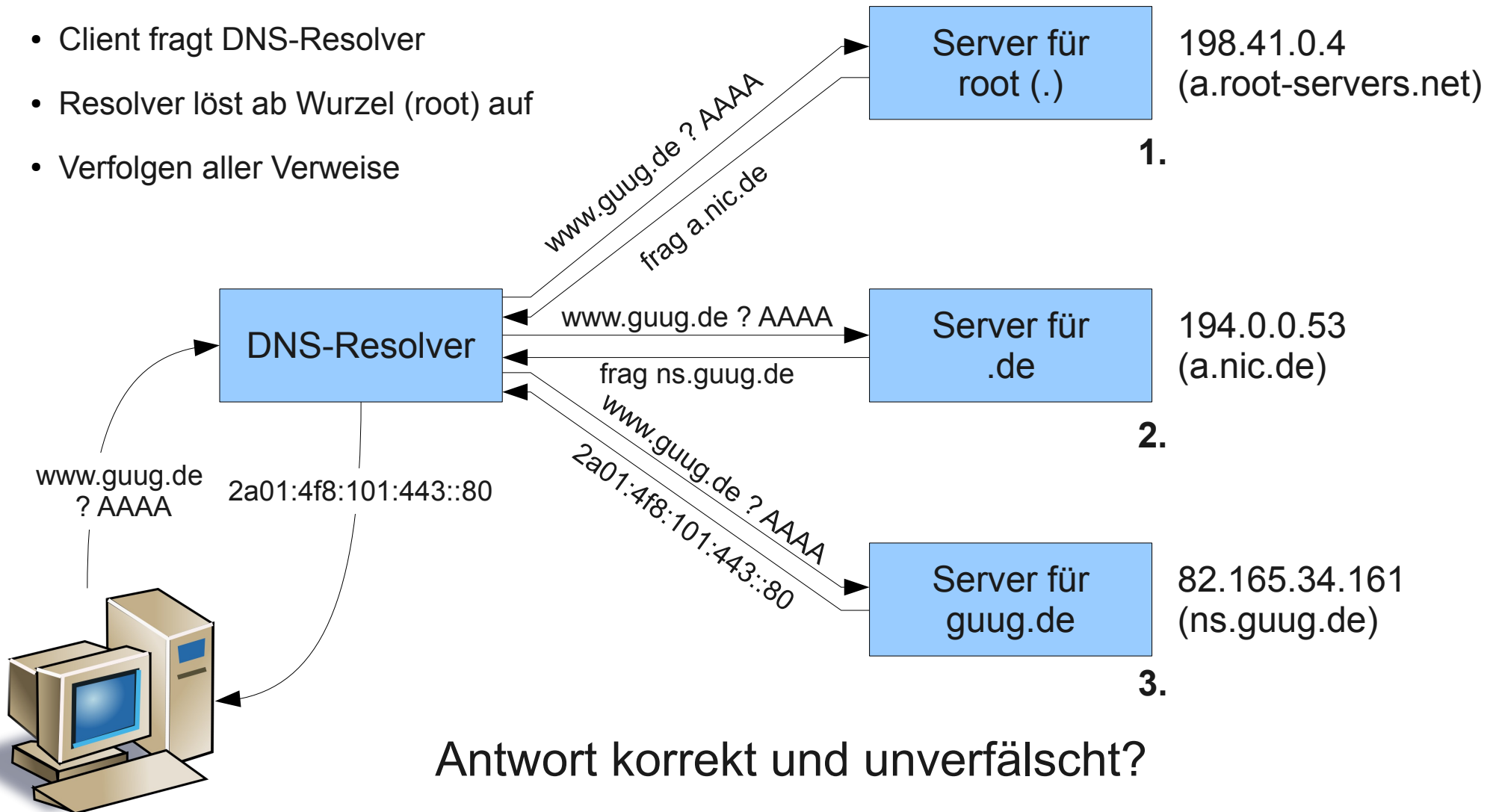
Vielseitig einsetzbar:

- Auflösung zu Adressen
 - IPv4 (A), IPv6 (AAAA)
- Auflösung zu Verweisen
 - Mail-Dienste (MX), Hostverweise auf andere Namen (CNAME)
- Vielzahl Zusatzeinträge
 - Verweise auf Dienste (SRV) – z.B. für XMPP
 - Kontakteinträge (NAPTR) – z.B. für SIP-Erreichbarkeiten
 - Beliebige Texteinträge (TXT)
 - Schlüssel, Geo-Positionen, ...



Auflösung mit Hierarchien und Verweisen:

- Client fragt DNS-Resolver
- Resolver löst ab Wurzel (root) auf
- Verfolgen aller Verweise



Korrektheit / Vertrauenswürdigkeit:

- DNS ist „unsicher“
 - 1 Paket (UDP), 1 Antwort
 - Leicht verfälschbar
- DNS-Spoofing: falsche Antworten einschleusen
- Cache-Poisoning: falsche Antworten im DNS-Cache platzieren

Lösungsansätze:

