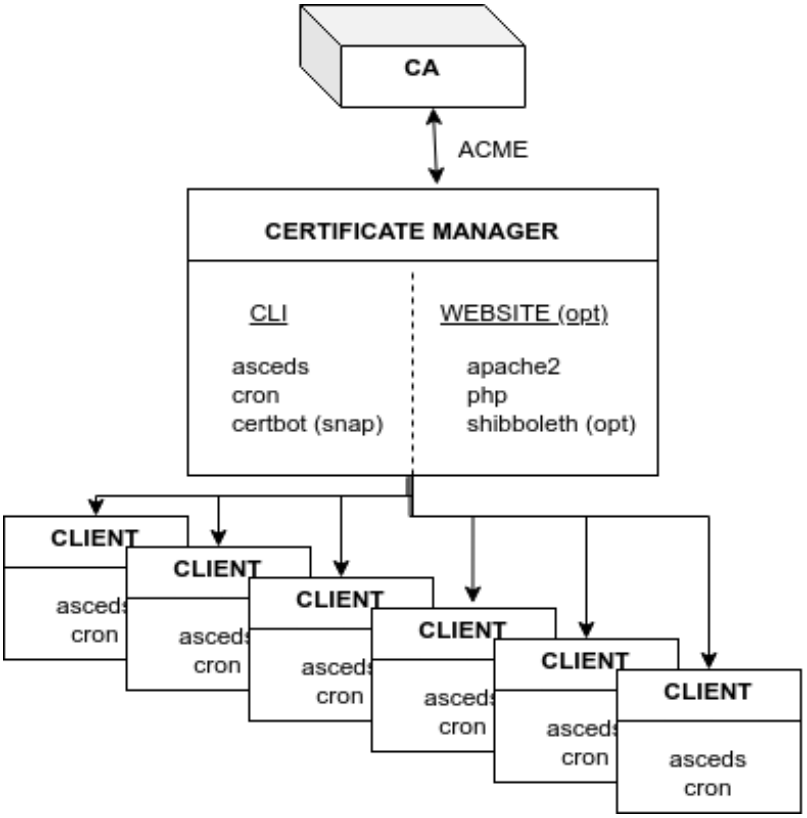
ASCEDS

Release 0.1

Florin Manlache

May 09, 2022

CONTENTS



Automated SSL Certificate Distribution System (ASCEDS) is a distribution system for certificates which can be obtained from a provider offering an ACME interface. The system defines a site containing:

- one certificate manager which can get SSL certificates for wilcard hostnames in a set of domains through certbot;
- a set of managed clients getting automatic certificate renewal or reconfiguration of SANs; services using certificates are reconfigured/restarted automatically upon renewal;
- a web interface handling certificates for managed or unmanaged clients.

ASCEDS was developed by Florin Manolache at Carnegie Mellon University for interfacing certbot with InCommon SSL certificates, and automatically distributing them to the servers on the network.

ASCEDS is released under GPLv2 or any later version.

Check out the *Usage* section for further information, including how to install the project.

Note: This project is under active development.

CHAPTER

ONE

CONTENTS

1.1 Installation

Run

- install and configure dependencies: snap, certbot, apache2
- install asceds:

```
echo "deb [trusted=yes] https://cert.mcs.cmu.edu/debian ./" | sudo tee -a /etc/apt/
↪sources.list
sudo apt update
sudo apt install asceds
```

Case 1: Certificate manager

- initialize the certificate manager: asceds-certmanager-setup -w
- configure the client (Case 2 below)
- customize the website:
  - edit the apache2 site configuration file typically in /etc/apache2/sites-available/asceds.conf
  - decide on the authentication and add the right .htaccess (see examples in /usr/share/doc/asceds/examples/site-*)
  - edit the web php configuration in /usr/share/asceds/etc/config.php (see examples in /usr/share/doc/asceds/examples/php-etc)
  - enable the website in apache2: a2ensite asceds
- add authorizied domains: asceds-authorized-domains -a <domainname>
- add authorized users: asceds-web-user -a <username>

Case 2: Client

- install asceds (no need for snap or certbot)
- get info: certificate manager name, root password/sudo account/password
- asceds-init -n -s <your.cert.manager>

  -n –> generates a new certificate

1.2 Usage

- The main body of documentation is installed by default in /usr/share/doc/asceds/doc
- Each script displays its function and usage with the "-h" option.
- The asceds command lists documentation and status for the current computer.
- A paper describing the general idea and the functional structure is in the public directory of the web interface, by default in /usr/share/asceds/html/public/asceds.pdf or at https://<your.cert.manager>/public/asceds.pdf

1.2.1 Certificates

on cert manager

on client

1.2.2 Action Chains

Setup of certificate manager, one per site

(root/sudo access, mailserver, certbot, webserver)
Cert_manager: asceds-certmanager-setup [-s <cert_manager>]
Cert_manager: asceds-update-siteconfig -s

1. **Initial setup or OS upgrades of clients**
    a. **managed clients, root/sudo@cert_manager ssh access:**

        Client: asceds-init -n [-s <cert_manager_URL>]
            Client: asceds-update-siteconfig -c
            Client: asceds-sans-update -l [-r]
            Cert_manager: asceds-client-setup -n -s -c <client>
                Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
                Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
            Client: asceds-services
                Client: asceds-service-reconfig -s <service>
                    Client: asceds-update-siteconfig -q -c

    b. **managed clients, no root/sudo@cert_manager ssh access, web interface:**

        Client: asceds-init [-s <cert_manager_URL>]
            Client: asceds-update-siteconfig -c
            Client: asceds-sans-update -l [-r]
        Cert_manager web interface: asceds_cert.php
            Cert_manager: asceds-test-return -c <client>
            Cert_manager: asceds-web-propagate -c <client>
                Cert_manager: asceds-client-setup -n -c <client>
                    Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
                    Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
        Client: asceds-services

