serverPKI Documentation

Release 0.9.10

Axel Rau

Aug 11, 2020

Contents

1 serverPKI **3**
- 1.1 What . . . 3
- 1.2 Prerequisites . . . 4
- 1.3 Sponsored . . . 4

2 Changelog **5**
- 2.1 0.9.0 (2017-07-18) . . . 5
- 2.2 0.9.1 (2017-07-28) . . . 5
- 2.3 0.9.2 (2018-03-19) . . . 5
- 2.4 0.9.3 (2019-02-11) . . . 6
- 2.5 0.9.4 (2019-02-21) . . . 6
- 2.6 0.9.6 (2020-03-11) . . . 6
- 2.7 0.9.10 (2020-08-06) . . . 6
- 2.8 0.9.11 (2020-08-11) . . . 7

3 Installation and Configuration **9**
- 3.1 Installation . . . 9
- 3.2 Configuration . . . 10

4 Tutorial **13**
- 4.1 Setting up encrypted key storage . . . 13
- 4.2 Creating our first local certificate . . . 13
- 4.3 Creating our first Let's Encrypt certificate . . . 14

5 The database **17**
- 5.1 Model . . . 17
- 5.2 Tables . . . 19
- 5.3 Views . . . 23
- 5.4 Functions . . . 24

6 Operation **27**
- 6.1 Management of configuration . . . 27
- 6.2 Management of cert instances . . . 28
- 6.3 State table of cert instances . . . 33

7 Indices and tables **35**

Index **37**

serverPKI is a tool to issue, renew and distribute SSL certificates for internet servers without manual intervention. Distribution to target hosts and reloading of server configuration is done via ssh/sftp. Configuration and cert/key data is stored in a relational database.

CHAPTER 1

serverPKI

serverPKI Python PKI for internet server infrastructure

Copyright Copyright (c) 2015-2020 Axel Rau axel.rau@chaos1.de

License GPLv3

Homepage https://github.com/mc3/serverPKI

Documentation https://serverpki.readthedocs.io

1.1 What

serverPKI is a tool to issue, renew and distribute SSL certificates for internet servers. Distribution to target hosts and reloading of server configuration is done via ssh/sftp. Configuration and cert/key data is stored in a relational database.

serverPKI includes support for

- local CA
- LetsEncrypt CA (supports only acme v2 api, see https://letsencrypt.org/docs)
- FreeBSD service jails via ssh access to host
- publishing of DANE RR in DNS, using BIND 9 and TLSA key rollover (see RFC 6698)
- controlling DNS zone info for LetsEncrypt challenges und TLSA RR via dynamic DNS updates (recommended) or via zone files.
- unattended operation via cronjob
- extensive logging
- alerting via mail

1.2 Prerequisites

- PostgreSQL 12+ server
  - The contrib utilities from the PostgreSQL distribution are required (serverPKI needs the citext extension for case insensitive indexes)
  - a DB account with super user privileges [dba] or assistance of a DB admin (serverPKI uses a dedicated DB user [pki_op] and a dedicated DB)
  - authentication record in pg_hba.conf to allow access of pki_op from local host (client cert authentication recommended)
- PostgreSQL 12+ client installation on local host
- bind 9 DNS server (9.16+ should be used)
  - If DNS is handled via zone files,
    - serverPKI must be run on the master (hidden primary) DNS server.
    - signed Zones being maintained by serverPKI must be run in auto-dnssec maintain + inline-signing operation mode.
    - Zone files must be writable by serverPKI process to allow publishing of acme_challenges and TLSA resource records for DANE
- Python 3.7+ must be installed (tested with Python 3.8.3)
- Running serverPKI in a Python virtual environment is recommended for ease of upgrading. The author uses *virtualenvwrapper*.

1.3 Sponsored

This project is being developed with the powerful Python IDE PyCharm, which is particularly useful during remote debugging sessions. A professional license has been granted by JetBrains, https://www.jetbrains.com/.

CHAPTER 2

Changelog

2.1 0.9.0 (2017-07-18)

- Initial public release.

2.2 0.9.1 (2017-07-28)

- Documentation at https://serverpki.readthedocs.io

2.3 0.9.2 (2018-03-19)

- Python 3.6 supported
- Omit disabled certs from list of certs to be renewed.
- BUGFIX: Bind place to jail not to disthost (disthost->jail->place)
- Do not expire certs one day before "not_after" but one day after instead
- Allow "distribute only" with –renew-local-certs
- **New Feature: –renew-local-certs REMAINING_DAYS** Renews local certs, which would expire within REMAINING_DAYS. Gives a nice tabular display of affected certs
- New Feature: Allow encrypted storage of keys in DB

  2 new action commands: –encrypt-keys and –decrypt-keys

  New configuration parameter: db_encryption_key
- **Upgrading:** Create new table Revision in DB - see install/create_schema_pki.sql:

```
pki_op=# CREATE TABLE Revision (
id                  SERIAL          PRIMARY KEY,            -- 'PK of Revision'
schemaVersion       int2            NOT NULL  DEFAULT 1,    -- 'Version of DB␣
↪schema'
keysEncrypted       BOOLEAN         NOT NULL  DEFAULT FALSE -- 'Cert keys are␣
↪encrypted'
);
pki_op=# INSERT INTO revision (schemaVersion) values(1);
```

Then create passphrase and encrypt DB (see tutorial).

2.4 0.9.3 (2019-02-11)

- Python 3.7 supported
- With pyopenssl 19 on FreeBSD 12 (which has OpenSSL 1.1.1a-freebsd in base system), paramiko 2.4 works out-of-the-box. No longer need for paramiko workarounds like package paramiko-clc.
- Now recovering from “Letsencrypt forgetting authorizations”, which happened at begin of 2019.
- Fixing bug where one letsencrypt authorization was requested multiple times (happened once per distribution target).
- Being more patient with Letsencrypt’s response to challenges

2.5 0.9.4 (2019-02-21)

- INCOMPATIBLE CHANGE in configuration file syntax: dbAccounts keyword has been changed from ‘pki_dev’ to ‘serverpki’. See install example_config.py
- Multiple local CA certs for CA cert roll over
- Increased hash size to 512 (CA cert) resp. 384 bits (server/client cert)
- Cert (including CA cert) export by cert serial number implemented.
- Listing of cert meta info now also lists (issued) cert instances.
- requirement for PyOpenSSL removed.
- BUGFIXES e.g. Allow to enter 1st cert into empty CertInstances table

2.6 0.9.6 (2020-03-11)

- Supporting and (requiring) V2 of ACME protocoll.
- New fields in DB for upcoming support of certs with elliptic algorithm. (in addition to rsa). Run install/upgrade_to_2.sql in psql, connected to pki DB.

2.7 0.9.10 (2020-08-06)

- New object oriented architecture, abstracting relational model

- Support for dynamic DNS update mode of operation supported
- Support for dual algo certs (rsa + ec)
- Support for OCSP_must_staple attribute
- New config file format
- BUGFIXES mainly in ACMEv2 handshaking code
- For upgrade run install/upgrade_to_{3456}.sql in psql, connected to pki DB.

