ACME

Not just for rockets anymore!

ConFoo 2017
Montreal, Canada

Magnus Hagander
magnus@hagander.net

Image: Kenneth Lu (flickr)

ACME

New ways of blowing things up

Image: wikipedia

Magnus Hagander

- Redpill Linpro
  - Infrastructure services
  - Principal database consultant
- PostgreSQL
  - Core Team member
  - Committer
  - PostgreSQL Europe

A small case study

The environment

- The postgresql.org infrastructure
- Around 65 VMs
  - 5 datacenters (4 countries)
  - 1 cloud (aws)
- Around 0 staff
  - (4-5 with 0 dedicated time, at best)

The environment

- Debian jessie
  - Has been lenny>squeeze>wheezy>
- Custom config management
  - Not puppet/chef/etc
  - Because they sucked at the time
  - And considering problem scope
- (Almost) fully automated

The challenge

- Encrypt everything
  - (well...)
  - https everywhere the obvious
  - Also smtp, imap, pgsql, etc, etc
  - Both public and restricted
- *Certificate management*

The dark ages

- Individual service certificates
  - Manual issuing
  - Manual renewal
- Domain level wildcard certificate
  - For *.postgresql.org
    - Nothing for other domains
  - Shared private keys
  - Still manual

Enter ACME

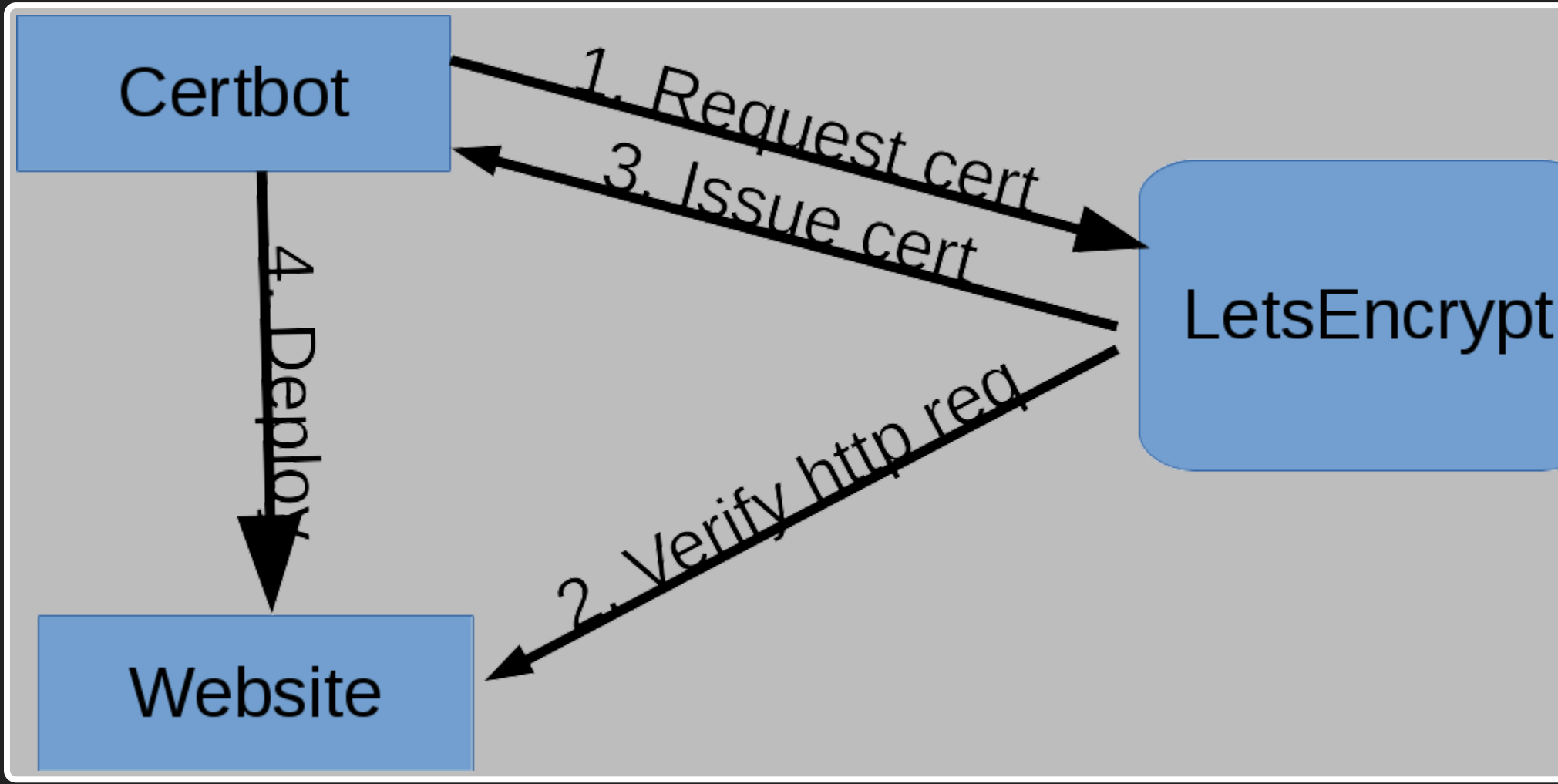
- Automatic Certificate Management Environment
- Best known implementation: LetsEncrypt