Client: asceds-service-reconfig -s <service>
Client: asceds-update-siteconfig -q -c

c. **unmanaged clients, web interface:**

Cert_manager web interface: request_cert.php
Cert_manager: asceds-web-unmanaged -n -c <client>
Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
Cert_manager web interface: host_check.php
Cert_manager: download certificate files
Client: reconfigure services

2. Push automatically renewed certificates to clients

a. **fully managed clients:**

Automatic renewal by certbot (systemd)
Cert_manager crontab: asceds-cert-propagate
Cert_manager: asceds-propagate-certbot
Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
Client crontab: asceds-cert-refresh
Client: asceds-service-reconfig -t
Client: asceds-update-siteconfig -q -c

b. **privately managed clients, root/sudo@cert_manager ssh access:**

Automatic renewal by certbot (systemd)
Cert_manager crontab: asceds-cert-propagate
Cert_manager: asceds-propagate-certbot
Cert_manager: asceds-mailto-alarm
Client: allow r/w ssh access for asceds account from Cert_manager
Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
Client: asceds-service-reconfig -t
Client: asceds-update-siteconfig -q -c

c. **privately managed clients, no root/sudo@cert_manager ssh access, web interface:**

Automatic renewal by certbot (systemd)
Cert_manager crontab: asceds-cert-propagate
Cert_manager: asceds-propagate-certbot
Cert_manager: asceds-mailto-alarm
Client: allow r/w ssh access for asceds account from Cert_manager
Cert_manager web interface: asceds_cert.php
Cert_manager: asceds-test-return -c <client>
Cert_manager: asceds-web-propagate -c <client>
Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
Client: asceds-service-reconfig -t
Client: asceds-update-siteconfig -q -c

d. **unmanaged clients, web interface:**

Cert_manager crontab: asceds-cert-propagate
Cert_manager: asceds-propagate-certbot
Cert_manager: new certificate email notification
Cert_manager web interface: host_check.php

Cert_manager: download certificate files
Client: reconfigure services

3. SANs refresh (manual sequence)

   a. **managed clients, root/sudo@cert_manager ssh access:**

      Client: asceds-sans-update [-r] [-a <hostname>,...] [-d <hostname>,...]
      Cert_manager: asceds-client-setup -n -c <client>
          Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
          Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
      Client: asceds-service-reconfig -t
          Client: asceds-update-siteconfig -q -c

   b. **managed clients, no root/sudo@cert_manager ssh access, web interface:**

      Client: asceds-sans-update [-r] [-a <hostname>,...] [-d <hostname>,...]
      Cert_manager web interface: asceds_cert.php
          Cert_manager: asceds-test-return -c <client>
          Cert_manager: asceds-web-propagate -c <client>
              Cert_manager: asceds-client-setup -n -c <client>
                  Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
                  Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
      Client: asceds-service-reconfig -t
          Client: asceds-update-siteconfig -q -c

   c. **unmanaged clients, web interface:**

      Cert_manager web interface: request_cert.php
          Cert_manager: asceds-web-unmanaged -n -c <client>
              Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
      Cert_manager web interface: host_check.php
          Cert_manager: download certificate files
      Client: reconfigure services

4. Certificate revoke

   a. **managed clients, root/sudo@cert_manager ssh access:**

      Cert_manager: asceds-certbot-revoke -e -c <client>

   b. **any client, web interface:**

      Cert_manager web interface: revoke_cert.php
          Cert_manager: asceds-web-unmanaged -r -c <client>
              Cert_manager: asceds-certbot-revoke -e -c <client>

5. Site config file management

   a. **create:**

      Cert_manager: asceds-certmanager-setup [-s <certmanager_cname>]
          Cert_manager: asceds-update-siteconfig -s ->
              asceds-utils/asceds-build-siteconf

   b. **update:**

      Cert_manager: asceds-update-siteconfig -s ->
          asceds-utils/asceds-build-siteconf

c. **initialize on managed clients:**

Client: asceds-init [-s <certmanager>]
Client: asceds-update-siteconfig -c ->
asceds-utils/asceds-parse-siteconf

d. **automatic propagate trough http on fully managed clients:**

Cert_manager crontab: asceds-cert-propagate
Certmanager: asceds-propagate-certbot
Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>
Client crontab: asceds-cert-refresh
Client: asceds-service-reconfig -t
Client: asceds-update-siteconfig -q -c

e. **manual propagate on privately managed clients:**

Client: allow r/w ssh access for asceds account from Cert_manager
Client: asceds-update-siteconfig -c

6. Web interface

Apache authenticated web interface on Cert_manager for:

a. **Unmanaged client, generate new certificate:**

Cert_manager web interface: request_cert.php
Cert_manager: asceds-web-unmanaged -n -c <client>
Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
Cert_manager web interface: host_check.php
Cert_manager: download certificate files
Client: reconfigure services

b. **Managed client, propagate ASCEDS data (including generate certificate):**

Cert_manager web interface: asceds_cert.php
Cert_manager: asceds-test-return -c <client>
Cert_manager: asceds-web-propagate -c <client>
Cert_manager: asceds-client-setup -n -c <client>
Cert_manager: asceds-certbot-gencert -e -c <client> -s <sans>
Cert_manager: sudo -u asceds -i asceds-send-cert -c <client>

c. **Revoke certificate (any client):**

Cert_manager web interface: revoke_cert.php
Cert_manager: asceds-web-unmanaged -r -c <client>
Cert_manager: asceds-certbot-revoke -e -c <client>

1.2.3 Config Files

Asceds default (overwritten by upgrades): /usr/lib/asceds/etc
Local setup (preserved between upgrades): /etc/asceds/

Executed in the following order: | local –> variables for asceds setup on the local computer | site –> config variables common to the site, distributed by the certificate manager | hostconf –> config variables describing the host OS and services | certbot –> config variables for certbot management

First all the default config files are sourced, then the local setup to overwrite the default values.