2.8 0.9.11 (2020-08-11)

- Using automatoes 0.9.5. Got hotfix from automatoes maintainer

CHAPTER 3

Installation and Configuration

3.1 Installation

- Installation of PostgreSQL client package:
- Installation of PostgreSQL server (if none exists) and related packages on DB server host:

```
pkg install databases/postgresql12-server
pkg install databases/ip4r
```

- Installation of Python packages from PyPI:

```
pip install serverPKI
```

- Creation of DB user and DB

  host db1, port 2222, user dba and user pki_op are examples. dba must be pgsql superuser. In scripts create_schema_pki.sql and create_triggers_pki.sql are GRANT statements which allow usage of objects by user serverPKI. To change this, you must edit those scripts. Create ~/.pgpass or client cert in ~/.postgresql:

```
psql -h db1 -p 2222 -U dba postgres
postgres=> CREATE ROLE pki_op LOGIN CREATEDB;
psql -h db1 -p 2222 -U pki_op postgres
postgres=> CREATE DATABASE pki_op;
psql -h db1 -p 2222 -U pki_op -d pki_op -f install/fresh_install/create_
↪schema_dd.sql
psql -h db1 -p 2222 -U pki_op -d pki_op -f install/fresh_install/create_
↪extension_citext.sql
psql -h db1 -p 2222 -U pki_op -d pki_op -f install/fresh_install/create_
↪schema_pki.sql

# optional (usefull examples for demo):
psql -h db1 -p 2222 -U pki_op -d pki_op -f install/fresh_install/load_
↪testdata.sql
```

(continues on next page)

(continued from previous page)

```
psql -h db1 -p 2222 -U pki_op -d pki_op -f install/fresh_install/create_
↪triggers_pki.sql
#
psql -h db1 -p 2222 -U pki_op
pki_op=> set search_path to pki,dd;
SET
pki_op=> \d
                   List of relations
 Schema |          Name          |   Type   |   Owner
--------+------------------------+----------+-----------
 pki    | certificates           | table    | pki_op
 pki    | certificates_id_seq    | sequence | pki_op
 pki    | certificates_services  | table    | pki_op
 pki    | certinstances          | table    | pki_op
 pki    | certinstances_id_seq   | sequence | pki_op
 pki    | certkeydata            | table    | pki_op
 pki    | certkeydata_id_seq     | sequence | pki_op
 pki    | certs                  | view     | pki_op
 pki    | certs_ids              | view     | pki_op
 pki    | disthosts              | table    | pki_op
 pki    | disthosts_id_seq       | sequence | pki_op
 pki    | inst                   | view     | pki_op
 pki    | jails                  | table    | pki_op
 pki    | jails_id_seq           | sequence | pki_op
 pki    | places                 | table    | pki_op
 pki    | places_id_seq          | sequence | pki_op
 pki    | revision               | table    | pki_op
 pki    | revision_id_seq        | sequence | pki_op
 pki    | services               | table    | pki_op
 pki    | services_id_seq        | sequence | pki_op
 pki    | subjects               | table    | pki_op
 pki    | subjects_id_seq        | sequence | pki_op
 pki    | targets                | table    | pki_op
 pki    | targets_id_seq         | sequence | pki_op
(24 rows)

serverpki=> \q
```

3.2 Configuration

Copy install/example_config.py to /usr/local/etc/serverPKI/serverPKI_config.py or to VIRTUAL_ENV/etc/serverPKI_config.py and edit the copy. The config file is in ini file format with nested sections.

The following variables can be set:

3.2.1 Pathes

Section containg filesystem path information

home Root of the work area and credential storage, usually somewhere at var. This variable must be set to a save place in order to use serverPKI

db Some credentials stored here, like:

ca_cert, ca_key Cert and key of the local (internal) CA, in case, there exists one when you begin with serverPKI. Will be imported into DB with issuence of 1st local cert. The flat files can be deleted then. Not needed, if local CA cert created with "serverPKI –issue-local-CAcert".

db_encryption_key All keys in DB are encrypted with this key. After setting this up, encrypt keys in DB:

```
operate_serverPKI --encrypt-keys -v
```

Before changing the passphrase, decrypt all keys:

```
operate_serverPKI --decrypt-keys -v
```

le_account Credentials of Lets Encrypt account in json format. See manuale register in tutorial.

work Work direcory

work_tlsa TLSA resource records are being accumulated here for named zone update.

tlsa_dns_master Host of DNS master. Empty means: Local host. Must be empty for now. Will be used with ddns with remote master in the future.

Next 6 variables are for historical DNS control via zone files and should not be used for new installations:

zone_file_root

zone files are kept in DSKM format: zone_file_root/example.com/example.com.zone

dns_key rndc key for triggering named reload.

zone_tlsa_inc_mode, zone_tlsa_inc_uid, zone_tlsa_inc_gid file permission and ownership for files, incuded by zone files.

zone_file_include_name The filename of the file, included from zone file with the challenges.

ddns_key_file The filename of a named dynamic dns key file, used to secure dns update transactions.

3.2.2 X509atts

Section of local X509 certificate standard attribute defaults

names and extensions Cert fields used for CA cert and server/client certs.

lifetime and bits are used for server/client certs

3.2.3 DBAccount

Configuration of account data and credentials for the PostgreSQL DB. Passwords may be stored in pki_op's HOME in HOME/.pgpass or client certs in HOME/.postgresql.crt and HOME/.postgresql.key

dbHost host name of DB server

dbPort port number of DB instance

dbUser DB role name, used for accessing the DB

dbDbaUser Role name for tasks requiring super user rights. Empty, if person who runs program is DBA

dbSslRequired If "yes" then connecting will be made with TLS

dbDatabase name of database, used for serverPKI (contains schemas dd and pki)

dbSearchPath search_path set at login

dbCert path of file containg cert for TLS

dbCertKey path of file containg key for TLS

3.2.4 Misc

Section with miscellaneous config parameters

SSH_CLIENT_USER_NAME user name on target hosts for cert/key distribution

LE_SERVER

URL of Lets Encrypt server, either (for testing): 'https://acme-staging-v02.api.letsencrypt.org'

or (for production): 'https://acme-v02.api.letsencrypt.org'

LE_EMAIL e-mail address for letsencrypt.org registration, used for notifications by LE

LE_ZONE_UPDATE_METHOD Zone update method for challenges, either 'ddns' (the default) for dynamic updates or 'zone_file' for updates via zone file)

LOCAL_CA_BITS LOCAL_CA_LIFETIME Number of bits and lifetime of local CA cert.

SUBJECT_LOCAL_CA Subject name of local CA in table Subjects (may be changed only initially)

SUBJECT_LE_CA Subject name of Lets Encrypt CA in table Subjects (may be changed only initially)

PRE_PUBLISH_TIMEDELTA New certs are published that many days before they become active (with 2nd TLSA RRs) for rollover

LOCAL_ISSUE_MAIL_TIMEDELTA = timedelta(days=30) E-Mail to administrator will be sent that many days before expiration of local certs. (Must be issued manually, using pass phrase)

MAIL_RELAY, MAIL_SUBJECT, MAIL_SENDER and MAIL_RECIPIENT Characteristics of mail service for notification mails.

SYSLOG_FACILITY Facility for syslog log messages

serverPKI uses levels DEBUG, INFO, NOTICE and ERR

CHAPTER 4

Tutorial

In the following examples, client certs are used as PostgreSQL authentication method. su is used to run the commands as user pki_op, who has the client cert installed. It is assumed that :ref: Configuration of serverPKI has been completed.