LetsEncrypt

- Issues *domain validated* certificates
  - Same as we had before
- Fully automated validation
- Short lifetime (90 days)
  - *Requires* automation

certbot

- Default client for LetsEncrypt



certbot

- Requires exposed http services
- Tries to auto-config webserver
  - SCARY

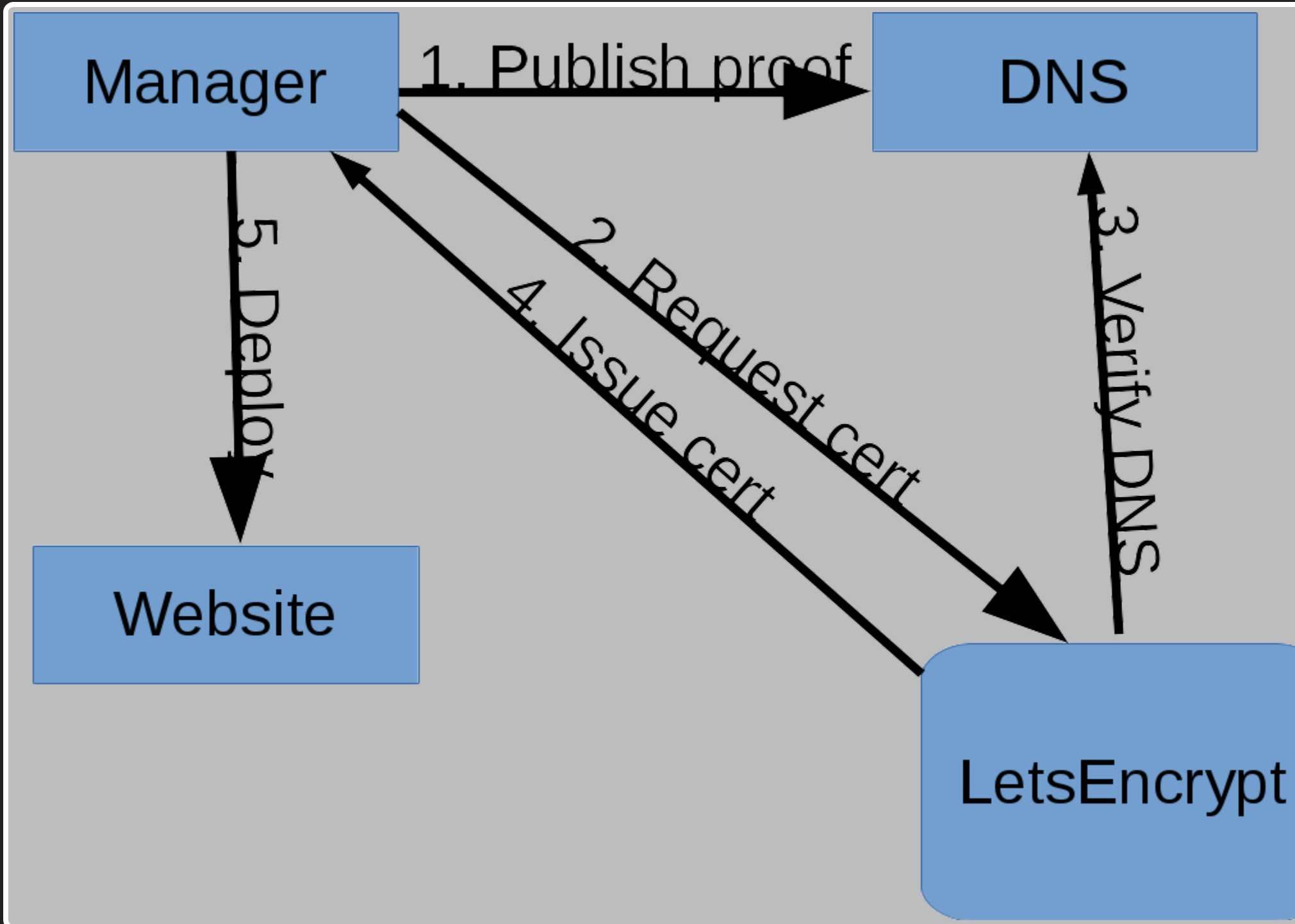
ACME

- Is a protocol
- Not a client
- Multiple ways to verify exists
  - Just not in default client

ACME dns-01

- Issue TXT records in DNS
- Better suited for central management
  - DNS probably already is

ACME dns-01


Manager
1. Publish proof
DNS
2. Request cert
3. Verify DNS
4. Issue cert
5. Deploy
Website
LetsEncrypt