Scripts set sourcing order in ${ASCEDSCONF.
Refreshing is done through asceds-utils/asceds-source-etcconf

asceds-site.conf:

```
# alarm email address
ALARM=""
# certificate manager hostname
ASCEDSCRTMGR=""
# wget -> for self signed certificates on the certificate manager,
# add '--no-check-certificate' to wget
WGET='/usr/bin/wget'
# asceds ssh public key
ASCEDSSSHPUBKEY=''
# list of encryption algorithms (rsa,ecc)
ASCEDSENCLIST="rsa"
```

1.2.4 Cron Jobs

certificate manager

asceds-cert-propagate

asceds-propagate-certbot -q check if new key/cert copy in ~asceds/<client>/ asceds-send-cert -q -c <client>

asceds-web-actions asceds-web-unmanaged -q -n -c <client> –> newcert asceds-web-unmanaged -q -r -c <client> –> revoke asceds-web-propagate -q -c <client> –> generate and propagate

client

asceds-cert-refresh

asceds-service-reconfig -q -t check if service reconfig trigger ~asceds/.asceds-reconf asceds-service-reconfig ${RM} ~asceds/.asceds-reconf

1.2.5 EMAIL

Email capabilities (mailutils+postfix) are required for certificate managers and optional for clients.

Email ALARM (lib/asceds-utils: asceds-mailto-alarm):

If ${ALARM} is empty, no email will be sent. Email messages are sent to the ALARM address (admin or ticketing system)

If:

a. **expired cert** cert_manager: asceds-propagate-certbot –> asceds-expiration-test local: asceds-service-reconfig –> asceds-expiration-test

b. systems with noreturn policy (ro like pi-kvm) –> asceds-propagate-certbot

c. certs cannot be moved (network or ssh access problems) –> asceds-send-cert

d. TBI (not sure if needed) if errors –> create digest

Email messages are sent to the requestor (all from the certificate manager):

- NO_ASCEDS in cert.conf file
- REQUESTED_BY in web request file

If:

a. **revoked certificate for unmanaged client** asceds-web-unmanaged -r –> if running the queue (not with -c)

b. **generate/renew certificate for unmanaged client** asceds-web-unmanaged -n –> if running the queue (not with -c)

c. **certificate was automatically renewed ny certbot for unmanaged client** asceds-propagate-certbot –> for unmanaged clients

d. certificate was generated/renewed through the website request for managed client

   asceds-web-propagate –> if running the queue (not with -c)

1.2.6 Scripts

asceds:

```
status/info and general description
(interactive only), runs as root/sudo, ${ASCEDSBINDIR}/
asceds [-h] [-d] [-l] [-c] [-m <string>] [-t <type>] [-u <username>]
Shows a summary of current status of certificate files and managed services
-h --> display usage and exit
-d --> describes in detail asceds utilities and usage
-l --> shows summary of warnings/errors from log files
-c --> show details about clients
-m <string> --> name or SANs of the client matches <string> (no regexp)
-t <type> -> type of the client matches type (unman*, pman*, fman*)
-u <username> --> lists unmanaged clients requested by web <username>;
```

(continues on next page)

(continued from previous page)

```
              works only with "-t unman"
-m and -t work only with -c, otherwise they are ignored
```

asceds-authorized-domains:

```
manages domains which are authorized by the CA to get certificates
using this server's EABKID/EABHMACKEY
asceds-authorized-domains [-h] [-a <domainname>,...] [-d <domainname>,...]
-a <domainname>[,<domainname>,...] adds comma separated domain names
-d <domainname>[,<domainname>,...] deletes comma separated domain names
If no argument, displays current configuration and goes into interactive mode;
Deletes domain names from ${ASCEDSETCDIR}/asceds-certbot.conf
  ${ASCEDSWEBDIR}/etc/users.php and ${ASCEDSWEBDIR}/html/public/.htaccess.
  Deletes clients containing the removed domains from certbot and from
  asceds on the cert manager; do not revoke the existing certificates.
Adds domain names to ${ASCEDSETCDIR}/asceds-certbot.conf
  ${ASCEDSWEBDIR}/etc/users.php and ${ASCEDSWEBDIR}/html/public/.htaccess.
```