4.1 Setting up encrypted key storage

Create a new key pair for encryption of cert keys in the DB.:

```
ssh-keygen -t ed25519 -f db_encryption_key.pem
# Find a secure place and configure its path in config parameter.
# Convert database into key encryption state:
operate_serverPKI --encrypt-keys
```

4.2 Creating our first local certificate

Create meta data in the DB:

```
# su -l pki_op -c "psql -h db1 -p 2222 -U pki_op serverpki"
serverpki=> set search_path to pki,dd;
serverpki=> select * from add_cert('test.com', 'server', 'local', 'ec', false, 'www.
↪test.com', NULL, NULL, NULL, NULL, NULL);
                  add_cert
--------------------------------------------
 (server,test.com,local,,www.test.com,,,,,)
(1 row)
serverpki=> \q
```

Now issue one cert:

```
# su -l pki_op  -c "/usr/local/py_venv/test/bin/python3 /usr/local/py_venv/test/bin/
↪operate_serverPKI -C -d -o test.com"
[operateCA started with options all debug verbose create ]
[1 certificates in configuration]
[----------- 1   test.com        local   False   None    server]
[altname:www.test.com    disthost:       jail:   place:]
[tlsaprefixes of test.com: {}]
[Selected certificates:
['test.com']]
[Creating certificates.]
%No CA cert found. Creating one.
[Please enter passphrase for new CA cert (ASCII only).]
passphrase:
[Please enter it again.]
passphrase:
[CA cert serial 1 with 4096 bit key, valid until 2027-06-05T17:07:22.818955 created.]
[Hash is: 20639CDB63F6A470141F4697919D71EAC85619B09C4056638A92BF43A4BD489F]
[Serial of new certificate is 7523957]
[Creating key (2048 bits) and cert for server test.com]
[Certificate for server test.com, serial 2740072, valid until 2018-05-18T17:07:23.
↪498130 created.]

 # psql -h db1 -p 2222 -U dba postgres
 serverpki=> set search_path to pki,dd;
 SET
 serverpki=# select * from inst;
  id |    name    | state  |         not_before          |          not_after          |     ␣
↪                          hash                              |          updated
 ----+----------+--------+----------------------------+----------------------------+-----
↪------------------------------------------------------------+----------------------
↪------
   1 | Local CA | issued | 2017-05-07 17:07:22        | 2027-06-05 17:07:22        |␣
↪20639CDB63F6A470141F4697919D71EAC85619B09C4056638A92BF43A4BD489F | 2017-05-08␣
↪17:06:48.654368
   2 | test.com | issued | 2017-05-07 17:07:23.4981 | 2018-05-18 17:07:23.49813 |␣
↪EBB7CCBEDD38496D3D979C48E9183E1C1E7CC875740BB1711375C248A055E517 | 2017-05-08␣
↪17:06:48.654368
 (2 rows)
```

4.3 Creating our first Let's Encrypt certificate

Create Letsencrypt account:

```
su -l pki_op -c '/usr/local/py_venv/pki_op_p38/bin/operate_serverPKI -v --register'
[Using config file /usr/local/py_venv/pki_op_p38/etc/serverpki.conf]
[operateCA [pki_op-0.9.10] started with options register verbose ]
[43 certificates and CAs ['Local CA'] in DB]
[Registering a new Let's Encrypt Account.
 With URI:https://acme-staging-v02.api.letsencrypt.org
 and e-mail admin@example.org]
Candango Automatoes 0.9.4. Manuale replacement.



You're about to register a new account with e-mail admin@example.org as contact.␣
↪Continue? [Y/n] Y
```

(continues on next page)

(continued from previous page)

```
Generating a new account key. This might take a second.
  Key generated.
Registering...
  Retrieving terms of agreement ...
  This server requires you to agree to these terms:
    https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf
Agreed? [Y/n] Y
Account https://acme-staging.api.letsencrypt.org/acme/reg/12345678 created.
Wrote account to account.json.

What next? Verify your domains with 'authorize' and use 'issue' to get new␣
↪certificates.
```

Last message can be ignored (its meaningless with serverPKI).

Create meta data in the DB:

```
# su -l pki_op -c "psql -h db1 -p 2222 -U pki_op serverpki"
serverpki=> set search_path to pki,dd;
serverpki=> select * from add_cert('martin-frankowski.de.zone', 'server', 'LE', 'NULL
↪', NULL, NULL, NULL, NULL, NULL);
                  add_cert
--------------------------------------------
 (martin-frankowski.de.zone,LE,,,,,,,)
(1 row)

serverpki=> \q
```

Now authorize fqdn and issue one cert:

```
# su -l pki_op  -c "/usr/local/py_venv/test/bin/python3 /usr/local/py_venv/test/bin/
↪operate_serverPKI -C -d -o martin-frankowski.de"
[operateCA started with options debug only_cert(martin-frankowski.de) verbose create ]
[3 certificates in configuration]
[----------- 3    martin-frankowski.de     LE       False    None     server]
[altname:        disthost:       jail:    place:]
[tlsaprefixes of martin-frankowski.de: {}]
[Selected certificates:
['martin-frankowski.de']]
[Creating certificates.]
[Requesting challenge for martin-frankowski.de.]
[Calling zone_and_FQDN_from_altnames()]
[/usr/local/etc/namedb/master/signed/martin-frankowski.de]
[zones: {'martin-frankowski.de': ['martin-frankowski.de']}]
[fqdn: martin-frankowski.de]
[Writing RRs: ['_acme-challenge.martin-frankowski.de.  IN TXT
↪"i2DtFJ7qT8cWyvIKbcBGLFupLiEkmODHZtK1kFYq7JI"\n']]
[Updating SOA: zone file /usr/local/etc/namedb/master/signed/martin-frankowski.de/
↪martin-frankowski.de.zone]
[Updating SOA: SOA before and after update:
                                2017051002      ; Serial number
                                2017051101      ; Serial number]
[Reloading nameserver]
server reload successful
[martin-frankowski.de: Waiting for DNS propagation. Checking in 10 seconds.]
[]
[martin-frankowski.de: waiting for verification. Checking in 5 seconds.]
```

(continues on next page)

(continued from previous page)

```
[Authorization lasts until 2017-06-10 08:21:35+00:00]
[martin-frankowski.de: OK! Authorization lasts until 2017-06-10T08:21:35Z.]
[Updating SOA: zone file /usr/local/etc/namedb/master/signed/martin-frankowski.de/
↪martin-frankowski.de.zone]
[Updating SOA: SOA before and after update:
                                2017051101      ; Serial number
                                2017051102      ; Serial number]
[Reloading nameserver]
server reload successful
[1 fqdn(s) authorized. Let's Encrypt!]
[Creating key (2048 bits) and cert for server martin-frankowski.de]
[Requesting certificate issuance from LE...]
[Certificate issued. Valid until 2017-08-09T07:22:00]
[Hash is: 7C5B315103626D76C2AB14343176F50805A1C94E9CEEE442BCEEC7C8C092B505]

# su -l pki_op -c "psql -h db1 -p 2222 -U pki_op serverpki"
serverpki=> set search_path to pki,dd;
serverpki=# select * from certs;
 Subject |       Cert Name        | Type  | authorized |   Alt Name    | TLSA | Port |␣
↪Dist Host | Jail | Place
---------+----------------------+-------+------------+--------------+------+------+---
↪---------+------+-------
 CA      | Lets Encrypt CA      | LE    |            |              |      |      | ␣
↪          |      |
 CA      | Local CA             | local |            |              |      |      | ␣
↪          |      |
 server  | martin-frankowski.de | LE    | 2017-06-10 |              |      |      | ␣
↪          |      |
 server  | test.com             | local |            | www.test.com |      |      | ␣
↪          |      |
(4 rows)

Time: 5,400 ms
serverpki=# select * from inst;
 id |         name          | state  |         not_before         |          not_after      ␣
↪     |                                hash                                |          ␣
↪updated
----+----------------------+--------+--------------------------+----------------------
↪-----+------------------------------------------------------------------+-----------
↪-----------------
  1 | Local CA             | issued | 2017-05-07 17:07:22      | 2027-06-05 17:07:22 ␣
↪     | 20639CDB63F6A470141F4697919D71EAC85619B09C4056638A92BF43A4BD489F | 2017-05-
↪08 17:06:48.654368
  2 | test.com             | issued | 2017-05-07 17:07:23.4981 | 2018-05-18 17:07:23.
↪49813 | EBB7CCBEDD38496D3D979C48E9183E1C1E7CC875740BB1711375C248A055E517 | 2017-05-
↪08 17:06:48.654368
  3 | Lets Encrypt CA      | issued | 2016-05-23 22:07:59      | 2036-05-23 22:07:59 ␣
↪     | A99C1B71DA32ADD9429714F71E740AFDC543C4F7F012A748D24A789B8BF3D6C7 | 2017-05-
↪11 08:21:21.487583
  4 | martin-frankowski.de | issued | 2017-05-11 07:22:00      | 2017-08-09 07:22:00 ␣
↪     | 7C5B315103626D76C2AB14343176F50805A1C94E9CEEE442BCEEC7C8C092B505 | 2017-05-
↪08 15:34:20.582733
(4 rows)
```

