Konfiguration der Nameserver-Optionen

- Kurzakte -

Es darf nur eine Deklaration "options" geben. Existieren mehrere, wird nur die erste abgearbeitet.

wichtige Optionen:

- version <Versionsstring oder "not revealed">
  Die Versionsangabe. Wird diese Angabe weggelassen, so wird die wahre Version des NAMED verwendet. (Für die Freunde der Übertreibung ist es also möglich hier die Version 99.9.1 anzugeben).
- directory <Pfad>
  Das Arbeitsverzeichnis des Servers. Alle späteren Dateiangaben, die keinen absoluten Pfad beinhalten werden relativ zu diesem Pfad gewertet.
- dump-file <Dateiname>
  Der Dateiname, in den named seinen Server-Dump schreibt, wenn er das Signal SIGINT bekommt.
- recursion <yes|no>
  Wenn hier yes steht und eine Anfrage Rekursion erfordert, wird der Server alle Arbeit erledigen und die Antwort dem Client zurückgeben. Ansonsten wird er nur eine Referenz eines übergeordneten Servers zurückgeben.
- fetch-glue <yes|no>
  Wenn Rekursion abgeschaltet wurde, sollte auch fetch-glue abgeschaltet werden.
- data-size <Zahl>
  Die maximale Speichergröße, die ein Server nutzen darf.
- allow-transfer {Adressenliste};
  Gibt an, welche Hosts die Erlaubnis haben, Zonentransfers vom Server zu empfangen.

```
options {
        directory "/var/cache/bind";
        pid-file "/var/run/bind/named.pid";
        notify yes;
        allow-transfer { 195.20.224.97; 195.20.225.34; };
        forwarders { 195.20.224.234; 195.20.224.99; };
        forward first;
        listen-on port 53 { 127.0.0.1; 217.160.xyz.abc; };
        listen-on-v6 { none; };
        allow-query { 127.0.0.1; 217.160.xyz.abc; };
        allow-recursion { 127.0.0.1; 217.160.xyz.abc; };
        auth-nxdomain no;    # conform to RFC1035
//        version "My version is so secret that I even dont know what Im running on";
// Wer seine Bind Version "verstecken" will, kann die beide // vor version entfernen.
        // If there is a firewall between you and nameservers you want
        // to talk to, you might need to uncomment the query-source
        // directive below.  Previous versions of BIND always asked
        // questions using port 53, but BIND 8.1 and later use an unprivileged
        // port by default.

        // query-source address * port 53;

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

        // forwarders {
        //      0.0.0.0;
        // };

        // auth-nxdomain no;   # conform to RFC1035

};
```

Die wichtigsten Konfigurationsoptionen im Abschnitt options

directory "filename";
: gibt das Verzeichnis an, in dem der BIND die Dateien mit den Zonendaten findet; dies ist in der Regel `/var/lib/named`.

forwarders { ip-address; };
: verwendet man, um den oder die Nameserver (meist des Providers) anzugeben, an den oder die die DNS-Anfragen weitergereicht werden, die nicht direkt beantwortet werden können. Anstelle von *ip-address* verwenden Sie eine IP-Adresse wie `10.0.0.1`.

forward first;
: bewirkt, dass die DNS-Anfragen zu erst geforwarded werden, bevor versucht wird diese über die Root-Nameserver aufzulösen. Anstelle von `forward first` kann man auch `forward only` schreiben, dann werden alle Anfragen weitergeleitet und die Root-Nameserver werden gar nicht mehr angesprochen. Das kann für Firewall-Konfigurationen sinnvoll sein.

listen-on port 53 { 127.0.0.1; ip-address; };
: sagt BIND, auf welchen Netzwerkinterfaces und welchem Port er Anfragen der Clients entgegen nehmen soll. Die Angabe `port 53` kann man sich dabei sparen, da `53` ohnehin der Standardport ist. Mit `127.0.0.1` lässt man Anfragen von localhost zu. Lässt man diesen Eintrag komplett weg, werden standardmäßig alle Interfaces verwendet.

listen-on-v6 port 53 { any; };
: sagt dem BIND, auf welchem Port er auf Anfragen der Clients horcht, die IPv6 verwenden. Außer `any` ist alternativ nur noch `none` erlaubt, da der Server stets auf der IPv6-Wildcard-Adresse horcht.