asceds-certmanager-setup:

```
setup the certificate manager
(interactive only), runs as root/sudo, ${ASCEDSCBDIR}/
asceds-certmanager-setup [-h] [-s <certmanager>]
-h --> display usage and exit
-s <certmanager> --> FQDN of the certificate manager (if different from hostname)
Checks/fixes network configuration including hosts, postfix, openssl.cnf.
Checks for certbot; if not found, offer self signed certbot (for test sites)
  or requests certbot install.
Checks for mailer; if not found, requests mailer install.
Collects certbot credentials and authorized domains in asceds-certbot.conf.
Creates asceds user ssh keys and the local authorized_keys.
Creates/updates the site config file asceds-site.conf.
Creates symlinks to certobot-related ASCEDS scripts.
Sets cron asceds-cert-propagate for driving asceds-propagate-certbot
  to push renewed cert files to ~asceds/<client> and propagate
  them to managed clients through asceds-send-cert.
Web interface setup:
  Gets info: organization name, website url, request execution methods;
  Customizes config.php and asceds.php;
  Copies and reconfigs asceds-site-apache2.conf;
  Initializes request log file;
  Builds directory structure for the website;
  Creates symlink to make site config file available by wget to clients;
  Creates ssh keys for root requests by ssh;
  Sets asceds-web-actions crontab for running the request queues generated
      by the web interface.
```

asceds-client-setup:

```
gets client parameters, generates new certificate, copies files to the client;
to be run on the cert manager as root/sudo through asceds-init on the client;
(interactive only), runs as root/sudo, ${ASCEDSBINDIR}/
asceds-client-setup [-h] [-n] [-s] [-q] -c <client_name>
```

(continues on next page)

(continued from previous page)

```
-h --> display usage and exit
-c <client_name> --> client to be setup; print usage if missing
-q --> no question asked; sends output to the logs
-n --> generate new certificate
-s --> clean old keys of the client in ~asceds/.ssh/known_hosts
Scp client's cert.conf to ${ASCEDSHOMEDIR}/${DOMAIN_NAME}_cert.conf
Validity checks
  DOMAIN_NAME --> CLIENT_DOMAIN_NAME
  SANS_LIST --> CLIENT_SANS_LIST
Generates new certificate if required: asceds-certbot-gencert
Creates flag for transferring the certificates .newcerts
Transfers cert back: asceds-send-cert -c <client_name>
```

asceds-init:

```
initializes certificate client
(interactive only), runs as root/sudo, ${ASCEDSBINDIR}/
asceds-init [-h] [-n] [-r] [-f] [-s <certmanager>]
-h --> display usage and exit
-n --> generate new certificate
-r --> set noreturn policy, client is read-only
-f --> site using self signed certificates
-s <certmanager> --> the site certificate manager
Checks/fixes network configuration including hosts, postfix, openssl.cnf.
Builds certificate manager URL:
  if no cert manager is mentioned in the command line, ask for it;
Retrieves the site config file from the certificate manager,
  parse and create {ASCEDSETCDIR}/asceds-site.conf.
Sets the certificate encoding in {ASCEDSETCDIR}/asceds-hostconf.conf.
Generates certificate configuration in cert.conf.
Copies logrotate asceds config file.
Creates renewal crontab: asceds-cert-refresh
Work done on root/sudo@certmanager if access available,
      otherwise prints instructions and exits:
  Runs setup script: asceds-client-setup [-n] -s -c ${CLIENT_DOMAIN_NAME}
Selects and configures services which depend on certificates and
      have reconfig scripts: asceds-services
Removes the reconf trigger ${ASCEDSHOMEDIR}/.asceds-reconf
```

asceds-sans-update:

```
change SAN values, recreate certificate, reconfigure services
(interactive only), runs as root/sudo, ${ASCEDSBINDIR}/
asceds-sans-update [-h] [-l] [-r] [-a <hostname>,...] [-d <hostname>,...]
-h --> display usage and exit
-r --> set noreturn policy if client is read-only
-l --> local mode (appropriate for asceds-init)
-a <hostname>[,<hostname>,...] adds comma separated SAN values
-d <hostname>[,<hostname>,...] deletes comma separated SAN values
Refresh cert.conf according to the current version of asceds;
If no -a/-d, displays current configuration and goes into interactive mode;
Deletes SAN values from /etc/hosts, /etc/postfix/main.cf, openssl.cnf
```

(continues on next page)

(continued from previous page)

```
Checks DNS records of SANs to be added; configures with the host IP address
Adds SAN values to /etc/hosts, /etc/postfix/main.cf, openssl.cnf
Refreshes ~asceds/cert.conf
Provides instructions to manually complete the action.
```

asceds-send-cert:

```
copies new site config and certificate files to the clients;
it does not check for the type of client, attepts transfer anyway.
(interactive + script), runs as asceds, ${ASCEDSBINDIR}/
used by asceds-init, also when certificates are generated
(renewal, add-sans, del-sans)
asceds-send-cert [-h] [-q] -c <client_name>
-h --> display usage and exit
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-c <client_name> --> transfer the certificate files to client_name
Checks for a new site config file (flagged by <client_name>:.newsiteconf).
SCP the new site config file to asceds@<client_name>:.
Removes the trigger for uploading the site config file.
Sets the trigger for site config file update <client_name>:.site-reconf.
Checks if new certificate files exist (flagged by <client_name>:.newcerts).
SCP the new certificate files to asceds@<client_name>:certs/.
Removes the trigger for uploading new certificate files.
Sets the trigger for service reconfiguration <client_name>:.asceds-reconf.
If the client is unreachable, sets a flag ~asceds/down/<client_name> and
  emails to ALARM; if client is reachable and the flag exists, removes it.
```