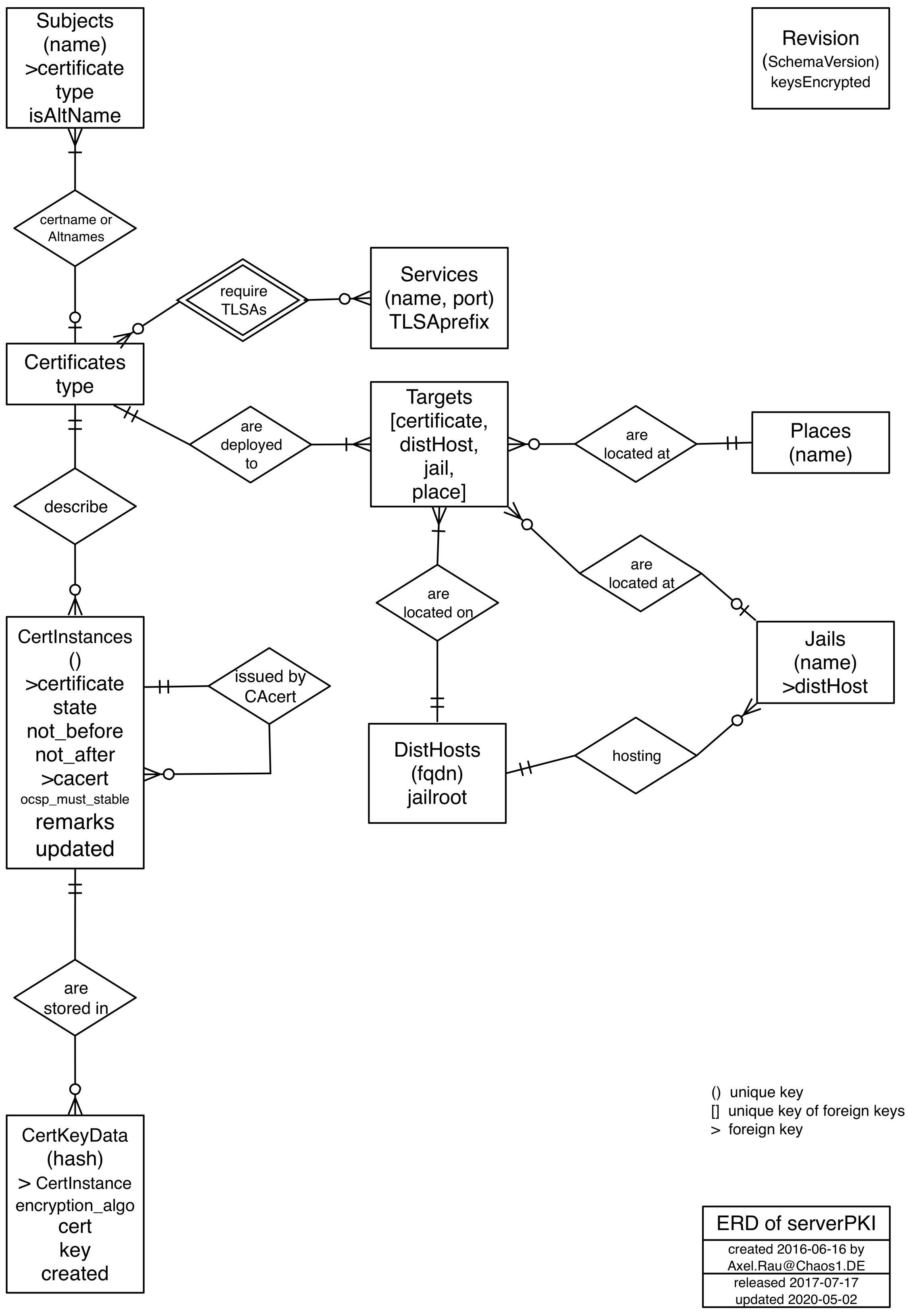
CHAPTER 5

The database

5.1 Model

- The entity relation diagram shows 10 entities, related to certificates and their deployment. The normalized schema has rules and triggers to ensure integrity.
- Common columns - All relations have the following columns:
  - id - synthetic primary key
  - created - date and time of tuple creation
  - updated - date and time of last tuple update
  - remarks - arbitrary text
- columns, which together must be unique are in **bold**

This is the entity relation diagram:


Subjects
(name)
>certificate
type
isAltName
Revision
(SchemaVersion)
keysEncrypted
certname or Altnames
require TLSAs
Services
(name, port)
TLSAprefix
Certificates
type
Targets
[certificate,
distHost,
jail,
place]
are deployed to
are located at
Places
(name)
describe
are located at
are located on
Jails
(name)
>distHost
CertInstances
()
>certificate
state
not_before
not_after
>cacert
ocsp_must_stable
remarks
updated
issued by CAcert
DistHosts
(fqdn)
jailroot
hosting
are stored in
CertKeyData
(hash)
> CertInstance
encryption_algo
cert
key
created
() unique key
[] unique key of foreign keys
> foreign key
ERD of serverPKI
created 2016-06-16 by
Axel.Rau@Chaos1.DE
released 2017-07-17
updated 2020-05-02

5.2 Tables

- **Subjects** - holds all the subject names
  - **name** - name of subject
  - type - subject type, one of
    - 'server' - server subject
    - 'client' - client (or personal) subject
    - 'CA' - certificate authority
    - 'reserved' - type of a placeholder, initially loaded
  - isAltName - true if subject is an alternate name
  - *certificate* - reference to Certificates
- **Certificates** - one entry per defined certificate (holds cert meta data)
  - type - type of certificate, one of
    - LE - to be issued by Let's Encrypt CA
    - local - local cert (to be issued by local CA)
  - disabled - true means: Do not issue/create or distribute this cert.
  - authorized_until:
    - if type is 'LE': Needing new authorization with Let's Encrypt via DNS challenge after this date
    - if type is 'local': date and time of last mail to admin, to ask him to issue a new local cert
  - encryption_algo - encryption algorithm to be used by certs issued in the future, one of
    - rsa
    - ec
    - rsa plus ec
  - ocsp_must_staple - if true then the OCSP staple protocoll will be required by the cert (and server must be configured to support this)
- **Certinstances** - issued certificate instances.
  - state - state of instance (see *State Table*), one of
    - reserved - being issued
    - issued - cert is issued (or renewed)
    - prepublished - cert published in DNS (vis TLSA RR) prior to usage
    - deployed - cert is distributed and in use by server
    - revoked - cert is revoked
    - expired - cert is expired
    - archived - cert is archived (will be removed soon)
  - not_before - start date and time for cert usage
  - not_after - end date and time for cert usage
  - *certificate* - reference to cert in Certificates

  - *cacert* - reference to cacert instance in Certinstances, describing CA which issued this cert
  - ocsp_must_staple - True, if this instance requires OCSP must staple

There may be more than one tuple per cert type, if cacerts are renewed.