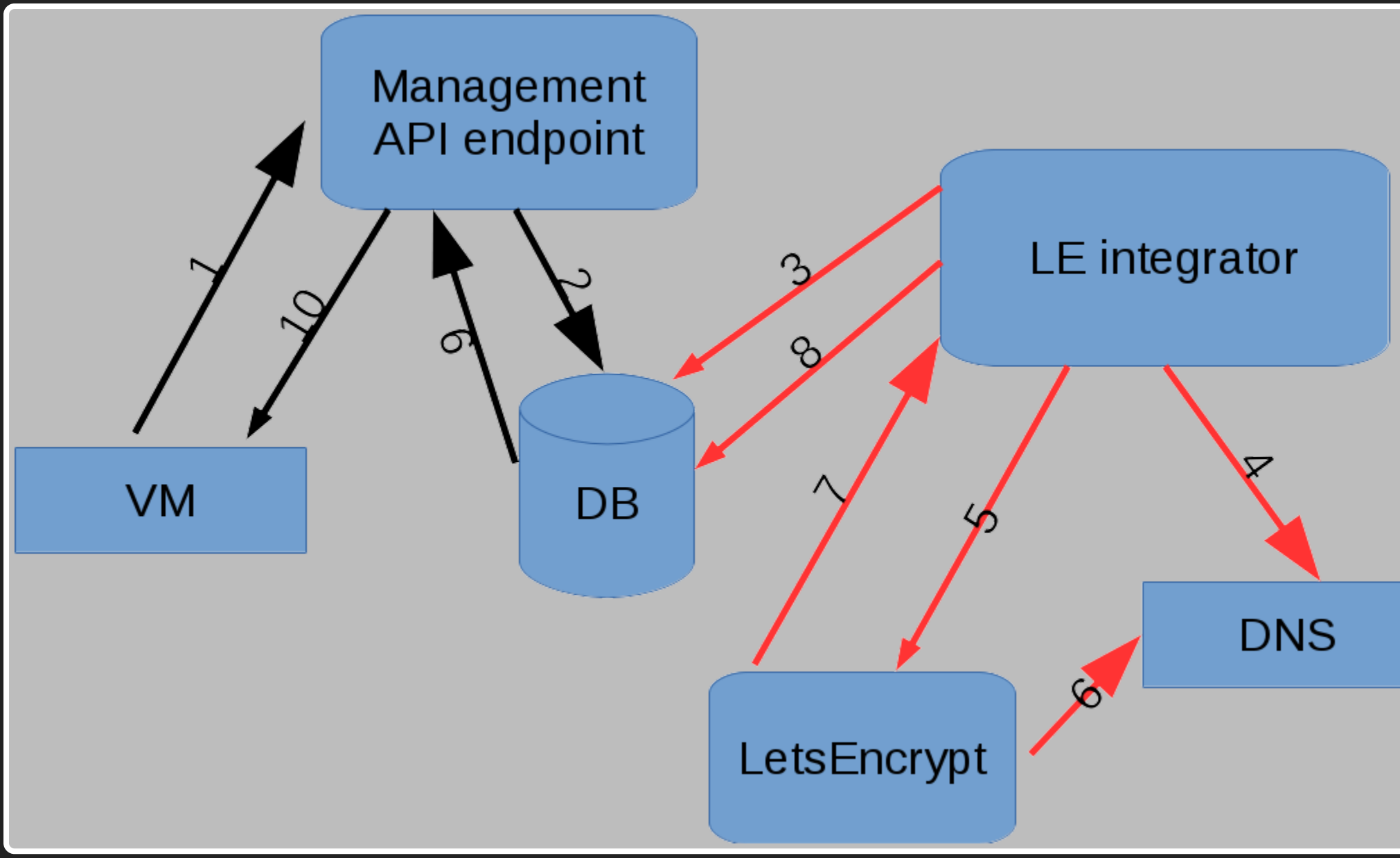
New set of problems

- Centralized key distribution
  - Private keys in one place
  - Not good for security!
- Or distributed access to DNS
  - Doable with dynamic DNS
  - As long as it's controlled

Back to postgresql.org

- Existing simple config management
- Central API
- Client certificate authenticated
- Can be leveraged

ACME in pginfra


Management
API endpoint
LE integrator
VM
DB
DNS
LetsEncrypt
1
2
3
4
5
6
7
8
9
10

ACME in pginfra

PostgreSQL Infrastructure Management

Home › Letsencrypt › Certificates › Add certificate

Add certificate

Primaryname: testhost.postgresql.org

Secondarynames: foo.postgresql.org,bar.postgresql.org