*query-source address * port 53;*
: Dieser Eintrag kann notwendig sein, wenn eine Firewall die externen DNS-Abfragen blockiert. So wird BIND dazu gebracht, Anfragen nach außen von Port 53 aus und nicht von den hohen Ports > 1024 zu stellen.

*query-source-v6 address * port 53;*
: Dieser Eintrag muss für Anfragen über IPv6 verwendet werden.

allow-query { 127.0.0.1; net; };
: bestimmt die Netze, aus denen Clients DNS-Anfragen stellen dürfen. Anstelle von *net* trägt man Adressenangaben wie `192.168.1/24` ein; dabei ist `/24` eine Kurzschreibweise für die Anzahl der Bits in der Netzmaske, in diesem Fall `255.255.255.0`.

*allow-transfer { ! *; };*
: regelt, welche Rechner Zonentransfers anfordern dürfen, dieses Beispiel unterbindet sie, aufgrund des `! *` komplett. Ohne diesen Eintrag können Zonentransfers ohne Einschränkungen von überall angefordert werden.

statistics-interval 0;
: Ohne diesen Eintrag produziert BIND stündlich mehrere Zeilen Statistikmeldungen in `/var/log/messages`. Die Angabe von `0` bewirkt, dass diese komplett unterdrückt werden; hier kann man die Zeit in Minuten angeben.

cleaning-interval 720;
: Diese Option legt fest, in welchem Zeitabstand BIND seinen Cache aufräumt. Die Aktivität führt jedes Mal zu einem Eintrag in `/var/log/messages`. Die Zeitangabe erfolgt in Minuten. Voreingestellt sind 60 Minuten.

interface-interval 0;
: BIND durchsucht regelmäßig die Netzwerkschnittstellen nach neuen oder nicht mehr vorhandenen Interfaces. Setzt man diesen Wert auf `0`, so wird darauf verzichtet und BIND lauscht nur auf den beim Start gefundenen Interfaces. Alternativ kann man das Intervall in Minuten angeben. Voreingestellt sind 60 Minuten.

notify no;
: Das `no` bewirkt, dass keine anderen Nameserver benachrichtigt werden, wenn an den Zonendaten Änderungen vorgenommen werden oder der Nameserver neu gestartet wird.

Der Konfigurationsabschnitt Logging

Was und wie wohin mitprotokolliert wird, kann man beim BIND recht vielseitig konfigurieren. Normalerweise sind die Voreinstellungen ausreichend. Datei 22.12. “Logging wird unterdrückt” zeigt die einfachste Form eines solchen Eintrags und unterdrückt das „Logging“ komplett.

Beispiel 22.12. Logging wird unterdrückt

```
logging {
        category default { null; };
```

```
};
```

Die Zonen-Optionen:

type master;

Das `master` legt fest, dass diese Zone auf diesem Nameserver verwaltet wird. Das setzt eine korrekt erstellte Zonendatei voraus.

type slave;

Diese Zone wird von einem anderen Nameserver transferiert. Muss zusammen mit `masters` verwendet werden.

type hint;

Die Zone `.` vom Typ `hint` wird für die Angabe der Root-Nameserver verwendet. Diese Zonendefinition kann man unverändert lassen.

file "meine-domain.zone" oder file "slave/andere-domain.zone";

Dieser Eintrag gibt die Datei an, in der die Zonendaten für die Domain eingetragen sind. Bei einem `slave` braucht die Datei nicht zu existieren, da ihr Inhalt von einem anderen Nameserver geholt wird. Um Master- und Slave-Dateien auseinander zu halten, gibt man für die Slave-Dateien das Verzeichnis `slave` an.

masters { server-ip-address; };

Diesen Eintrag braucht man nur für Slave-Zonen und er gibt an, von welchem Nameserver die Zonendatei transferiert werden soll.