Here is the state transition diagram:

Triggers for state transitions and actions

init	during cert.Certificate.__init__()
issued local	issue_local.issue_local_cert(cert-meta) done
issue LE	issue_lLE.issue_LE_cert(cert-meta) done
distributed	certdist.deployCerts(certs) done
dis'd_TLSA	certdist.deployTLSA(certs) done

Actions

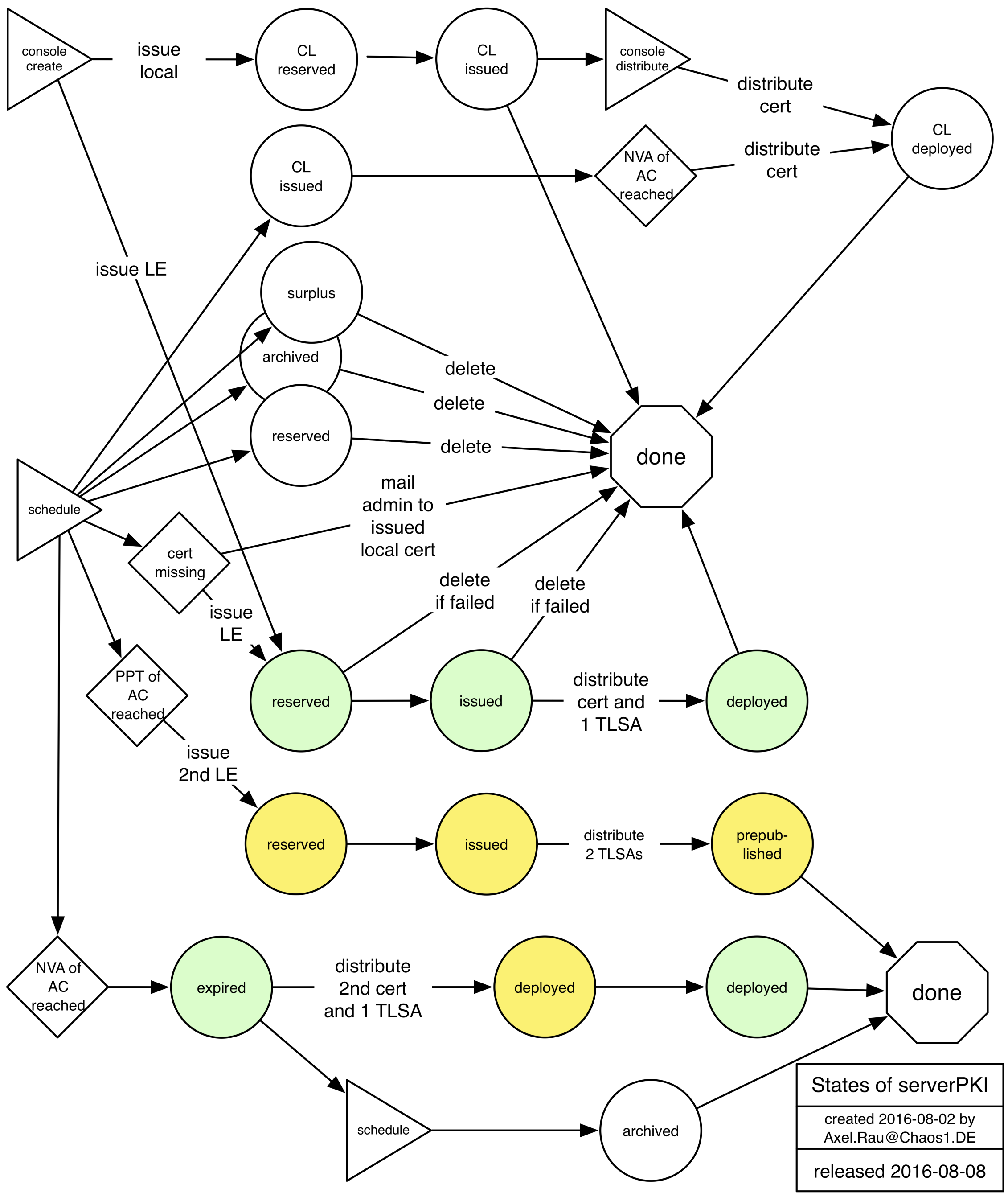
delete	delete cert instance in DB
issue_local	issue_local.issue_local_cert(cert-meta)
issue_LE	issue_lLE.issue_LE_cert(cert-meta)
distribute	certdist.deployCerts(certs)
distr_TLSA	certdist.deployCerts(certs)

Dates, Times and Time Deltas

NVB	not valid before (issue date)
NVA	not valid after
RNT(D)	renew time (delta)
PPT(D)	pre-publish time (delta)

Certificate roles

AC	active certificate	
FC	future certificate	(LE only)
CL	local certificate	🟢
CLE	LE certificate	🟡



- **CertKeyData** - the cert/key material (one tuple per algorithm).
  - **encryption_algo** - encryption algorithm, used with this cert (unique together with certinstance)
    - rsa
    - ec
  - cert - the certificate in binary PEM format
  - key - the key in binary PEM format (encrypted, if DB encryption in use)
  - **hash** - the binascii presentation of the SHA256 hash of the certificate
  - **certinstance** - reference to cert in Certinstances (unique together with encryption_algo)
- **Services** - stores service and port combinations for TLSA RR
  - **name** - name of service
  - **port** - tcp/udp port number of service
  - TLSAprefix - named zone resource record entry with place holder for hash, something like:

    _443._tcp.{}. 3600 IN TLSA 3 0 1
- **Certificates_Services** - junction relation between Certificates and Services
  - **certificate** - reference to cert in Cerificates
  - **service** - reference to service in Services
- **Jails** - One row describes one jail. A jail is a hosted entity on FreeBSD's lightweight virtualization environment. serverPKI connects to the jail host (Disthost) and places certs and keys on the jail, using the filesystem view of the host.
  - **name** - name of jail
  - *disthost* - reference to the disthost, hosting the jail in Disthosts
- **Disthosts** - One row per host to which cert and key should be distributed.
  - **FQDN** - fully qualified domain name of disthost
  - jailroot - optional path to root of jails on disthost. If empty, no jails are on this disthost.
- **Places** - Place, where to deploy cert deployment details, related to one cert / disthost (or jail) combination.
  - **name** - name of place
  - cert_file_type - one of
    - 'cert only' - deploy only cert, no key
    - 'separate' - cert and key are in separate file
    - 'combine key' - cert and key are combined in one file
    - 'combine cacert' - cert is combined with cacert (intermediate if LE), key is in separate file
    - 'combine both' - cert is combined with both key and cacert
  - cert_path - absolute path of cert directory with placeholder '{}' of login
  - key_path - absolute path of key, if different from cert_path
  - uid - let key file be owened by uid
  - gid - let key file be owned by gid
  - mode - mode of key file if different from 0o400

  - chownboth - set owner of cert file to that of key file
  - pglink - link cert / key file to postgresql.crt / postgresql.key
  - reload_command - command to reload service after distribution of cert/key. In case of jail, '{}' is the placeholder for the jail name.
- **Targets** - binds one place, disthost/jail to a certificate
  - **distHost** - references distHost
  - **jail** - references jail
  - **place** - references place
  - **certificate** - references certificate
- **Revision** - holds revision of schema and key encryption state of DB
  - schemaVersion - Version of database schema
  - keysEncrypted - True, if keys are encrypted

5.3 Views

Some views simplify common queries. For each view the result columns are listed.