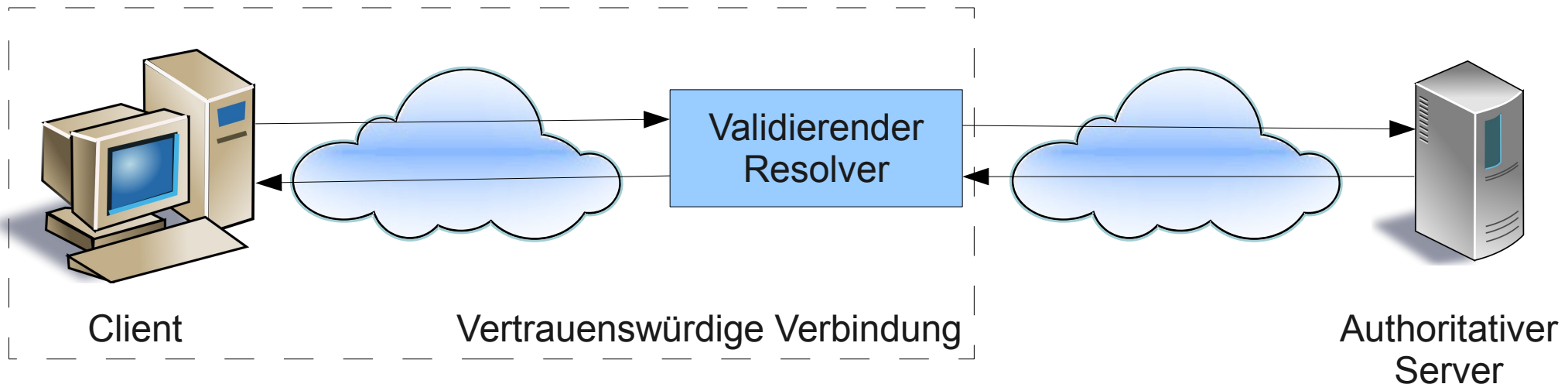
- Prüfung auf höherer Ebene
 - SSL: Name im Zertifikat korrekt? CA vertrauenswürdig? → siehe Comodo-Hack
 - Unmittelbare Prüfung überhaupt möglich?
 - Cache-Poisoning ggf. jedoch bereits erfolgt!
- Prüfung auf DNS-Ebene
 - Und dann evtl. zusätzlich (!) z.B. SSL-Zertifikate per DNSSEC validieren

DNSSEC für „vertrauenswürdigen DNS“:

- Public-key-Kryptographie / Digitale Signaturen
- Ermöglicht Prüfung der DNS-Antworten
 - Prüfung der Quelle
 - Prüfung der Integrität
- Prüfung entlang der DNS-Hierarchie-Kette
- Keine Verschlüsselung
 - DNS-Antworten les- und cachebar
 - Authentizität der Antworten - nicht Vertraulichkeit

DNSSEC für „vertrauenswürdigen DNS“:

- Erfordert keine Ende-zu-Ende-Verbindung
 - Stufenweise Auflösung/Validierung möglich
 - Cachen/Weitergabe von Antworten möglich
- Validierende Stelle muss vertrauenswürdig sein
 - Verbindung muss vertrauenswürdig / abgesichert sein
 - Optimal: Lokal validieren :-)



Arbeitsweise:

- Public-key-Kryptographie (asymmetrisch)
- Mindestens 1 Key, in der Regel 2
 - Zone-Signing-Key (ZSK) → für Signierung des Inhalts einer Zone (resource records)
 - Key-Signing-Key (KSK) → für Signierung des/der Zone-Signing-Key(s)
- Zusätzliche DNS-Records
 - DNSKEY: Public Key(s), für ZSK und KSK → für Prüfungen der Signaturen
 - DS: Hash des Public-Key (KSK) der nächsten Ebene → verweist auf gültigen DNSKEYs
 - RRSIG: Signatur für Antworten (resource records) → für Validierung der Antwort (RRs)
 - NSEC/NSEC3: Nachweis für Nicht-Existenz von Einträgen
- Auch Nicht-Existenz von Records (NXDOMAIN) muss vertrauenswürdig prüfbar sein
 - Zu vermeiden: Abzählbarkeit der Zone → Vertrauliche Information!
Möglich per NSEC3 statt NSEC

Arbeitsweise:

- Beispiel DNS-Records einer Zone:

Ressource-Records:

```
six53.net.      2811  IN   A     91.184.36.210
```

Signatur für Ressource-Records:

```
six53.net.      2811 IN RRSIG A 8 2 3600 20110331000000 (
 20110317000000 56311 six53.net.
 FiGGxP++9EZctArEFzrnf3YIFLQJNBO2TVLUJJnqPrhF
 SqHyxnPuLgFrwYMSk1qn1AZk9+bXaONP9s6uQ7m4AkEN
 bej4A6lxGttoNFNaPX8Nm/y7Jg+jX9Ux7c/Bprq0xNln
 dhuGN2hWaAaqQLKR30X1ld0v5GTgJHxsY8oUdvl= )
```

8 = Algorithmus:
RSASHA256

2 = Digest: SHA-2

3600 = TTL

20110331000000 =
Ablauf Gültigkeit

20110317000000 =
Beginn Gültigkeit

56311 = ID für Key

Public-Keys:

```
six53.net.      2101 IN DNSKEY 256 3 8 (
 AwEAAbxzsCoIXMZ6mybrLYMsO+4uOxMESrnS0Gem61fM
 qCd62i+PQxsu5Yi9Qq+ZJ35Uc3D7UMOGkY83W2SfXOlf
 bFNOF3v8gQMZtmDD7D3X7ubQZ3gTKGBEgbP3eXlln9vS
 MBiHY8XUE4Wml8Ku6ONPckbSS0xq59Wt7FCAV50+OX0/
 ) ; key id = 56311
```