Server: borka

Comment:

Csr:

ACME in pginfra

On the VM

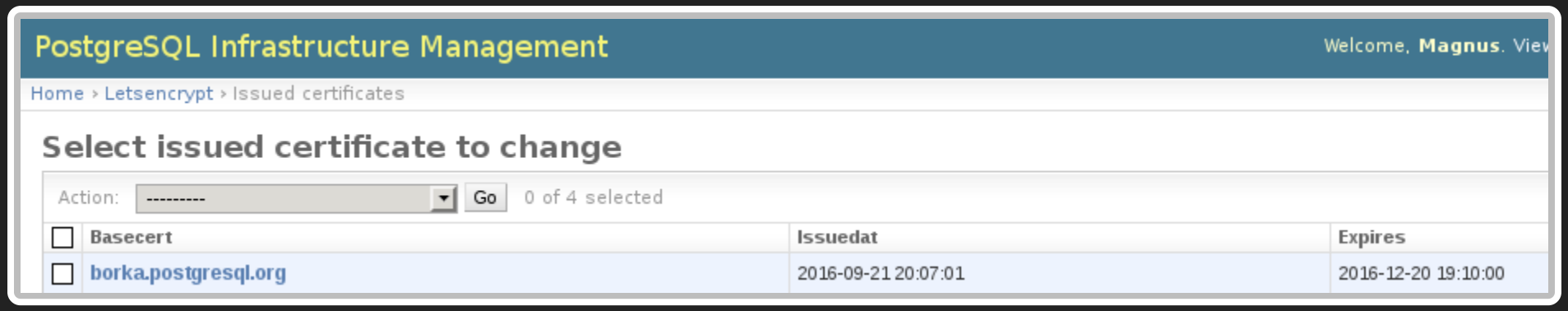
```
... borka pginfra: Completed user and package checks.
... borka pginfra: Creating certificate request for 5-borka.postgr
```

ACME in pginfra

On central server

```
~$ ./letsencrypt_cron.py
Getting challenges for 1 identifiers
Setting up for 1 remaining challenges
Waiting for 8 more records to show up in DNS
Waiting for 8 more records to show up in DNS
Waiting for 4 more records to show up in DNS
Waiting for 2 more records to show up in DNS
Waiting for 1 more records to show up in DNS
All records present in DNS
Waiting for 1 challenges...
Issued certificate for borka.postgresql.org
```