asceds-service-reconfig:

```
updates the site config file if needed,
copies cert filess to the right locations and restarts services using them
(interactive + script), runs as root/sudo, ${ASCEDSBINDIR}/
needs to be run by trigger every time the cert files are changing
    cron job <-- ~asceds/.asceds-reconf
(init, renewal, add-sans, del-sans)
asceds-service-reconfig [-q] [-t] [-h] [-s <service>]
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-t --> reconfigure services only if trigger file .asceds-reconf exists
-s service --> reconfigures the <service> only (one service);
               in cron mode, -s options are ignored
-h --> display usage and exit
Updates the site config file using the trigger ${ASCEDSHOMEDIR}/.site-reconf;
Checks if the available certificate is not stale
  (expired or expiring in the next 5 days);
  if the certificate is stale, refreshes the site configuration file
  to allow pushing new certificate files which are blocked when
  the asceds ssh keys change on the certificate manager;
  if it is almost expired, send message to ALARM;
Reads parameters (CLIENT_DOMAIN_NAME, CLIENT_SANS_LIST) <-- ~asceds/cert.conf;
Copies cert/key from ~asceds/certs/ --> /etc/ssl ; fix permissions;
Reconfigures/restarts services through ${ASCEDSSERVCONF}/*.sh ;
    to avoid sourcing: rename so it doesn't match *.sh;
```

(continues on next page)

(continued from previous page)

```
    available service templates: ${ASCEDSSERVDIR}/*.sh.proto;
Removes the reconfig trigger file ${ASCEDSHOMEDIR}/.asceds-reconf
```

asceds-services:

```
configure services handled by ASCEDS
(interactive only), runs as root/sudo, ${ASCEDSBINDIR}/
asceds-services [-h] [-a <service>] [-d <service>]
-h --> display usage and exit
-d <service> --> deactivate one service (before activating)
                 multiple [-d <service>] are allowed
-a <service> --> activate or refresh one service,
                 multiple [-a <service>] are allowed,
                 each activated service is reconfigured through
                       asceds-service-reconfig -q -s <service>
Shows available services to be updated by
      scripts in ${ASCEDSSERVDIR}/*.sh.proto;
Shows activated service update scripts;
Shows which local scripts (if any) are different from templates;
Shows which local scripts (if any) have no template;
Deactives local .sh scripts in ${ASCEDSSERVCONF}/, keeping a backup;
Activates local .sh scripts in ${ASCEDSSERVCONF}/ starting from templates;
With no argument, offers a simple interface to activate/deactivate services
      by typing in a space separated list.
To refresh a local script from template, both deactivate and activate it.
```

asceds-update-siteconfig:

```
Configures/updates site config file ${ASCEDSETCDIR}/asceds-site.conf
(interactive + script(on client only)), runs as root/sudo, ${ASCEDSBINDIR}/
asceds-update-siteconfig [-h] [-q] [-s] [-c]
-h --> display usage and exit
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-s --> create/update the site config file on the certificate manager
-c --> reconfigure client (-s and -c are mutually exclusive)
On the certificate manager:
  Creates/updates the site config file ${ASCEDSETCDIR}/asceds-site.conf.
    Allows ASCEDSCRTMGR change only if ran from asceds-certmanager-setup.
    If certificate manager name is a CNAME (not fqdn of localhost):
      checks if it resolves to the same IP address;
      adds it to hosts, postfix, openssl.cnf.
  Refreshes symlink to make site config file available by wget to clients.
  Copies the new site config file into ${ASCEDSHOMEDIR}/<managed_client>/ and
    adds .newsiteconf trigger to all managed clients except cert manager itself.
  On the cert manager, copies the new site config file into ${ASCEDSHOMEDIR}/.
On the client (no prompts in quiet mode):
  Certificate manager URL priority: ${ASCEDSALTCRTMGR}, ${ASCEDSCRTMGR}, input.
  Retrieves the site config file from the certificate manager URL
    to ${ASCEDSHOMEDIR}/asceds-site.conf.
  Parses and creates ${ASCEDSETCDIR}/asceds-site.conf.
  Uses the site ssh public key to create/update authorized_keys.
```

asceds-web-user:

```
Adds/removes/reconfigures web users
asceds-web-user [-h] [-l]
                [-p] [-a <username>] [-d <username>]
                [-e] [-c] [-x] [-m <username>]
-h --> display usage and exit
l --> list current users and their domains, then exit
-p --> use simple auth password file ${ASCEDSWEBDIR}/html/cert/.htpasswd
-a <username> --> adds username to the website;
-d <username> --> deletes user from the website;
-m <username> --> modifies (if -e, -c, or -x present) or deletes/re-adds user;
-e --> modifies email address (works with -m);
-c --> modifies authorized domains (works with -m);
-x --> modifies simple auth password (works with -m if .htpasswd exists);
Options -a/-d/-m are mutually exclusive.
Wildcard ALL gives user authority over all available domains.
Sets/removes list of authorized domains in ${ASCEDSWEBDIR}/etc/users.php
Sets/removes user password for simple auth in ${ASCEDSWEBDIR}/html/cert/.htpasswd
  (if it exists or -p)
If file ${ASCEDSWEBDIR}/html/cert/.htpasswd exists, -p is forced.
<Username> must be valid email address used for sending notifications,
  or a mail alias is set.
User deletion is allowed only if no unmanaged clients are recorded to the user.
```