*allow-update { ! *; };*

diese Option regelt den Schreibzugriff von extern auf die Zonendaten. Damit wäre es Clients möglich, sich selbst im DNS einzutragen, was aus Sicherheitsgründen nicht wünschenswert ist. Ohne diesen Eintrag, sind Zonen-Updates generell untersagt, dieses Beispiel würde daran auch nichts ändern, da `! *` ebenfalls alles verbietet.

Konfiguration des Loggings

Es darf nur eine Deklaration "logging" existieren. (Ansonsten wird die erste benutzt)

Channels
legen fest, was mit zu ihnen gelieferten Meldungen passiert, Es können beliebig viele Kanäle definiert werden.

- file Dateiname
  Meldungen an diesen Kanal werden in die angegebene Datei geschrieben. Zusätzlich können hier noch Angaben über die maximale Größe der Datei (size 10m), sowie die Anzahl der zu erstellenden Versionen gemacht werden (versions 5).
- syslog (Herkunft)
  Meldungen an diesen Kanal werden unter der Verwendung der angegebenen Herkunft (kern, user, mail, daemon, auth, syslog, lpr, news, uucp, cron, authpriv, ftp, local0-7) an den Syslog-Daemon weitergegeben. Die Priorität wird später (für alle drei denkbaren Kanaltypen gleich) unter dem Begriff severity angegeben.
- null
  Diese Meldungen werden ignoriert.

Optionale Einstellungen pro Kanal

- severity
  Entspricht der Priorität bei syslog. Mögliche Werte sind critical, error, warning, notice, info, debug [level], und dynamic (d. h., Priorität wird übernommen).
- print-category (yes|no)
  Soll die Kategorie der Meldung mit ausgegeben werden.
- print-severity (yes|no)
  Soll die Priorität der Meldung mit ausgegeben werden.
- print-time (yes|no)
  Soll die Zeit mit ausgegeben werden. Der syslogd gibt die Zeit immer mit an, hier kann das also ausgeschaltet sein (no), in Dateien ist es aber sehr praktisch, wenn sie mit angegeben ist (yes).

Kategorien (category)
legen fest, welche Meldungen auf welchen Kanal geschickt werden
wird keine default-Kategorie festgelegt, wird diese verwendet:

```
category default { default_syslog; default_debug; };
```

- db alle Datenbankoperationen
- default alles (catch-all)
- cname
- config High-Level Konfiguration
- eventlib Debugging-Infos vom Event-System, nur ein einziger Channel möglich, zwingend vom typ file
  falls nicht festgelegt: category eventlib { default_debug; };
- insist interne Probleme beim Konsistenzcheck
- lame-servers Meldungen wie "Lame Server on ...")
- load Meldungen vom Laden der Zone
- maintenance periodische Maintenace-Events
- ncache negatives Caching
- notify
- os Probleme mit dem Betriebssystem
- packet Dumps von gesendeten und empfangenen Paketen, nur ein einziger Channel, vom typ file
  falls nicht festgelegt: category packet { default_debug; };
- panic die Ursache. die den Name-Server zwingt, sich herunterzufahren
  falls nicht festgelegt: category panic { default_syslog; default_stderr; };
- parser Low-Level Konfiguration
- queries eine Meldung für jede Abfrage
- response-checks. Meldungen von Verbindungschecks
- security
- statistics Statistiken
- update dynamische Updates
- xfer-in eingehende Zonen-Transfers
- xfer-out ausgehende Zonen-Transfers

Beispielskonfiguration für das Logging:

```
  logging {
    // Zuerst definieren wir ein paar Kanäle
    channel alle_abfragen {
      file "/var/log/dns-queries" versions 5 size 10m;
      print-time yes;
      print-category no;
      print severity no;
    };

    channel normallog {
      syslog daemon;
      severity dynamic;  // Die Priorität wird übernommen
    };

    channel muelleimer {
      none;
    };

    // Jetzt bestimmen wir, welche Kategorie in welchen Kanal geht
    category default {
      normallog;
    };

    category queries {
      alle_abfragen;
    };

    category lame-servers {
      muelleimer;
    };
  };

As an example, let's say you want to log security events to a file, but
     you also want keep the default logging behavior.  You'd specify the
fol
     lowing:

          channel my_security_channel {
              file "my_security_file";
              severity info;
          };
          category security { my_security_channel;
                              default_syslog; default_debug; };

     To discard all messages in a category, specify the null channel:

          category lame-servers { null; };
          category cname { null; };
```