ACME in pginfra



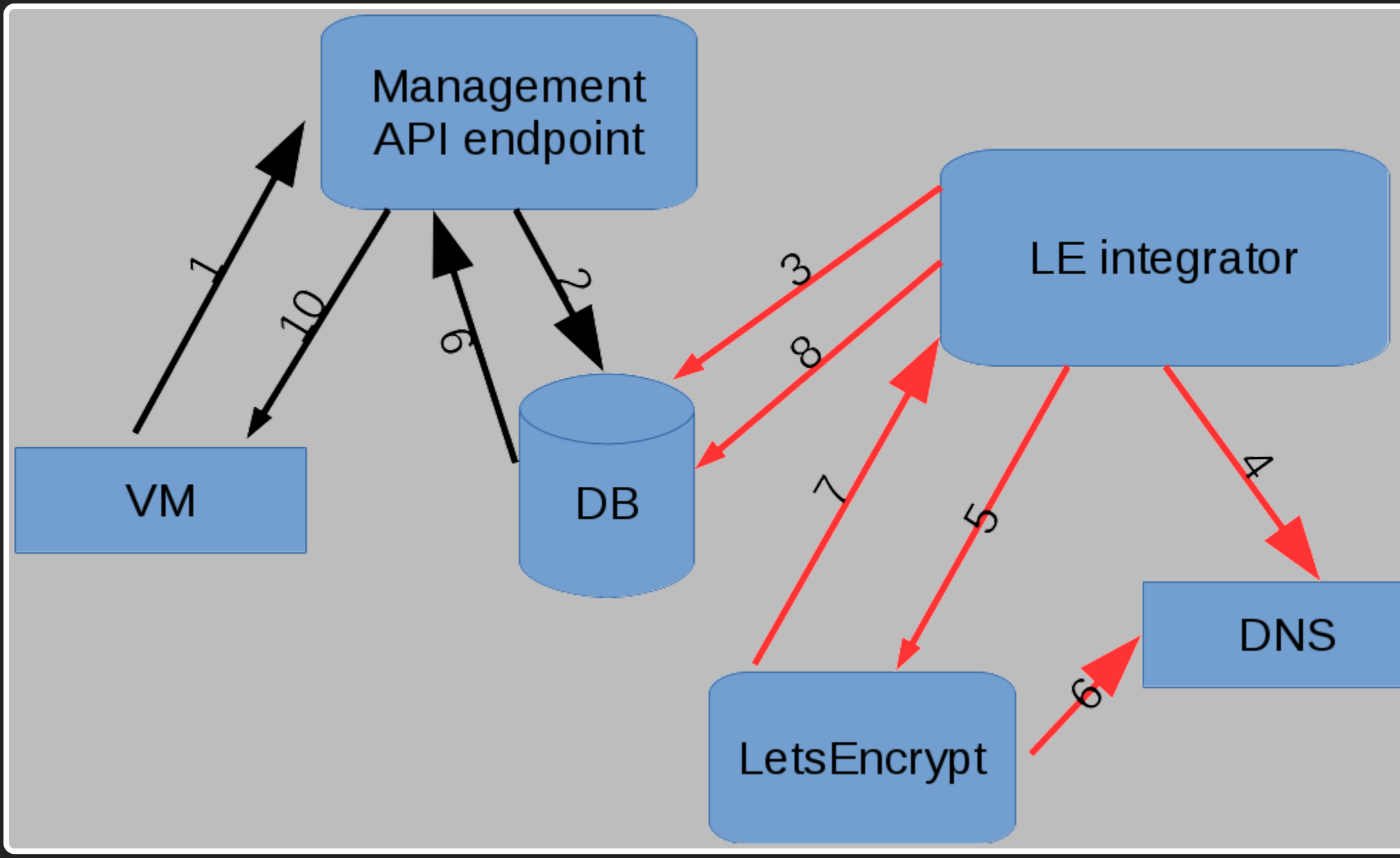
ACME in pginfra

Back on the VM

```
borka pginfra: Downloading certificate 5-borka.postgresql.org
borka pginfra: Replaced file /etc/lighttpd/certfiles/5-borka.postg
borka pginfra: Replaced file /etc/lighttpd/certfiles/5-borka.postg
borka pginfra: Replaced file /etc/lighttpd/conf-available/_pginfra
borka pginfra: Completed user and package checks.
borka pginfra: Restarting service lighttpd
```