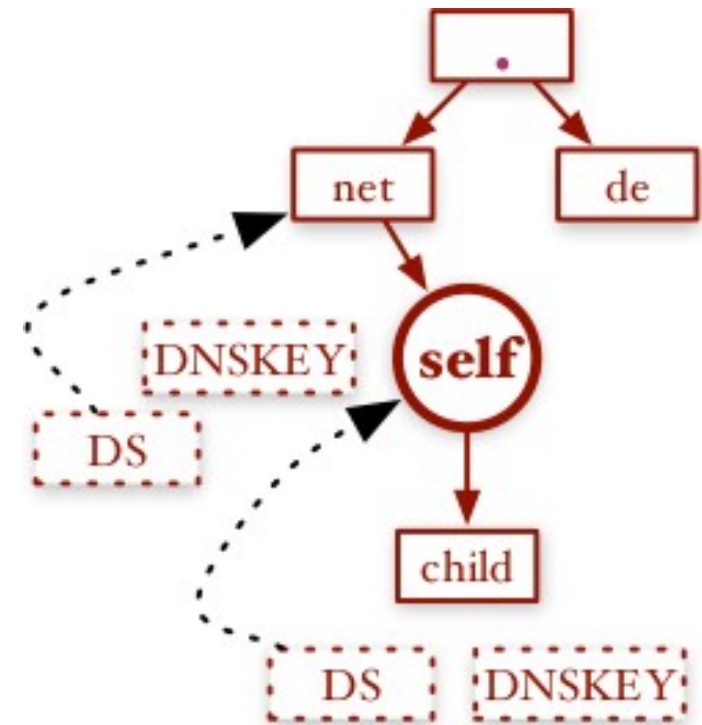
256 = Zone Signing
Key (257 für KSK)

3 = Protokoll:
DNSSEC

8 = Algorithmus:
RSASHA256

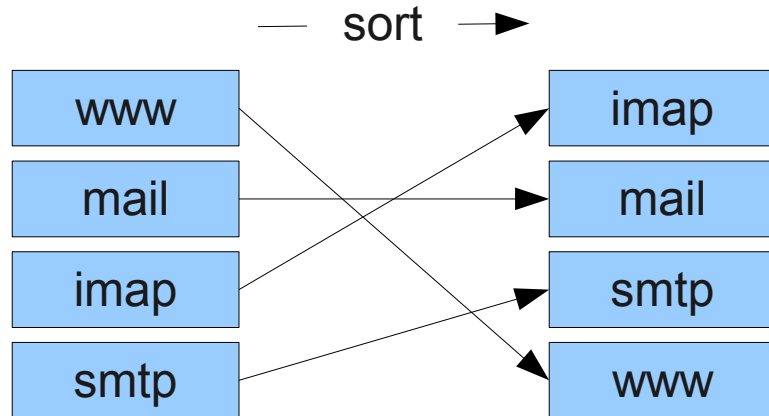
Hierarchische Beziehung/Validierung der Signaturen:

- Public-Key der Wurzel muss bekannt sein („trust-anchor“)
- Root-Zone enthält folgende Einträge:
 - NS-Einträge für .net-Nameserver (Verweis)
 - DNSKEY (Public-Key) der Root-Zone
 - Signatur (RRSIG) über NS-Einträge
 - Ein/mehrere DS-Einträge als Hash über Public-Key der .net-Zone
- .net-Zone enthält
 - NS-Einträge für six53.net-Nameserver (Verweis)
 - DNSKEY (Public-Key) der .net-Zone
 - Signatur (RRSIG) über NS-Einträge
 - Ein/mehrere DS-Einträge als Hash über Public-Key der six53.net-Zone

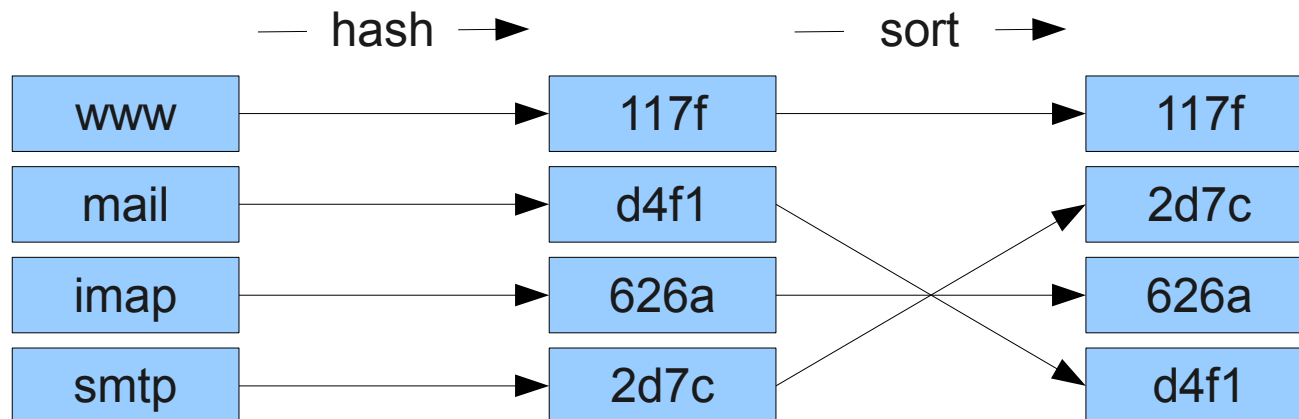


Nicht-Existenz von Records (NXDOMAIN):

- NSEC: Elemente abzählbar (Datenschutz-Problem)



- NSEC3: Abzählbarkeit durch vorheriges Hashing vermieden



Nicht-Existenz von Records (NXDOMAIN):

- NSEC: Elemente abzählbar (Datenschutz-Problem)

Frage: Existiert „ldap.example.com“?

Antwort: zwischen „imap“ und „mail“ sind keine Einträge vorhanden.

```
$ dig +dnssec ldap.example.com  
imap.example.com. 7200 IN NSEC mail.example.com. A RRSIG NSEC
```

- NSEC3: Abzählbarkeit durch vorheriges Hashing vermieden

Hash $H(\text{„ldap“})$ ist „de16“

Frage: Existiert „de16“?