- **certs** - display meta information about a certificate
  - Subject - *Subject type*
  - Cert Name - *Subject name*
  - Type - *Type of certificate*
  - algo - *Cert encryption algorithm*
  - ocsp_ms - *Cert ocsp_must_staple attribute*
  - authorized - *authorized until*
  - Alt Name - *Alternative cert name*
  - TLSA - *Service name*
  - Port - *Service port number*
  - Dist Host - *Disthost name*
  - Jail - *Jail name*
  - Place - *Place name*
- **certs_ids** - like certs, but include primary keys of referenced tables
  - c_id - cert id
  - s1_id - subject id of none-altname subject
  - Subject Type - *Subject type*
  - Cert Name - *Subject name*
  - Type - *Cert type*
  - algo - *Cert encryption algorithm*
  - ocsp_ms - *Cert ocsp_must_staple attribute*

  - authorized - *authorized until*
  - s2_id - subject id of Alternative cert name subject
  - Alt Name - *Alternative cert name*
  - s_id - service id
  - TLSA - *Service name*
  - Port - *Service port number*
  - t_id - target id
  - d_id - disthost id
  - FQDN - *Disthost name*
  - j_id - jail id
  - Jail - *Jail name*
  - p_id - place id
  - Place - *Place name*
- **inst** - display certificate instances (one row per issued cert instance per algorithm)
  - id - serial of cert instance
  - name - *Subject name*
  - type - *Cert type*
  - state - *State of instance*
  - cacert - reference to cacert instance in Certinstances, describing CA which issued this cert
  - ocsp_must_staple - if true then the OCSP staple protocol will be required by the cert
  - not_before - *Start date for cert usage*
  - not_after - *End date for cert usage*
  - encryption_algo - *Cert encryption algorithm*
  - hash - *Hash of cert*

5.4 Functions

Functions are provided for common operations to abstract foreign key handling. All arguments are text (mostly case insensitive [=citext]), exceptions are mentioned (e.g. boolean), to omit an argument, use *null*. Functions may be called with select in psql:

```
serverpki=> select * from add_cert('test.com', 'server', 'local', 'ec', false, 'www.
↪test.com', NULL, NULL, NULL, NULL, NULL);
                  add_cert
--------------------------------------------
 (server,test.com,local,,www.test.com,,,,,)
(1 row)
serverpki=> \q
```

- **add_cert** - add a new cert to the database
  - the_name - *Subject name*

  - the_subject_type - *Subject type*
  - the_cert_type - *Cert type*
  - the_encryption_algo - *Cert encryption algorithm*
  - must_staple - if true then the OCSP staple protocoll will be required by the cert
  - the_altname - optional *Alternative cert name*
  - the_tlsa_name - optional *Service name*
  - the_tlsa_port - optional *Service port number*
  - the_disthost_name - optional :ref: *Name of disthost*
  - the_jail - optional *Jail name*
  - the_place - optional *Place name*
- **remove_cert** - delete a cert **and all issued cert instances with there CertKeyData from the database**
  - the_cert_name - *Subject name*
- **add_altname** - add an alternative name to an existing cert in the database
  - the_cert_name - *Subject name* to identify the cert, to which the altname should be added
  - the_altname - *Alternative cert name* to add
- **remove_altname** - remove an alternative name from the database
  - the_altname - *Alternative cert name* to be removed
- **add_service** - add an *existing service* to a certificate
  - the_cert_name - *Subject name* to identify the cert, to which the service should be added
  - the_service_name - *Service name*
  - the_port - *Service port number*
- **remove_service** - remove a *service* from a certificate
  - the_cert_name - *Subject name* to identify the cert, from which the service should be removed
  - the_service_name - *Service name*
  - the_port - *Service port number*
- **add_target** - add a *target* to a certificate
  - the_name - *Subject name* to identify the cert, to which the target should be added
  - the_disthost_name - *Disthost name* to identify the *target*
  - the_jail - optional *Jail name* to identify the *target*
  - the_place - optional *Place name* to identify the *target*
- **remove_target** - remove a *target* from a certificate
  - the_cert_name - *Subject name* to identify the cert, from which the target should be removed
  - the_disthost_name - *Disthost name* to identify the *target*
  - the_jail - optional *Jail name* to identify the *target*
  - the_place - optional *Place name* to identify the *target*

CHAPTER 6

Operation

Operation of the PKI is divided into

- Management of cert configuration, which is done via psql (PostgreSQL command line utility) because configuration is stored in a database. This meta data describes things like subject-, alt- name(s), subject- and cert- type, deployment target (host, jail and path), server reload command and DNS TLSA info (service and port).
- Management of cert instances of configured certs like issue, renewal, distribution, publishing and consolidation happens via the operate_serverPKI utility

6.1 Management of configuration

6.1.1 Creating and deleting Disthosts

Certs may be distributed to *Disthosts*. *Disthosts* are referenced by *Jails* and *Targets*.

Example of creating and deleting a *Disthost*:

```
pki_op=# INSERT INTO disthosts (fqdn, jailroot) values('host-with-jails.on.domain', '/
↪usr/jails');
INSERT 0 1
Time: 269,814 ms
pki_op=# INSERT INTO disthosts (fqdn) values('host-without-jails.on.domain');
INSERT 0 1
Time: 180,044 ms
pki_op=# DELETE FROM disthosts WHERE fqdn in ('host-with-jails.on.domain', 'host-
↪without-jails.on.domain');
DELETE 2
Time: 30,975 ms
pki_op=#
```

6.1.2 Creating and deleting Jails

Certs may be distributed to *Jails* on *Disthosts*. *Jails* are referenced by *Targets*.

Example of creating and deleting a *Jail*:

```
pki_op=# SELECT * FROM disthosts WHERE fqdn = 'host-with-jails.on.domain';
 id |            fqdn              |  jailroot  |           updated            |           ␣
↪created            | remarks
----+-----------------------------+------------+-----------------------------+-----------
↪-----------------+---------
 19 | host-with-jails.on.domain | /usr/jails | 2016-07-30 13:48:57.442189 | 2016-07-
↪30 13:48:57.431786 |
(1 row)

Time: 15,472 ms
pki_op=# INSERT INTO jails (name, disthost) VALUES('my_service_jail', 19);
INSERT 0 1
Time: 78,444 ms
pki_op=# DELETE FROM jails WHERE name = 'my_service_jail';
DELETE 1
Time: 18,563 ms
```

Note: A SELECT is used first to find the id of the required *Disthost*.

6.1.3 Creating and deleting of other objects

Functions are provided to create other objects.