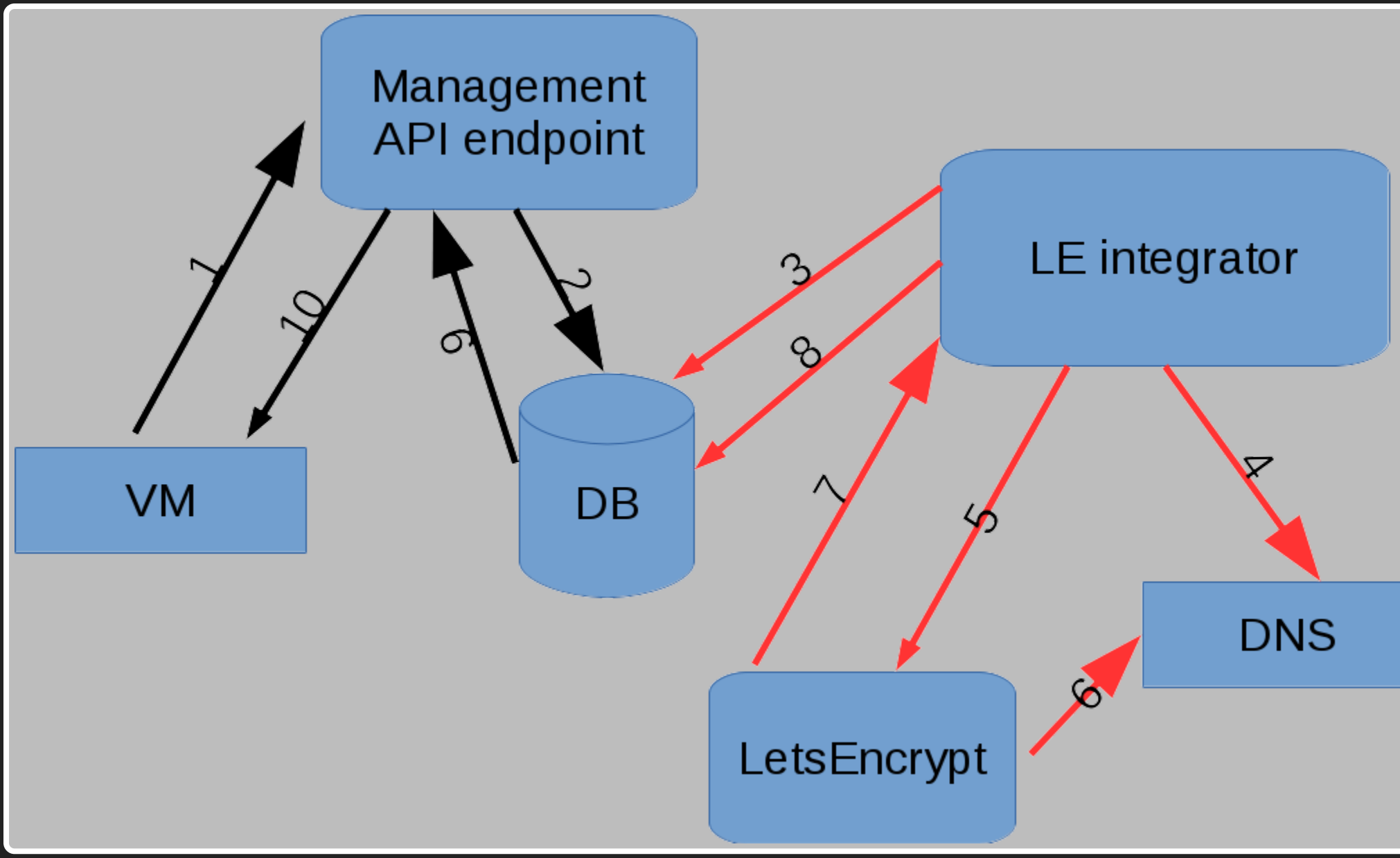
ACME in pginfra

Keys stay on VM


Management
API endpoint
LE integrator
VM
DB
LetsEncrypt
DNS
1
2
3
4
5
6
7
8
9
10

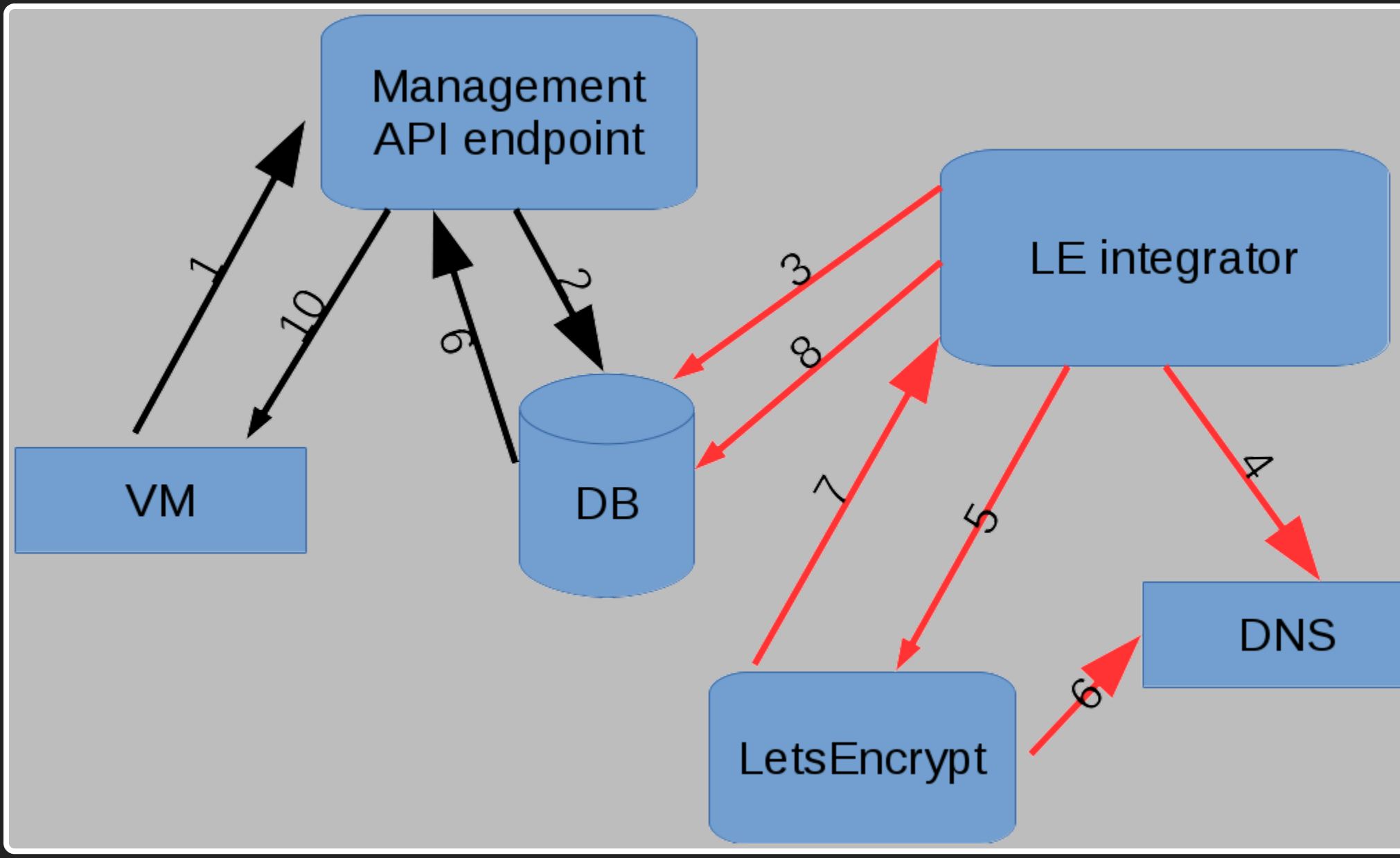
ACME in pginfra

Services never exposed


Management
API endpoint
LE integrator
VM
DB
DNS
LetsEncrypt
1
2
3
4
5
6
7
8
9
10

ACME in pginfra

Audit trail and certificates archived


Management
API endpoint
LE integrator
VM
DB
LetsEncrypt
DNS
1
2
3
4
5
6
7
8
9
10

What does it look like?