Antwort: zwischen „626a“ und „d4f1“ sind keine Einträge vorhanden.

```
$ dig +dnssec ldap.example.com  
626A.example.com.7200 IN NSEC3 1 0 10 5AD4B3 D4F1 A RRSIG
```

Optional: Verwendung mehrerer Keys / Hashes:

- Verwendung von separatem Zone Signing Key (ZSK) um diesen z.B. regelmäßig einfach wechseln zu können
- Wechsel des Key Signing Key (KSK) erfordert Aktualisierung der Eltern-Zone
- Mehrere Hashes mit unterschiedlichen Algorithmen / Digests möglich für Verbesserung der Kompatibilität (z.B. DS mit SHA1 und SHA256)
- Mehrere Schlüssel während Schlüsselwechsel („Key rollover“) benötigt
 - Neue Schlüssel hinzufügen
 - Ausreichende Propagierung neuer Einträge abwarten (Ablauf Cache über die Hierarchie-Ebenen hinweg)
 - Alte Keys entfernen

Verbreitung von DNSSEC bei TLDs:

- Root seit 15.07.2010 signiert
- Root-Signaturen in allen aktuellen Resolvern mitgeliefert
- 69 TLDs derzeit bereits signiert
(siehe Reports: <http://fupps.com/dnssec/gdr/>, http://stats.research.icann.org/dns/tld_report/)
- TLDs wie .at oder .de derzeit mit DNSSEC Testbed
- .de übernimmt DNSSEC ab 31.5.2011 in Produktion

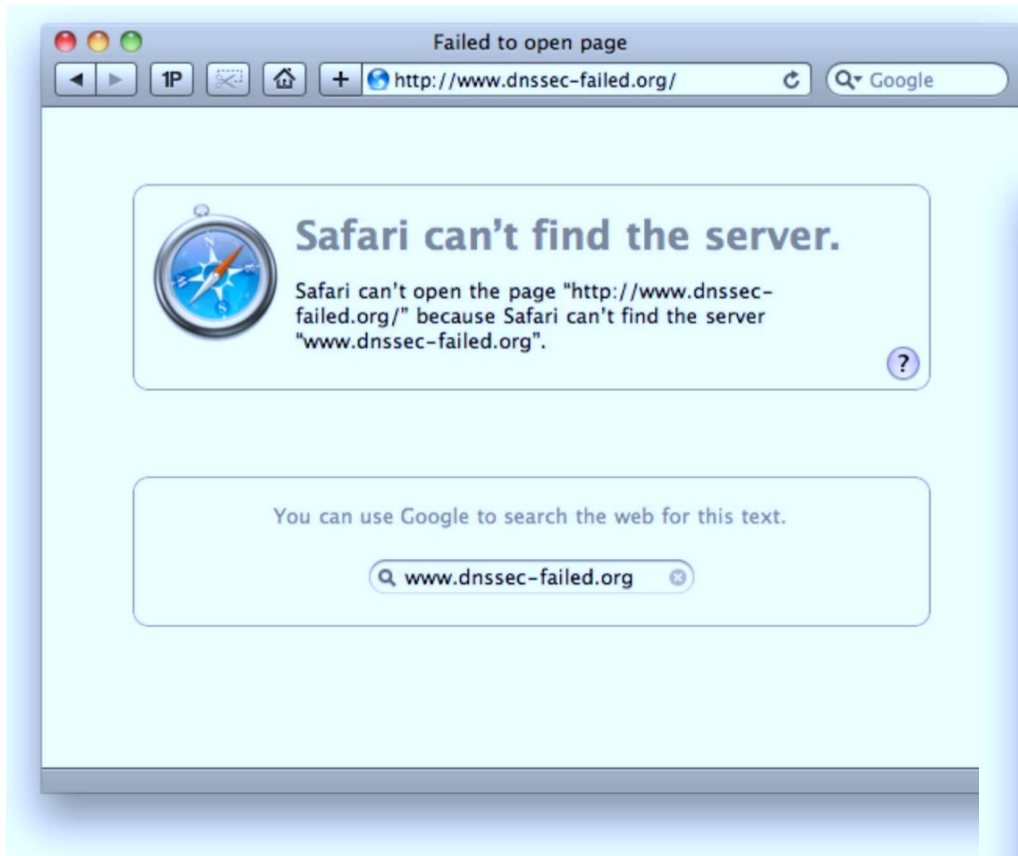
DLV als Alternative für die Übergangszeit:

- „Domain lookaside validation“ (DLV) des ISC (<http://dlv.isc.org/>) als zusätzliche Wurzel nutzbar
- Einträge kostenfrei möglich
- „Authentifizierung“ durch Hinzufügen spezieller Einträge in der eigenen Zone

Technik für Validierung:

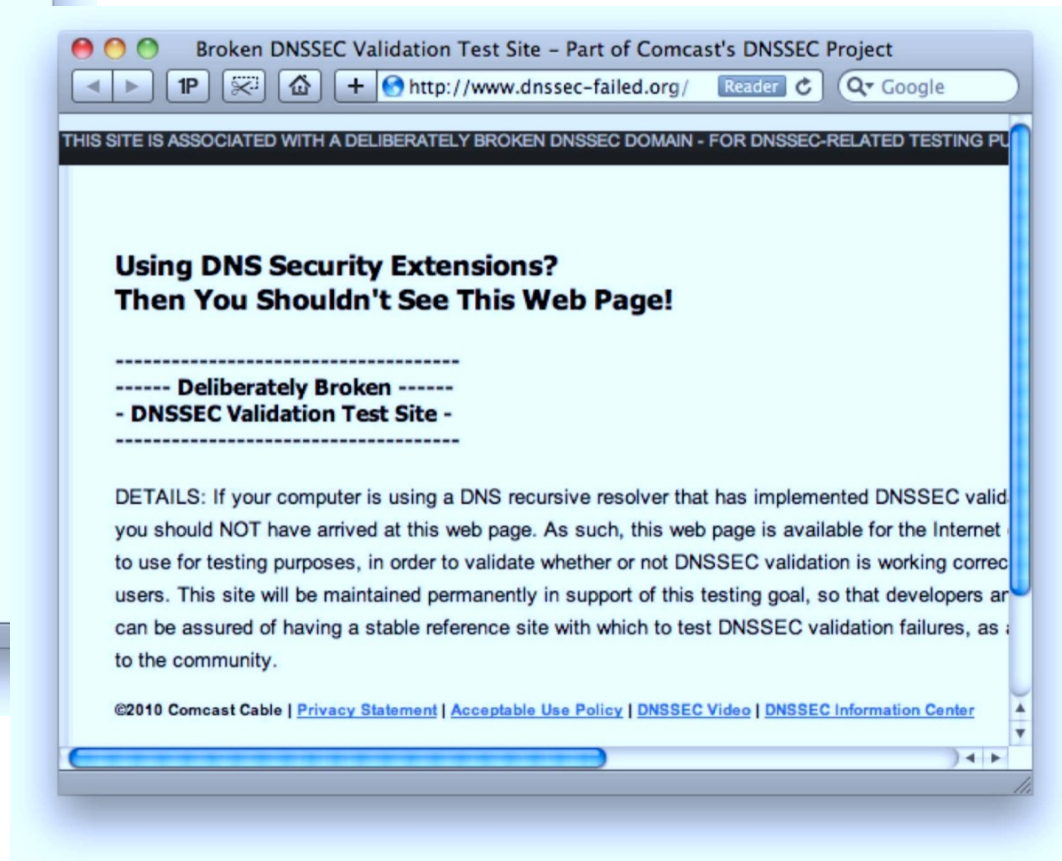
- EDNS-Unterstützung benötigt
 - Existierte bereits vor DNSSEC-Einführung als Standard (EDNS0 von 1999)
 - DNS: max. 512 Byte pro UDP-Antwort
- EDNS: größere UDP-Antworten, bis 4096 Byte
- Große Antworten (evtl. auch per TCP) auch bisher schon für große Zonen benötigt:
z.B. 20x AAAA-Einträgen (20x 128Bit) passen nicht in 512 Byte
 - DNSSEC-Antwort benötigt mit allen Inhalten z.B. leicht 1500/1600 Byte!
 - Ermöglicht zusätzliche Flags bei Abfragen/Antworten (für DNSSEC benötigt)
- Nicht-DNSSEC-fähige Software kann von validierendem Resolver im Fehlerfall trotzdem „SERVFAIL“ als Antwort erhalten
- DNSSEC-fähige Software kann zusätzliche Möglichkeiten nutzen
 - Explizite Anforderung von Signaturprüfung durch den Resolver;
Rückmeldung ob geprüft wurde
 - Explizite Anforderung von ungeprüften Antworten

Validierung:



Mit DNSSEC-Prüfung
(absichtlich fehlgeschlagen!)

Am Beispiel: www.dnssec-failed.org



Ohne DNSSEC-Prüfung

Validierung:

- Fehlschlagende, validierende Abfrage:

```
$ dig +dnssec www.dnssec-failed.org
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL
;; flags: qr rd ra; QUERY: 1, ANSWER: 0
```

- Abfrage ohne Prüfung („check disabled“):

```
$ dig +cd +dnssec www.dnssec-failed.org
;; ->>HEADER<<- opcode: QUERY, status: NOERROR
;; flags: qr rd ra cd; QUERY: 1, ANSWER: 2
;; ANSWER SECTION:
www.dnssec-failed.org. 5620 IN A 68.87.64.48
```

- Erfolgreiche Abfrage:

```
$ dig +dnssec www.six53.net
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25913
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 5, ADDITIONAL: 13
;; ANSWER SECTION:
www.six53.net.      3600  IN    A     91.184.36.210
www.six53.net.      3600  IN    RRSIG A 8 3 3600 20110331000000 20110317000000
56311 six53.net. FKFMYF2BJRhCGqLjFjFeqxDaMBGdF/1yh7Y8kFBbjbq68mKldvQINBbY
S+X3RJn2LD7JbHRZ/qcFzTfAkHutR9RiPD59PP/5oQmU/zR/lg1layVWNPp0zNotNVLZ[...]
```

Validierung:

- Überprüfen der Zertifikatskette:
 - (theoretisch) auch per dig möglich mit „+sigchase“
 - Mittel der Wahl „drill“ (basierend auf Idns)
 - Von Zone zum root: „sigchase“-Modus

Verweise auf verwendete
Schlüssel (DNSKEYs)

256=ZSK, 257=KSK

```
$ drill -D -S -k root.key six53.net
DNSSEC Trust tree:
six53.net. (A)
|---six53.net. (DNSKEY keytag: 56311 alg: 8 flags: 256)
|---six53.net. (DNSKEY keytag: 29596 alg: 8 flags: 257)
|---six53.net. (DS keytag: 29596 digest type: 1)
|---net. (DNSKEY keytag: 3980 alg: 8 flags: 256)
|---net. (DNSKEY keytag: 35886 alg: 8 flags: 257)
|---net. (DS keytag: 35886 digest type: 2)
|---. (DNSKEY keytag: 21639 alg: 8 flags: 256)
|---. (DNSKEY keytag: 19036 alg: 8 flags: 257)
```