6.2 Management of cert instances

These are the command line options. Arguments are in capital letters:

```
 Usage: operate_serverPKI [options]

Server PKI 0.9.11

Options:
  -h, --help            show this help message and exit

  Actions to issue and replace certificates.:
    -C, --create-certs  Scan configuration and create all certs, which are not
                        disabled or excluded. State will be "issued" of
                        created certs. Action modifiers may be used to select
                        a subset of certs to act on.
    -r REMAINING_DAYS, --renew-local-certs=REMAINING_DAYS
                        Scan configuration for local certs in state deployed
                        which will expire within REMAINING_DAYS days. Include
                        these certs in a --create-certs operation. If combined
                        with "--distribute-certs", do not create certs, but
                        instead distribute certs, which would expire within
                        REMAINING_DAYS days and are issued no longer than
```

(continues on next page)

(continued from previous page)

```
                        REMAINING_DAYS in the past.
  -S, --schedule-actions
                        Scan configuration and schedule necessary actions of
                        selected certs/hosts. This may trigger issuence or
                        distribution of certs/TLSA-RRS. With this options "--
                        create-certs" and "--distribute-certs" are ignored.
                        Any state transitions may happen

Actions to deploy or export certificates and deploy or delete DNS TLSA resource␣
↪records.:
  -D, --distribute-certs
                        Scan configuration and distribute (to their target
                        host) all certs which are in state "issued" and
                        currently valid and not disabled or excluded. Changes
                        state to "deployed". Corresponding TLSA RR are also
                        installed, if not suppressed with --no-TLSA-records-
  -K, --consolidate-certs
                        Consolidate targets to be in sync with DB. This
                        affects certs in state "deployed"  and effectively re-
                        distributes certs.
  -T, --consolidate-TLSAs
                        Consolidate TLSA-RR to be in sync with DB. This
                        affects certs in state "deployed" or "prepublished".
  -R, --remove-TLSAs  Remove TLSA-RRs i.e. make them empty.
  -E CERT_SERIAL, --export-cert-and-key=CERT_SERIAL
                        Export certificate and key with CERT_SERIAL to work
                        directory. CERT_SERIAL may be obtained from DB (column
                        "id" with command operate_serverPKI -n -v) This action
                        may not be combined with other actions.

Action modifiers, to select certificates or disthosts to act on.:
  -a, --all             All certs in configuration should be included in
                        operation, even if disabled.
  -i CERT_TO_BE_INCLUDED, --include=CERT_TO_BE_INCLUDED
                        Specify, which cert to be included, even if disabled,
                        in list of certs to be created or distributed. Is
                        cumulative if multiple times provided.
  -e CERT_TO_BE_EXCLUDED, --exclude=CERT_TO_BE_EXCLUDED
                        Specify, which cert to be excluded from list of certs
                        to be created or distributed. Is cumulative if
                        multiple times provided.
  -o ONLY_CERT, --only=ONLY_CERT
                        Specify from which cert(s) the list of certs to be
                        created or distributed. Is cumulative if multiple
                        times provided.
  -s SKIP_HOST, --skip-disthost=SKIP_HOST
                        Specify, which disthosts should not receive
                        distributions. Is cumulative if multiple times
                        provided.
  -l ONLY_HOST, --limit-to-disthost=ONLY_HOST
                        Specify, which disthosts should receive distributions
                        only (others are excluded). Is cumulative if multiple
                        times provided.
  -N, --no-TLSA-records
                        Do not distribute/change TLSA resource records.

Maintenance and administrative actions.:
```

(continues on next page)

(continued from previous page)

```
    -X, --encrypt-keys  Encrypt all keys in DB.Configuration parameter
                        db_encryption_key must point at a file, containing a
                        usable passphrase.
    -Y, --decrypt-keys  Replace all keys in the DB by their clear text
                        version.Configuration parameter db_encryption_key must
                        point at a file, containing a usable passphrase.
    -I, --issue-local-CAcert
                        Issue a new local CA cert, used for issuing future
                        local server/client certs.
    -Z, --register      Register a new account at LetsEncrypt, This action may
                        not be combined with other actions.
    -n, --check-only    Do syntax check of configuration data. Produce a
                        listing of cert meta and related cert instances if
                        combined with  --verbose. Listed certs may be selected
                        with --only.
    -d, --debug         Turn on debugging.
    -q, --quiet         Be quiet on command line. Do only logging. (for cron
                        jobs).
    -v, --verbose       Be more verbose.
    -f CONFIG_FILE, --config_file=CONFIG_FILE
                        Path of an alternate configuration file.
```

This script is run by cron (typically once an hour) like:

```
pki_op  /usr/local/py_venv/PKI_OP_published/bin/operate_serverPKI -S -q -a
```

The action –renew-local-certs=REMAINING_DAYS displays a table with certs and attributes, which would be renewed, if combined with the “-n” option, Like so:

```
+---------+-----------+-------+------------+----------+------+------+-------------+--
↪-----+---------+
| Subject | Cert Name |  Type | authorized | Alt Name | TLSA | Port |  Dist Host   | ␣
↪Jail |  Place   |
+---------+-----------+-------+------------+----------+------+------+-------------+--
↪-----+---------+
|  client |  gal1_op  | local |    None    |   None   | None | None | bh4.lrau.net |␣
↪erdb4 | gal1_db |
+---------+-----------+-------+------------+----------+------+------+-------------+--
↪-----+---------+
```

Listing of cert meta and related cert instances may be obtained with the combination of –check-only with –verbose. Listed certs may be selected with –only, Like so:

```
# su -l pki_dev -c "/usr/local/py_venv/pki_dev_p37/bin/python /usr/local/py_venv/pki_
↪dev_p37/bin/operate_serverPKI  -v -n -o -a"
[operateCA [serverPKI-0.9.9] started with options all check_only verbose config_file(␣
↪/Users/ajr/Projects/SERVICES/serverPKI/serverPKI/tests/conf/serverpki.conf) ]
[3 certificates and CAs ['Local CA'] in DB]
[No syntax errors found in configuration.]
+---------+-----------+-------+------+-----------+------------+----------+------+-----
↪-+-------------------------+------+---------+
| Subject | Cert Name |  Type | Algo | OCSP m st | authorized | Alt Name | TLSA |␣
↪Port |         Dist Host         | Jail |  Place   |
+---------+-----------+-------+------+-----------+------------+----------+------+-----
↪-+-------------------------+------+---------+
|  client |  client1  | local | rsa  |   False   |    None    |   None   | None |␣
↪None | axels-imac.in.chaos1.de | None | place_1 |
```

(continues on next page)