Fremde Server spezifizieren

Statement: server IP-Adresse

Damit können Eigenschaften fremder Server angegeben werden wie

bogus (yes|no); wenn bekannt, dass ein Server falsche Antworten versendet

transfer-format (one-answer|many-answers); Pakete des Zonentransfers enthalten entweder eine oder mehrere Antworten (seit V. 8.1)

keys key_id [key-id]; zur Zeit noch ohne Bedeutung, zur eindeutigen Identifizierung des Nameservers

Kontrollprogramm ndc

In den Versionen 8 und 9 wurde die Diagnose per Prozess-Signal abgelöst durch das Tool ndc (rndc bei V. 9)

Das Programm ndc (Name Daemon Control) erlaubt es, einem bind8 Nameserver Befehle im laufenden Betrieb zu übermitteln. Dazu wird ein gemeinsam genutzter Kanal (control-channel) genutzt, standardmäßig ist das die Datei /var/run/ndc, ein UNIX domain socket, der in dem Moment aufgebaut wird, in dem der Nameserver startet. Der Nameserver lauscht auf diesem Kanal nach Kommandos, das Programm ndc kann Kommandos auf diesen Kanal schreiben.

ndc öffnet eine interaktive Shell, die es erlaubt, dem Nameserver Kommandos zu schicken. Diese Shell kann mit dem Befehl /exit wieder geschlossen werden. Alternativ dazu kann ein Kommando auch gleich an die Kommandozeile angehängt werden.

Da es vom jeweils verwendeten Nameserver abhängt, welche Kommandos er versteht, gibt es nur einige wenige fest in ndc eingebaute Kommandos. Das wichtigste davon ist /help, das den Nameserver selbst fragt, welche Kommandos er anbietet und diese Kommandos dann auflistet.

Typische und wichtige Kommandos der Nameserver sind

• getpid		Gibt die PID des Nameservers zurück.
• status		Gibt den Status des Nameservers aus. Dazu zählen Informationen wie Version, Debuglevel und Anzahl gerade aktiver Querys
• stats	SIGABRT	
• trace	SIGUSR1	
• stop	SIGTERM	Beendet den laufenden Namedaemon. Danach ist nur noch ein /exit möglich, da der Kommandokanal jetzt nicht mehr existiert.
• halt	SIGKILL	sofortiges Töten
• reload [zone]	SIGHUP	Mit diesem Befehl können entweder alle oder einzelne Zonen neu geladen werden. Nützlich wenn Zonendateien manuell verändert wurden.
• dumpdb	SIGINT	Dieser Befehl erstellt eine Datei (meist /var/tmp/named_dump.db oder durch Umgebungsvariable _PATH_DUMPFILE festgelegt) in der der gesamte Cache des Nameservers steht.

Steuerung des Nameservers mit Signalen

Statt mit Hilfe des ndc-Programms können auch einige Aufgaben mit Signalen erledigt werden. Dazu muss mit kill ein bestimmtes Signal an die PID des Nameservers geschickt werden. Folgende Signale werden verarbeitet:

- SIGHUP Neu einlesen der Konfigurationsdatei und dadurch Neuladen der Datenbanken.
- SIGINT Dump des Cache in dieselbe Datei, in die auch ndc dumpdb schreiben würde.
- SIGTERM Sichert alle Zoneninfos und beendet den Nameserver

22.7.7. Sichere Transaktionen

Sichere Transaktionen kann man mithilfe der „Transaction SIGnatures“ (TSIG) verwirklichen. Dafür kommen Transaktionsschlüssel (engl. *Transaction Keys*) und -signaturen (engl. *Transaction Signatures*) zum Einsatz, deren Erzeugung und Verwendung in diesem Abschnitt beschrieben wird.

Benötigt werden sichere Transaktionen bei der Kommunikation von Server zu Server und für dynamische Aktualisierungen der Zonendaten. Eine auf Schlüsseln basierende Zugriffskontrolle bietet dafür eine weit größere Sicherheit als eine Kontrolle, die auf IP-Adressen basiert.