Root-Key separat
benötigt

- Vom root zur Zone: „trace“-Modus (umfangreiche Ausgaben aller Keys!)

```
$ drill -D -T -k root.key six53.net
```

DNSSEC-Server:

- Resolver:
 - Unbound
 - BIND
- Authoritative Server:
 - NSD
 - BIND
 - PowerDNS 3.0 (noch nicht erschienen)

Recursor mit Unbound:

- root-Key benötigt (bei vielen Distributionen bereits im Paket enthalten)
 - unbound-anchor pflegt root.key bei Aktualisierungen (ggf. Cronjob einrichten)

```
$ unbound-anchor -a root.key
```

- DNSSEC-Validierung aktivieren (unbound.conf)

```
server:  
  auto-trust-anchor-file: "root.key"  
  dlz-anchor-file: "dlz.key"
```

- optional: Zusätzliche Datei mit DNSKEY/DS-Einträgen für eine „island of trust“

```
trust-anchor-file: "my.keys"
```

- optional: unbound für Anfrage gegen nicht-delegierte Zone konfigurieren

```
stub-zone:  
  name: "six53.net"  
  stub-host: localhost  
  stub-addr: 127.0.0.1
```

Recursor mit Bind:

- root-Key downloaden (Achtung! Prüfung des Schlüssels per DNSSEC o.ä. notwendig!)

```
$ dig . dnskey
```

- DS (Signatur) hieraus erstellen

```
/usr/local/sbin/dnssec-dsfromkey -2 -f /path/to/keyfile .
```

- Als Key für root ergänzen und DNSSEC aktivieren (named.conf)

```
managed-keys {  
    "." initial-key 257 3 8 "AwEAAagAIKIVZrp...";  
};  
  
options {  
    dnssec-enable yes;  
    dnssec-validation yes;  
    dnssec-lookaside auto; // DLV  
}
```

- Bind ab 9.8.0 liefert root.key mit; „dnssec-enable“ damit ausreichend

Authoritativer Server mit Bind / NSD:

- Key generieren

- Ohne -f um einen ZSK zu erzeugen

```
$ dnssec-keygen -a algo -b bits -f ksk example.com
```

- Zone signieren („BIND smart-signing“)

- Erfordert Cronjob zur Aktualisierung
- -o für Zone Origin; optional: -f für Ausgabe-Datei

```
$ dnssec-signzone -S -o example.com zone.db
```

- BIND: DNSSEC aktivieren (named.conf)

```
options {  
    dnssec-enable yes;  
}
```

- NSD: Zone hinzufügen; signierte Zonen automatisch erkannt

```
zone:  
    name: "example.com"  
    zonefile: "zone.db.signed"
```

Authoritativer Server mit Bind / NSD:

- Alternativ: BIND auto-signing
 - Keys hier in „mykeydir“ erwartet
 - Signaturen werden automatisch erneuert etc.

```
zone "example.com" in {  
  type master;  
  key-directory "mykeydir";  
  update-policy local;  
  auto-dnssec maintain;  
  sig-validity-interval 30; // days  
  file "example.com";  
};
```

- Initiale Signierung durchführen

```
$ rndc signzone example.com
```


Dynamische Updates mit Bind:

- per nsupdate (RFC 2136, von April 1997)
- Abgesichert per SIG(0) mit Public-Private-Key

- SIG(0) Schlüsselpaar erzeugen:

```
$ dnssec-keygen -C -a algo -b bits -n HOST -T KEY my.name.
```

- Public-Key zum DNS hinzufügen:

```
my.name. IN KEY 512 3 3 CLb...gaNM
```

- Update-Policy definieren:

```
update-policy {  
  grant local-ddns zonesub ANY;  
  grant my.name. zonesub A AAAA MX TXT;  
};
```

- nsupdate durchführen:

```
$ nsupdate -l  
$ nsupdate -k Kmy.name*.private
```

Nsupdate-Kommandos (Auswahl):

```
server addr [port]  
zone zonename  
prereq nxdomain domain  
update delete name [ttl] [type [data]]  
update add domain ttl type data  
show  
send  
answer
```

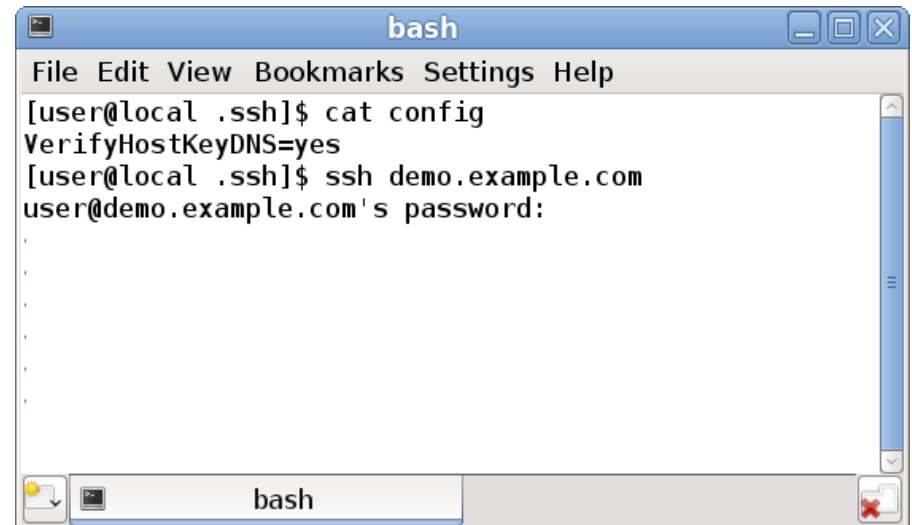
Validierung von SSH-Fingerprints mit OpenSSH:

- Für offiziellen OpenSSH nur über Patches verfügbar
- z.B. Fedora liefert standardmäßig einen angepassten OpenSSH-Client aus
- Aktivierung per „VerifyHostKeyDNS“-Parameter
- Ermöglicht SSH-Fingerprint-Prüfung als zusätzliches Entscheidungskriterium (Modus „Ask“) oder erlaubt automatisches akzeptieren von validen Fingerprints (Modus „Yes“)
- Beispiel:
demo.example.com. 300 IN SSHFP 1 1 ACC2E1E597682DE96AFAEC2D0C6E8D7E9625B7A0

Ohne SSHFP/VerifyHostKeyDNS:

```
$ ssh demo.example.com
The authenticity of host 'demo.example.com
(10.0.0.2)' can't be established.
RSA key fingerprint is
03:c0:1a:bd:5f:cd:2c:e1:e6:91:b7:58:80:d7:0f:d9.
No matching host key fingerprint found in DNS.
Are you sure you want to continue connecting
(yes/no)?
```

Mit SSHFP/VerifyHostKeyDNS:



```
bash
File Edit View Bookmarks Settings Help
[user@local .ssh]$ cat config
VerifyHostKeyDNS=yes
[user@local .ssh]$ ssh demo.example.com
user@demo.example.com's password:
```

DNSSEC-Validator für Firefox:

- <http://www.dnssec-validator.cz/>
- Zeigt Status der Prüfung in der URL-Zeile
- Verwendet Idns-Library für Prüfung

SSL-Prüfung per DNSSEC:

- <http://mens.de/:/bo>
- Aktuell Implementierungen im Status „proof-of-concept“
- Arbeitet unter Linux per LD_PRELOAD / OpenSSL
- Funktioniert mit Vielzahl von SSL-Anwendungen



Komplexität von DNS nicht unterschätzen:

- Timestamps in RRSIGs
 - Angegeben in UTC – nicht lokaler Zeit
- Konsistente Zoneneinträge
- Konsistente Daten auf Master-/Slave-Server
- Absicherung Updates an Signatur-Server (z.B. mit SIG(0) / TSIG)
- Keyrollover
 - Zone Signing Key (ZSK): erst neue Keys verteilen, dann neue Signaturen
 - Key Signing Key (KSK): Aktualisierung bei Parent-Zone (z.B. Registry) erforderlich
- DNSSEC-Prüfung schlägt fehl?
 - Dienste nicht erreichbar
 - Haltezeit fehlerhafte Einträge in DNS-Caches
- Monitoring (Verfügbarkeit, Signaturen, Updates, ...)

Die Leistungen von six53.net rund um DNS und Domains:

- Aktuell: Dual-Stack (IPv4/IPv6), DNSSEC, Anycast
- Zuverlässig: DNSSEC, Redundanzen, sicherer Betrieb
- Verfügbar:
 - Bereits „heute“ und ohne große Vorarbeiten nutzbar
 - Professioneller 24/7-Betrieb
 - Performant, Verteilter Betrieb per Anycast + Unicast
- Erprobt:
 - Intensiv getestet, Erfahrung mit Domains (direkter Registrar, Whitelabel-Domainlösung)
- Flexibel:
 - Per Web, API, dynamic updates, hidden primary; mit z.B. TSIG und SIG(0)
 - Integration in bestehende Landschaften möglich
- Professionelle Lösung: Schulung, Consulting, Individuelle Lösungen