asceds-certbot-gencert:

```
generates new cert files using certbot
(interactive + script), runs as root/sudo, ${ASCEDSCBDIR}/
asceds-certbot-gencert [-h] [-e] [-d] [-q] -c <client_name> [-s <SAN1>,<SAN2>,...]
Generates new cert files using certbot. Options:
-h --> display usage and exit
-e --> execute the certbot command (just echo the command by default)
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-d --> dry-run (off by default)
-c <client_name> --> name of the computer requesting a certificate
-s <SAN1>,... --> comma-separated alternate names of the computer requesting
                  a certificate not including domain_name
Figures out certbot's options based on the command line input.
Checks if cert manager is authorized to generate requested certificates
  by matching hostname and SANs with the authorized domains.
Generates new certificates using certbot
  Certs/key are generated by certbot and go in /etc/letsencrypt/live/<cert_name>/;
  Checks if a fresh certificate was generated.
  Certs/key are copied to ${ASCEDSHOMEDIR}/<client_name>/ and chown asceds:asceds;
  If client is unmanaged, copies the cert files in the web dir and chown ${ASCEDSWEBUSER}
↪.
```

asceds-certbot-revoke:

```
revoke certs
(interactive + script), runs as root/sudo, ${ASCEDSCBDIR}/
asceds-certbot-revoke [-h] [-e] [-q] [-d] -c <client_name>
-h --> display usage and exit
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
```

(continues on next page)

(continued from previous page)

```
-c <client_name> --> cert for <client_name> will be revoked (one per use)
-e --> execute certbot revoke (just echo the command by default)
-d --> delete the client from asceds and certbot; do not revoke certificate

Checks if certbot is installed and certificate files exist.
Displays status of the client and of the certificate files.
Checks if cert manager is authorized to revoke requested certificate.
Displays certbot command to be executed, or it executes it (if -e).
If -e or -d:
  Removes any certificate files from ~asceds/<client_name>/
      and /etc/letsencrypt/*/client_name>/*
  If client is unmanaged, removes the cert files from the web dir.
```

asceds-propagate-certbot:

```
detect and propagate new certificates based on automatic renewals by certbot
(script by cron, interactive), runs as root/sudo, ${ASCEDSCBDIR}/
asceds-propagate-certbot [-h] [-q] [-l] [-r] [-c <client>]
-h --> display usage and exit
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-l --> keep the certificate local, don't trigger asceds-send-cert
-r --> ignore noreturn policy and try to send the certificate
      (mutually exclusive with -l; needs -c)
-c <client> --> propagate only for <client>
Looks for renewed certs;
Copies renewd certs to ~asceds/<client>/; chown asceds:asceds
If client is unmanaged:
  Copies the cert files in the web dir for download.
  Sends email to requestor for downloading the renewed cert files.
If client is managed:
  Creates transfer cert flag ~asceds/<client>/.newcerts.
  If fully managed client and no -l option,
      or if privately managed client with -r option:
        Sends cert files to the client through asceds-send-cert.
  If privately managed client, sends email to ALARM announcing new cert.
Tries to re-send cert files to fully managed clients if .newcerts is present.
```

asceds-test-return:

```
Tests read/write scp access to asceds@client
(called by asceds_cert.php), ${ASCEDSHOME}/website/
asceds-test-return [-h] -c <client_name>
-h --> display usage and exit
-c <client_name> --> client name to test
Outputs: success = read/write successful
        readonly = read-only access
        noaccess = no ssh access at all
```

asceds-web-unmanaged:

```
Performs actions requested through the web interface for unmanaged clients
(request_cert.php, revoke_cert.php, cron script asceds-web-actions),
${ASCEDSHOME}/website/
```

(continues on next page)

(continued from previous page)

```
asceds-web-unmanaged [-h] [-q] [-n] [-r] [-c <client_name>]
-h --> display usage and exit
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-n --> new certificate actions only
-r --> revoke actions only
-c <client_name> --> act only for <client_name>
If both types of requests (new certificate and revoke) are found,
revoke requests are performed first.
Revoke: requests in ${ASCEDSWEBDIR}/cert_queue/*.rev:
  read .rev file;
  log: date Web user <username> revoked certificate for <hostname>;
  asceds-certbot-revoke -q -e -c <hostname>;
  remove .rev file;
Newcert: requests in ${ASCEDSWEBDIR}/cert_queue/*.cert:
  read .cert file;
  create/adjust ~asceds/<hostname>_cert.conf;
        populate populate ASCEDS_CERT_ID if empty;
  log: date Web user <username> requested certificate for <hostname>
        with SANs <sans>, client type changed to: <ack>;
  asceds-certbot-gencert -e -q -c <hostname> -s <sans>;
  remove .cert file.
```

asceds-web-propagate:

```
Propagates ASCEDS data requested through the web interface for managed clients
(asceds_cert.php, cron script asceds-web-actions), ${ASCEDSHOME}/website/
asceds-web-propagate [-h] [-q] [-c <client>]
-h --> display usage and exit
-q --> no question asked; sends output to the logs in ${ASCEDSLOGDIR}/
-c <client> --> propagate only <client>
Propagate: requests in ${ASCEDSWEBDIR}/cert_queue/*.asceds
If run the queue (by cron), sends notification by email to requestors.
If cert.conf exists and ASCEDS_CERT_ID not empty,
  removes web certificate files.
If ~asceds/<hostname>/.newcerts is present, just sends the certificate files
  through asceds-send-cert -c <client> (for atomic operation);
Else, performs a full setup (generates new certificate and sends it)
  through asceds-client-setup -n -c  <client>.
Removes request file.
```

*-head:

```
Header stub for scripts in the directory:
* save the arguments
* set default locations for config files
* load default config files as requested: local site hostconfig certbot
* load customized config files as requested: local site hostconfig certbot
* load libraries as requested
* echo greetings
* check if logging is possible
* normalize format of the arguments
```