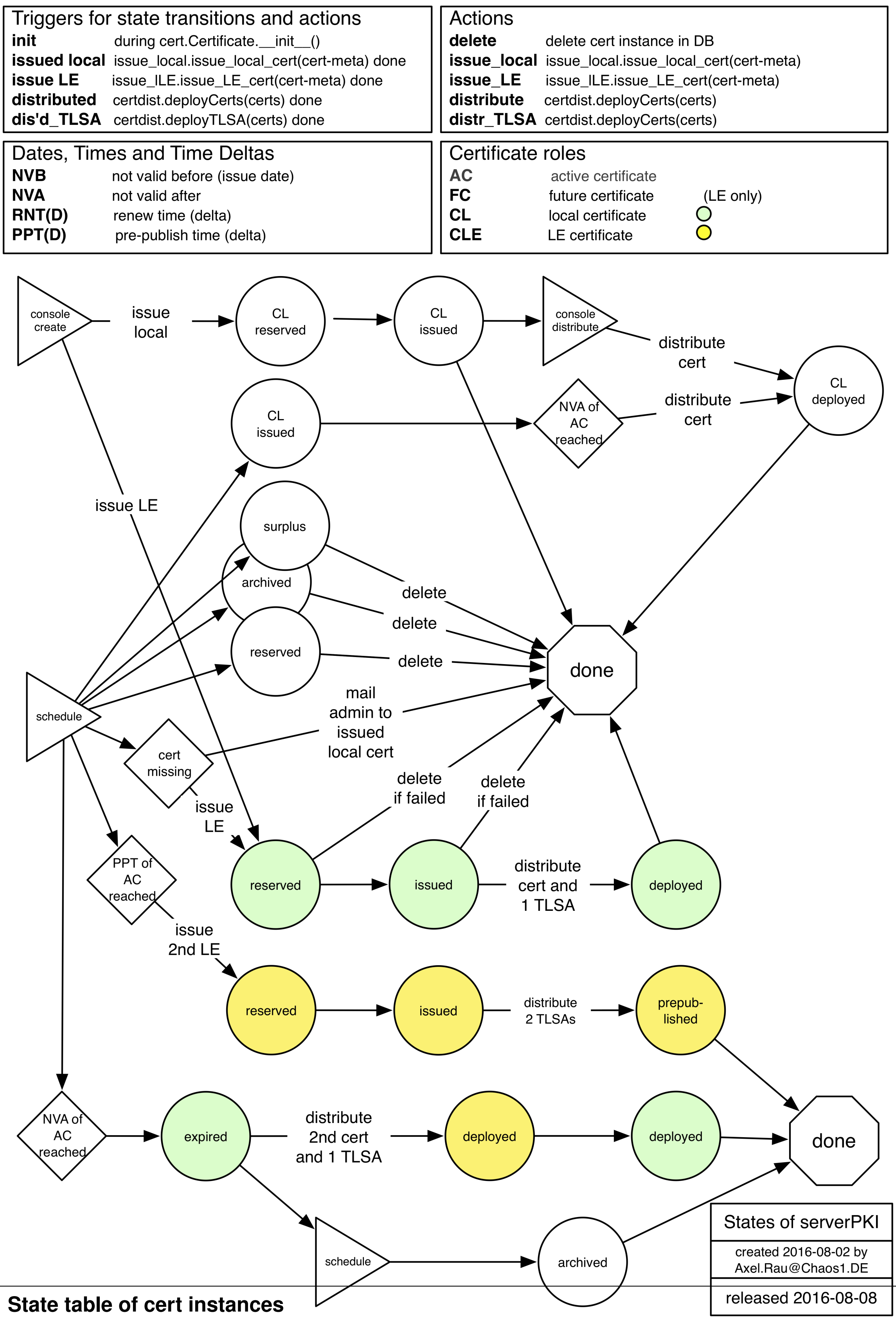
(continued from previous page)

```
|    CA    |  Local CA | local | rsa  |    False    |     None     |    None    | None |␣
↪None |            None             | None |    None   |
|    CA    |  No cert  | local | rsa  |    False    |     None     |    None    | None |␣
↪None |            None             | None |    None   |
+----------+-----------+-------+------+-----------+------------+----------+------+-----
↪-+------------------------+------+---------+

+---------+-----------+-------+--------+-------+-----------+--------------------+-----
↪----------------+------+-----------------------------------------------------------
↪------+---------------------------+
| Serial | Cert Name |  Type | State   | CI CA | OCSP m st |       not before       |     ␣
↪ not after       | ALGO |                                  Hash                      ␣
↪       |          updated          |
+---------+-----------+-------+--------+-------+-----------+--------------------+-----
↪----------------+------+-----------------------------------------------------------
↪------+---------------------------+
|   3    |  Local CA | local | issued |   3    |    False    | 2020-07-04 00:00:00 |␣
↪2030-08-02 00:00:00 | rsa  |␣
↪CF32D82E6A0D36258AAF05CBE62E4834C7EA254FEC5E0A88B08B3C773F2D5989 | 2020-07-05␣
↪13:34:37.768547 |
|   4    |  Local CA | local | issued |   4    |    False    | 2020-07-04 00:00:00 |␣
↪2030-08-02 00:00:00 | rsa  |␣
↪69DF3EAB1FD2D55A9BA42C8F590757B63EFDCF63D16EB7F83EC02B6ACC5A5280 | 2020-07-05␣
↪13:34:38.527877 |
+---------+-----------+-------+--------+-------+-----------+--------------------+-----
↪----------------+------+-----------------------------------------------------------
↪------+---------------------------+
```

Displayed serial number may be used for exporting a key pair with –export.

6.3 State table of cert instances



Triggers for state transitions and actions

init	during cert.Certificate.__init__()
issued local	issue_local.issue_local_cert(cert-meta) done
issue LE	issue_lLE.issue_LE_cert(cert-meta) done
distributed	certdist.deployCerts(certs) done
dis'd_TLSA	certdist.deployTLSA(certs) done

Actions

delete	delete cert instance in DB
issue_local	issue_local.issue_local_cert(cert-meta)
issue_LE	issue_lLE.issue_LE_cert(cert-meta)
distribute	certdist.deployCerts(certs)
distr_TLSA	certdist.deployCerts(certs)

Dates, Times and Time Deltas

NVB	not valid before (issue date)
NVA	not valid after
RNT(D)	renew time (delta)
PPT(D)	pre-publish time (delta)

Certificate roles

AC	active certificate	
FC	future certificate	(LE only)
CL	local certificate	🟢
CLE	LE certificate	🟡

CHAPTER 7

Indices and tables

- genindex
- search

Index

A

add_altname, 25
add_cert, 24
add_service, 25
add_target, 25

C

Certificates, 19
Certificates.authorized_until, 19
Certificates.disabled, 19
Certificates.encryption_algo, 19
Certificates.ocsp_must_staple, 19
Certificates.type, 19
Certinstances, 19
Certinstances.not_after, 19
Certinstances.not_before, 19
Certinstances.state, 19
CertKeyData, 21
CertKeyData.cert, 21
CertKeyData.encryption_algo, 21
CertKeyData.hash, 21
CertKeyData.key, 21
certs, 23
certs_ids, 23
created, 17

D

Disthosts, 22
Disthosts.FQDN, 22
Disthosts.jailroot, 22

E

entity relation diagram, 17
ERD, 17

F

Functions, 24

I

id, 17
inst, 24

J

Jails, 22
Jails.disthost, 22
Jails.name, 22

M

Management of cert instances, 28
Management of configuration, 27
Model, 17

O

operate_serverPKI -register, 14
Operation, 25

P

Places, 22
Places.cert_file_type, 22
Places.cert_path, 22
Places.chownboth, 22
Places.gid, 22
Places.key_path, 22
Places.mode, 22
Places.name, 22
Places.pglink, 22
Places.reload_command, 22
Places.uid, 22

R

remarks, 17
remove_altname, 25
remove_cert, 25
remove_service, 25
remove_target, 25
Revision, 23
Revision.keysEncrypted, 23
Revision.schemaVersion, 23

S

services, 22
Services.name, 22
Services.port, 22
Services.TLSAprefix, 22
Subject.certificate, *see* Certificates
Subjects, 19
Subjects.certificate, 19
Subjects.isAltName, 19
Subjects.name, 19
Subjects.type, 19

T

Tables, 18
Targets, 23
Targets.certificate, 23
Targets.distHost, 23
Targets.jail, 23
Targets.place, 23
The database, 16

U

updated, 17

V

Views, 23