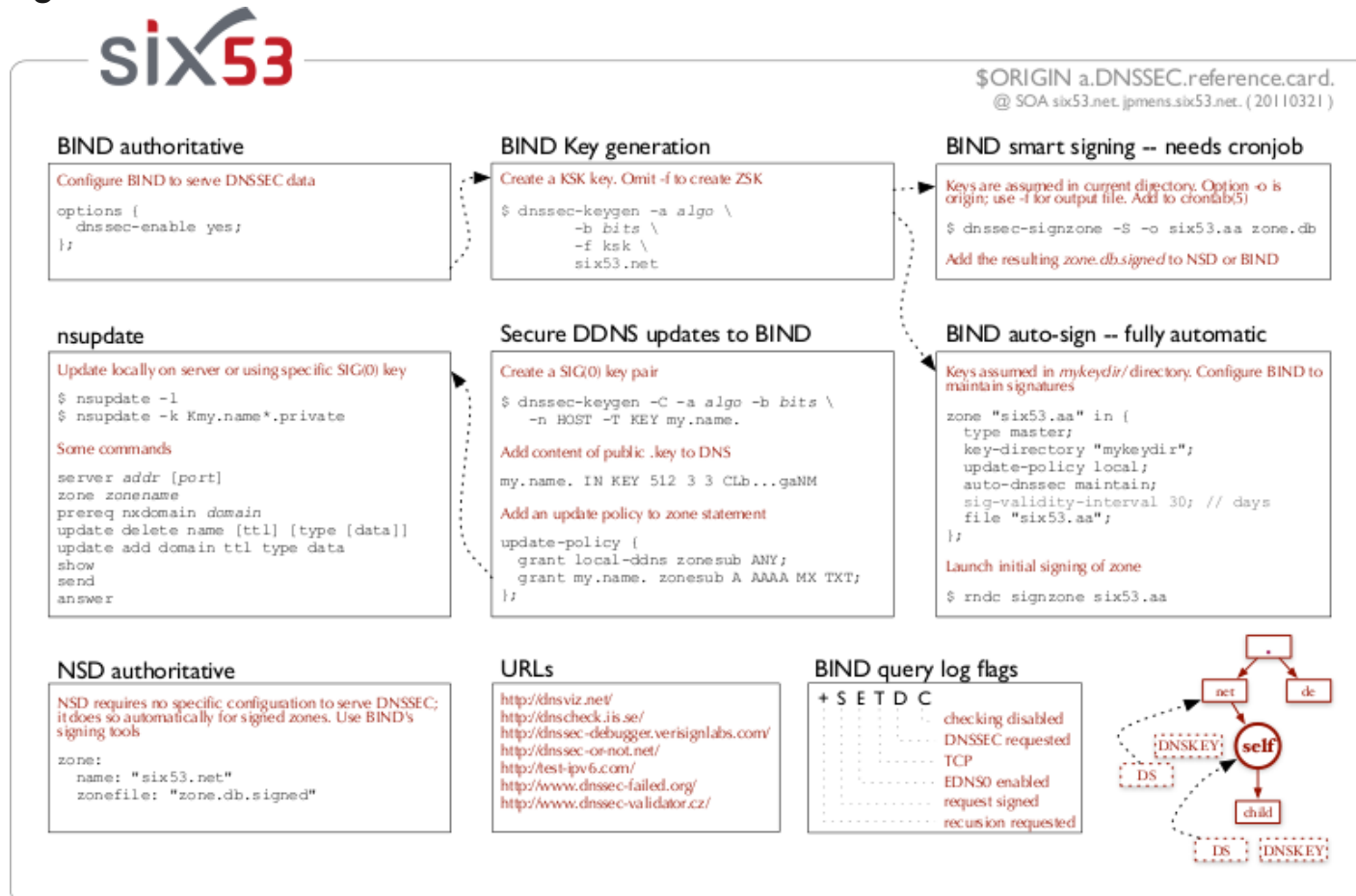
Kostenfreie Betaphase

Jetzt anmelden:
<http://six53.net/>

DNSSEC reference card:

<http://six53.net/refcard>

- Zum nachschlagen, ausdrucken und verschenken :-)
- Doppelseitig DIN-A5; seit dieser Woche erste „Beta-Version“ veröffentlicht



six53

\$ORIGIN a.DNSSEC.reference.card.
@ SOA six53.net.jpmins.six53.net. (20110321)

BIND authoritative

Configure BIND to serve DNSSEC data

```
options {
  dnssec-enable yes;
};
```

BIND Key generation

Create a KSK key. Omit -f to create ZSK

```
$ dnssec-keygen -a algo \
  -b bits \
  -f ksk \
  six53.net
```

BIND smart signing -- needs cronjob

Keys are assumed in current directory. Option -o is origin; use -f for output file. Add to crontab(5)

```
$ dnssec-signzone -S -o six53.aa zone.db
```

Add the resulting zone.db.signed to NSD or BIND

nsupdate

Update locally on server or using specific SIG(0) key

```
$ nsupdate -l
$ nsupdate -k Kmy.name*.private
```

Some commands

```
server addr [port]
zone zonedomain
prereq nxdomain domain
update delete name [ttl] [type [data]]
update add domain ttl type data
show
send
answer
```

Secure DDNS updates to BIND

Create a SIG(0) key pair

```
$ dnssec-keygen -C -a algo -b bits \
  -n HOST -T KEY my.name.
```

Add content of public .key to DNS

```
my.name. IN KEY 512 3 3 Cfb...gaNM
```

Add an update policy to zone statement

```
update-policy {
  grant local-ddns zonesub ANY;
  grant my.name. zonesub A AAAA MX TXT;
};
```

BIND auto-sign -- fully automatic

Keys assumed in mykeydir/ directory. Configure BIND to maintain signatures

```
zone "six53.aa" in {
  type master;
  key-directory "mykeydir";
  update-policy local;
  auto-dnssec maintain;
  sig-validity-interval 30; // days
  file "six53.aa";
};
```

Launch initial signing of zone

```
$ rndc signzone six53.aa
```

NSD authoritative

NSD requires no specific configuration to serve DNSSEC; it does so automatically for signed zones. Use BIND's signing tools


```
zone:
  name: "six53.net"
  zonefile: "zone.db.signed"
```

URLs

```
http://dnsviz.net/
http://dnscheck.is.se/
http://dnssec-debugger.verisignlabs.com/
http://dnssec-on-not.net/
http://test-ipv6.com/
http://www.dnssec-failed.org/
http://www.dnssec-validator.cz/
```

BIND query log flags

```
+ S E T D C
..... checking disabled
..... DNSSEC requested
..... TCP
..... EDNSO enabled
..... request signed
..... recursion requested
```



DNSSEC-Validator Firefox: <http://www.dnssec-validator.cz/>

DNSSEC-Funktionstests:

- <http://dnssec-or-not.org/>
- <http://dnssectest.sidn.nl/>
- <http://dnscheck.iis.se/>
- <http://dnssec-debugger.verisignlabs.com/>
- <http://test-ipv6.com/>
- <http://www.dnssec-failed.org/>

Visualisierung: <http://dnsviz.net/>

DNSSEC Reference-Card: <http://six53.net/refcard>

Danke fürs Zuhören
sowie
viel Erfolg und Spaß
mit DNSSEC!

Link zu den Slides: <http://talks.speedpartner.de>

DNSSEC reference card: <http://six53.net/refcard>

Bei Fragen stehen wir selbstverständlich gerne zur Verfügung:

Stefan Neufeind, neufeind@speedpartner.de
SpeedPartner GmbH, <http://www.speedpartner.de/>