1.2.7 Triggers

Transfer cert files flag:

```
asceds@<certmanager>:~asceds/<hostname>/.newcerts
* signals when new cert files are copied from /etc/letsencrypt/live/<hostname>/
  into ~asceds/<hostname>/ for managed clients.
* created by: asceds-propagate-certbot, asceds-certbot-gencert.
* deleted by: asceds-send-cert if the transfer was successful.
* used by: asceds-propagate-certbot to send certificates to clients;
           asceds-web-propagate: to decide if a new certificate is generated;
               it allows the web interface to propagate automatic renewals or
               to generate and propagate new certificates for managed clients.
```

Reconfigure service flag:

```
asceds@<client>:~asceds/.asceds-reconf
* signals when new cert files were copied successfully from <certmanager>
  and the services need to be reconfigured with the new certificates.
* created by: asceds-send-cert if the transfer was successful.
* deleted by: asceds-init, asceds-service-reconfig with no -s option.
* used by: asceds-service-reconfig to reconfigure services.
```

New site config file flag:

```
asceds@<certmanager>:~asceds/<hostname>/.newsiteconf
* signals when the site config file ${ASCEDSETCDIR}/asceds-site.conf
  was reconfigured and ready to be pushed to managed clients.
* created by: asceds-update-siteconfig -s
               asceds-certmanager-setup (indirectly)
               asceds-propagate-certbot
               asceds-certbot-gencert
* deleted by: asceds-send-cert if the transfer was successful.
* used by: asceds-propagate-certbot to send site config files to clients;
           asceds-web-propagate when/if sending new cert files.
```

Update site config file flag:

```
asceds@<client>:~asceds/.site-reconf
* signals when a new site config file was pushed by the certificate manager,
  parse it, and refresh the client ${ASCEDSETCDIR}/asceds-site.conf
* created by: asceds-send-cert if the client was reachable.
* deleted by: asceds-service-reconfig
* used by: asceds-service-reconfig to update the site config file on clients.
```

1.2.8 Client Type Change

What happens during client type transformations: client types have sense only if the web interface is used; otherwise only managed clients can be handled; so client type conversion should happen as close as possible to the web code.

revoke:

managed -> unmanaged: all certificate files are deleted asceds-web-unmanaged -r -> asceds-certbot-revoke

unmanaged -> managed: all certificate files are deleted asceds-web-unmanaged -r -> asceds-certbot-revoke

generate (for unmanaged):

managed -> unmanaged: request_cert.php -> asceds-web-unmanaged -n -> populate ASCEDS_CERT_ID if empty

propagate (for managed):

unmanaged -> managed: asceds_cert.php ->

asceds-web-propagate -> removes web certificate files if ASCEDS_CERT_ID not empty

1.2.9 Default Directories

ASCEDSHOME="/usr/lib/asceds"
ASCEDSWEBDIR="/usr/share/asceds"
ASCEDSETCDIR="/etc/asceds"
ASCEDSDOCDIR="/usr/share/doc/asceds/doc"
ASCEDSLOGDIR="/var/log/asceds"
ASCEDSWEBHISTFILE="${ASCEDSLOGDIR}/request.history"
ASCEDSHOMEDIR="${ASCEDSETCDIR}/home"
ASCEDSBINDIR="${ASCEDSHOME}/bin"
ASCEDSCONFDIR="${ASCEDSHOME}/etc"
ASCEDSLIBDIR="${ASCEDSHOME}/lib"
ASCEDSCBDIR="${ASCEDSHOME}/certbot"
website shell scripts="${ASCEDSHOME}/website"
ASCEDSSERVDIR="${ASCEDSHOME}/bin/service.d"
ASCEDSSERVCONF="${ASCEDSETCDIR}/service.d"

1.2.10 API Configuration

/etc/asceds/home/cert.conf:

```
# don't return certificate to client?
# empty for fully managed clients, non-empty otherwise
NO_RETURN=''

# fqdn name on the certificate
CLIENT_DOMAIN_NAME=qwes.math.cmu.edu

# SANS list of the certificate
CLIENT_SANS_LIST=nero.math.cmu.edu,smtp.math.cmu.edu,webmail.math.cmu.edu,sattva.math.
↪cmu.edu
```

(continues on next page)

(continued from previous page)

```
# certificate id
# empty for managed clients, non-empty for unmanaged clients
ASCEDS_CERT_ID=''

# should contain the email address or user name of the requestor for notifications
# empty for managed clients, non-empty for unmanaged clients
NO_ASCEDS=''

# client platform
# for managed clients it should be linux (for now)
# for unmanaged clients: unknown
CLIENT_PLATFORM='linux'

# client certificate encoding
# thos should be rsa (default) or ecc
CLIENT_ENC='rsa'
```

1.3 Build

These scripts perform two functions

1. create build for the intended platform
2. transfer build output to cert.mcs.cmu.edu:/usr/share/asceds/<platform>

Note: This project is under active development.

- asceds-build-arch
- asceds-build-arch-dev
- asceds-build-deb
- asceds-build-rpm

  Description: This script can be used to build and release packages for centOS.

  How:

  # Connects to a centOS host centos7.math.cmu.edu (hardcoded) # Uses rsync to sync below dirs

  - ./dist
  - ./pack/REDHAT
  - ./asceds/asceds-build-rpm-dev

  # Calls the asceds-build-rpm-dev to perform the actual build # Transfers the build output to cert.mcs.cmu.edu

- asceds-build-rpm-dev

  Description: This script can be used on centOS machine to build RPM package

  How:

Picks up the version from dist/usr/lib/asceds/etc/version and checks version format # Updates version number in pack/ARCH/PKGBUILD # Removes build/asceds

Dependencies: yum install makepkg

Important Notes

1. The public alias/dir is supposed to be unsecured.
2. If the website is not located at the root of the URL, the link to public/authorized_keys in asceds_cert.php may not work properly (test scenarios).