- Simple code
- acme python module
  - DNS support not released yet
  - Using git head version
  - Same as certbot...
- OpenSSL
  - ...

Generating CSR

```
def sync_public_certificates():
   ...
   for c in certdata:
      if c['csrneeded']:
         key = crypto.PKey()
         key.generate_key(crypto.TYPE_RSA, 4096)
         req = crypto.X509Req()
         req.get_subject().CN = hostname
         if c['secondary']:
            req.add_extensions([crypto.X509Extension(b'subjectAltN
                  value=", ".join("DNS:%s" % d for d in c['seconda
         req.set_version(2)
         req.set_pubkey(key)
         req.sign(key, "sha256")
         csrdata[c['name']] = crypto.dump_certificate_request(
               crypto.FILETYPE_PEM, req)
```

Central integration

```
def main():
   dns = LetsencryptDnsManager()
   curs.execute("""SELECT c.id, primaryname, secondarynames, csr
FROM letsencrypt_certificate c
LEFT JOIN letsencrypt_issuedcertificate ic
ON ic.basecert_id=c.id WHERE csr != ''
GROUP BY c.id HAVING max(issuedat) < now()-'60 days'::interval
OR max(issuedat) IS NULL""")
   leissuers = [LetsencryptIssuer(*r) for r in curs.fetchall()]

   if len(leissuers) == 0: sys.exit(0)

   leclient = LetsencryptClient()
```

Central integration

```
# Get all possible identifiers (the same one might be used more th
identifiers = set(chain.from_iterable([i.get_all_identifiers() for

leclient.get_challenges(identifiers)
remaining = leclient.remaining_challenges()
if remaining:
  for challenge in remaining:
    dns.add_challenge_record(challenge.get_dns_name(), challenge.g

  # Update zone serials and commit
  dns.flush_challenges()

  while True:
    n = dns.check_records()
    if n == 0: break
    time.sleep(30)
```

Central integration

```
# Trigger letsencrypt to check
for challenge in remaining:
  challenge.answer_challenge()

# Wait for all challenges to be confirmed
while True:
  remaining = leclient.remaining_challenges(True)
  if not remaining: break
  time.sleep(30)

for i in leissuers:
  (pemcert, pemchain, expires) = i.issue(leclient)
  curs.execute("INSERT INTO letsencrypt_issuedcertificate ...."

dns.cleanup()
```

Certificate deployment

- Certificates downloaded on next sync
- Written to standard Debian directories
  - /etc/ssl/certs
  - /etc/ssl/private
- List remembered for plugins

Certificate deployment

- Depends on webserver
- Already have plugin setups
- Note order of certs, keys and chains!
- Don't forget to restart!

Certificate deployment

```
for c in get_public_certificates():
  cf = StringIO()
  cf.write(read_file('/etc/ssl/certs/pginfra_public_{0}.crt'.forma
  cf.write(read_file('/etc/ssl/private/pginfra_public_{0}.key'.for
  cf.write(read_file('/etc/ssl/certs/pginfra_public_{0}.chain'.for
  cf.write(read_file('/etc/haproxy/dhparams.pem'))

  replace_file_from_string('/etc/haproxy/certfiles/{0}.combined'.f
      cf.getvalue(),
      'haproxy',
      0600)
```

Certificate renewal

- Same as reissue
- No special handling
- Separate rate limit

Rate limits

- Letsencrypt has rate limits
  - 20 new certs / domain / week
  - 100 names / cert
  - 5 duplicate certs / week
  - 500 registrations / ip / 3 hours
  - 300 pending authorization
- We're nowhere near these limits

Conclusions

- Much easier than before
  - Close to 0 work deployment
  - 0 work maintenance and renewal
- Better security
  - No shared keys

Conclusions

- Direct work with ACME is easy!
- Don't forget to monitor expiry!!

Thank you!

Magnus Hagander
magnus@hagander.net
@magnushagander
http://www.hagander.net/talks/