Ein Transaktionsschlüssel kann mit folgendem Kommando erzeugt werden (für mehr Informationen vgl. die Manualpage von **dnssec-keygen**):

```
dnssec-keygen -a hmac-md5 -b 128 -n HOST host1-host2
```

Es entstehen dadurch zwei Dateien mit beispielsweise folgenden Namen:

```
Khost1-host2.+157+34265.private
Khost1-host2.+157+34265.key
```

Der Schlüssel ist in beiden Dateien enthalten (z.B. ejIkuCyyGJwwuN3xAteKgg==). Zur weiteren Verwendung sollte Khost1-host2.+157+34265.key auf sicherem Wege (zum Beispiel mit **scp**) auf den entfernten Rechner übertragen und dort in der /etc/named.conf eingetragen werden, um eine sichere Kommunikation zwischen host1 und host2 zu bewirken:

```
key host1-host2. {
  algorithm hmac-md5;
  secret "ejIkuCyyGJwwuN3xAteKgg==";
};
```

Zugriffsrechte von /etc/named.conf

Achten Sie darauf, dass die Zugriffsrechte auf /etc/named.conf eingeschränkt bleiben; die Vorgabe ist 0640 für root und die Gruppe named; alternativ kann man die Schlüssel auch in ein eigene geschützte Datei auslagern und diese dann includieren.

Damit auf dem Server host1 der Schlüssel für host2 mit der Beispielsadresse 192.168.2.3 verwendet wird, muss auf dem Server in der /etc/named.conf eingetragen werden:

```
server 192.168.2.3 {
  keys { host1-host2. ;};
};
```

In den Konfigurationsdateien von host2 müssen entsprechende Einträge vorgenommen werden.

Zusätzlich zu den ACLs auf Basis von IP-Adressen und Adress-Bereichen, soll man, um sichere Transaktionen auszuführen, TSIG-Schlüssel hinzufügen; ein Beispiel dafür kann so aussehen:

```
allow-update { key host1-host2. ;};
```

Mehr dazu findet man im *BIND Administrator Reference Manual* zu update-policy.

22.7.8. Zonendaten dynamisch aktualisieren

Dynamische Aktualisierungen (engl. *Dynamic Update*) ist der Terminus, der das Hinzufügen, Ändern oder Löschen von Einträgen in den Zonen-Dateien eines Masters bezeichnet. Beschrieben ist dieser Mechanismus im RFC 2136.

Dynamische Aktualisierungen werden je Zone mit den Optionen allow-update oder update-policy bei den Zonen-Einträgen konfiguriert. Zonen, die dynamisch aktualisiert werden, sollten nicht von Hand bearbeitet werden.

Mit **nsupdate** werden die zu aktualisierenden Einträge an den Server übertragen; zur genauen Syntax vgl. die Manualpage von **nsupdate**. Die Aktualisierung sollte aus Sicherheitsüberlegungen heraus unbedingt über sichere Transaktionen (TSIG) geschehen; vgl. Abschnitt 22.7.7. “Sichere Transaktionen”.

22.7.9. DNSSEC

DNSSEC (engl. *DNS Security*) ist im RFC 2535 beschrieben; welche Tools für den Einsatz von DNSSEC zur Verfügung stehen, ist im BIND-Manual beschrieben.

Eine sichere Zone muss einen oder mehrere Zonen-Schlüssel haben; diese werden, wie die Host-Schlüssel, auch mit **dnssec-keygen** erzeugt. Zur Verschlüsselung wählt man momentan DSA.

Die öffentlichen Schlüssel (engl. *public keys*) sollten in die Zonen-Dateien mit $INCLUDE eingebunden werden.

Alle Schlüssel werden mit **dnssec-makekeyset** zu einem Set zusammengefasst, das auf sicherem Wege an die übergeordnete Zone (engl. *Parent Zone*) zu übertragen ist, um dort mit **dnssec-signkey** signiert zu werden.

Die bei der Signierung erzeugten Dateien müssen zum Signieren von Zonen mit **dnssec-signzone** verwendet werden und die dabei entstandenen Dateien sind schließlich in /etc/named.conf für die jeweilige Zone einzubinden.