1.4 Website

*** Login (shibboleth or local) username (determines authorized domains for certs)

*** Helper files: css/cert.css ../etc/asceds.php ../etc/config.php ../etc/users.php header.php footer.php

*** Navigation index.php –> host_check.php [client_type] [=independent] O-> edit_cert.php –> request_cert.php OOO> host_check.php [=asceds] O-> asceds_cert.php –> revoke_cert.php

*** Pages –1. index.php: ASCEDS certificate management system (index.php) Purpose: * Entry point for identifying the client and for the general presentation of the ASCEDS architecture.

Info/downloads: * All domains available to the cert manager * Domains authorized to the user

Input: username

Collects (all mandatory): * client_type: managed/unmanaged –> asceds/independent * hostname: text * domain: select from domains authorized to the user

Output (GET to index.php): client_type, hostname, domain

–2. host_check.php: ASCEDS certificate inquiry Purpose: * Presents information about existing certificate for the client, if any. * Offers links to existing certificate files for unmanaged clients.

Info/downloads: if the client has a current certificate: * certificate validity (effective/expiration) * SANS list * links to existing certificate files for unmanaged clients

Input (GET from index.php,request_cert.php): client_type, hostname, domain

Collects: * client type changing acknowledgement (reguired if any change) –> propagates to logs through request_cert.php,asceds_cert.php ack = same (no change) tonotmanaged (from managed to unmanaged) tomanaged (from unmanaged to managed)

Output1 (POST to edit_cert.php if client_type=independent): hostname, domain, sans, ack, addhost='', adddomain='', delhost=''

Output1 (POST to asceds_cert.php if client_type=asceds): hostname, domain, sans, ack

Output2 (POST to revoke_cert.php): hostname, domain, sans

–3. edit_cert.php: ASCEDS new certificate edit Purpose: * Edits the SANS list for the new certificate

Info/downloads: * Hostname (fqdn) * Current SANS list, as configured so far * Number of hostnames in the SANS list

Input (POST from host_check.php,edit_cert.php): hostname, domain, sans, ack, addhost, adddomain, delhost

Collects: * Host name to add to the SANS list: text * Domain of the host above: select from domains authorized to the user * Full hostname to remove from the SANS list: select from current SANS list elements

Output1 (POST to edit_cert.php): hostname, domain, sans, ack, addhost, adddomain, delhost

Output2 (POST to request_cert.php): hostname, domain, sans, ack

–4. request_cert.php: ASCEDS new certificate create Purpose: * Creates a new certificate immediately (by ssh) or after a while (by cron) –> asceds-web-unmanaged -q -n -c –> asceds-certbot-gencert -q -e -c -s * Copies the certificate files to /keys// * Updates cert.conf info if needed

Info/downloads: * (if by cron) When are the certificate files available for download

Input (POST from edit_cert.php): hostname, domain, sans, ack, addhost, adddomain, delhost

Collects: nothing

Output1 (request file .cert): REQUEST_DATE= CLIENT_DOMAIN_NAME= REQUESTED_BY= CLIENT_SANS_LIST= ACK_CHANGE_TYPE=

Output2 (log file /var/log/asceds/request.history): date, full hostname, username, sans, ack log: Web user requested certificate for with SANS , client type changed to:

–5. revoke_cert.php Purpose: * Revokes current certificate immediately (by ssh) or after a while (by cron) –> asceds-web-unmanaged -q -r -c –> asceds-certbot-revoke -q -e -c

Info/downloads: * (if by cron) Timeline for certificate revocation

Input (POST from host_check.php): hostname, domain, sans

Collects: nothing

Output1 (request file .rev): REQUEST_DATE= CLIENT_DOMAIN_NAME= REQUESTED_BY=

Output2: full hostname, username * log: Web user revoked certificate for

–6. asceds_cert.php Purpose: * Creates certificate files for managed clients; * propagates certificate files to the clients O-> asceds-test-return -c –> asceds-web-propagate q -c –> asceds-client-setup -q -n -c –> asceds-certbot-gencert -q -e -c -s –> asceds-send-cert -q -c

Info/downloads: * (if by cron) Timeline for data propagation

Input (POST from host_check.php): hostname, domain, sans, ack

Collects: nothing

Output1 (request file .asceds): REQUEST_DATE= CLIENT_DOMAIN_NAME= REQUESTED_BY= ACK_CHANGE_TYPE=

Output2 (log file /var/log/asceds/request.history): date, full hostname, username, ack log: Web user propagated asceds data for

Output3 (POST to asceds_cert.php): hostname, domain, sans, ack * loop to fix rw acces to asceds@client

Website accounts (simple auth) login should be valid email of requestors

certadmin@localhost:q1q2q3a4 -> we'll make it admin certadmin2@localhost:q1q2q3a5 -> math,phys,logic certadmin3@localhost:q1q2q3a6 -> chem certadmin4@localhost:q1q2q3a7 -> bio,mbic,mcs certadmin5@localhost:q1q2q3a8 -> phys certadmin6@localhost:q1q2q3a9 -> nothing

htpasswd [-c] .htpasswd certadmin?