Security for the **WoT**

Niels Olof Bouvin

Overview

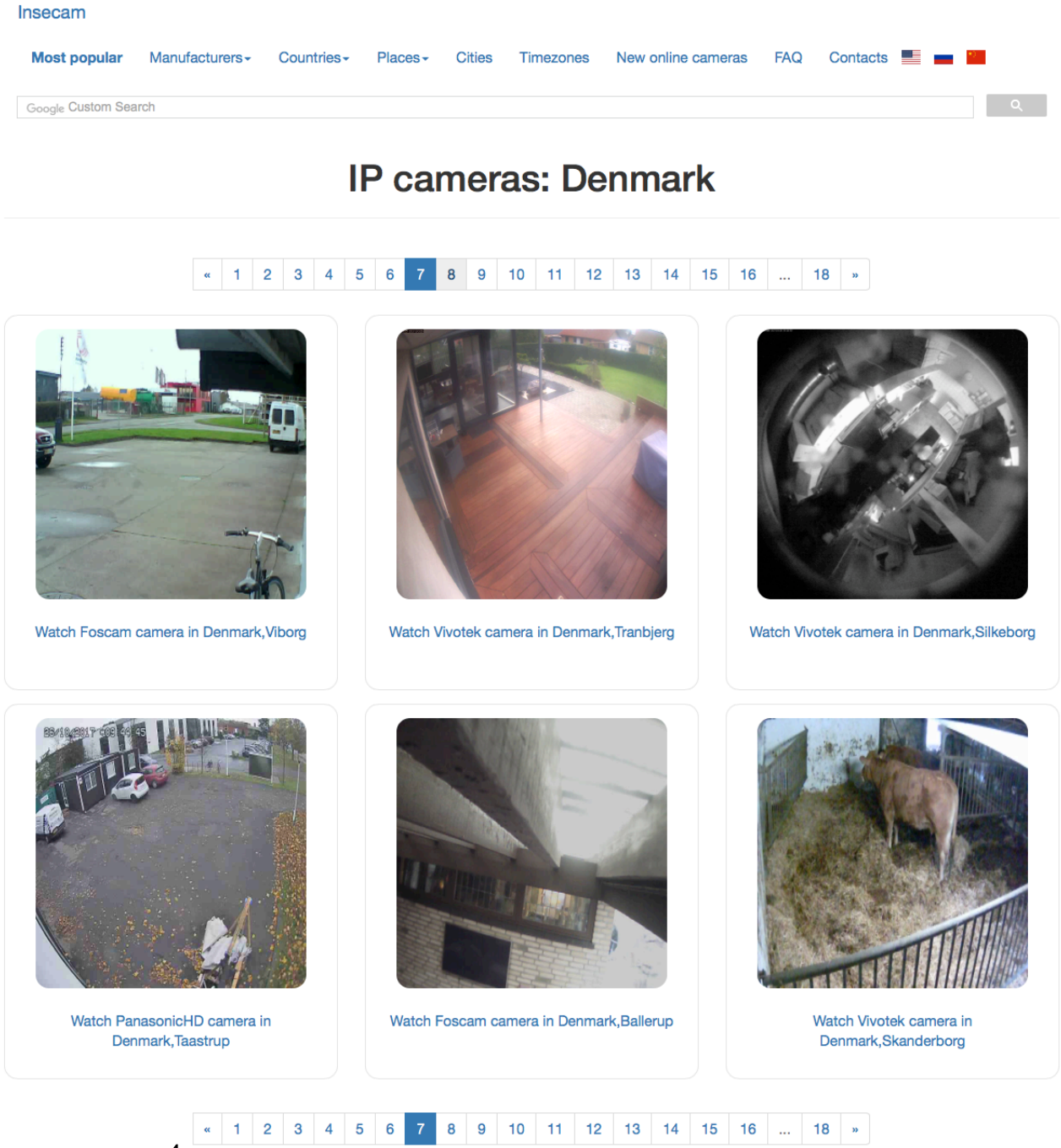
- The state of IoT security
- Securing and sharing the Web of Things

The Internet of Things

- **Myriad interconnected devices, reporting and controlling**
  - many different suppliers
  - many different architectures and systems
  - many different use situations ranging from trivial to absolutely crucial
  - many different actors and agendas
- **What could *possibly* go wrong?**
- **Very early days, yet things are not well**

http://www.insecam.org/en/bycountry/DK/

- Maybe, just *maybe*, you should set a password?



Types of attacks

- **Denial of Service**
  - removing ability to use a device or a service
  - annoying, if I can't use my IoT toaster; catastrophic, if national power grid is down
- **Surveillance**
  - by the state, by commercial interests, by criminals
- **Intrusion**
  - a root kit on a smart device inside an installation could potentially compromise all devices within network reach

Fear the IoT_Reaper

- **A new Botnet is attacking surveillance cameras, home routers, NAS boxes, etc, *right now***
  - devices from D-Link, TP-Link, Avtech, Netgear, MikroTik, Linksys, Synology, and others
- **Current estimates put the number of infected machines around 1-2 mill.**
- **It is still growing and it has not yet been used for anything**

How did we get here?

- **Many IoT devices have *very* poor security**
  - unencrypted traffic
  - firmware not being patched—either by the manufactor or the owner
  - best security practices not followed
- **They are in homes and companies—*inside* the firewall**
  - and can thus act as trojan horses or vectors for attacks
  - as well as surveillance and industrial espionage
  - (this is why *all* communication inside and outside your network should be encrypted)
- **This is not bad. This is really, *really* bad**

Network level security

- **Challenge: Heterogeneity**
  - strong cryptography may be straightforward to implement on ordinary computers, but what about much more constrained devices?
  - public key infrastructure may be difficult to handle in a large IoT setting
  - gateways can handle part of the burden
- **Centralistic solution simplest, but also a single of point of failure**
  - competent actors will act responsible, but that still leaves the rest…

Privacy

- **A user's data should belong to the user**
  - unless this can be ensured, the IoT can become the perfect surveillance infrastructure
- **Centralistic solutions easier to exploit**
  - how can the user ensure proper treatment of collected data?
- **Distributed solutions keep data closer to the user (and their control)**
  - but leaves more points to attack

Identity

- **IoT objects must have identities that can be found and authenticated by other services**
  - identities can be fixed (5794-118), or fluid (lecture hall for P2P/IoT course)
  - identities can be revealed or hidden (behind authenticated third parties)
- **Typical much easier to implement in a centralised system—many challenges remain in a distributed system with ad-hoc connections**

Trust

- **How can trust be built?**
- **One thing is the negotiation between devices**
  - based on authentication, negotiation, and observation
- **Another is the trust of users in the IoT**
  - transparency
  - control

Fault tolerance

- **Things will go wrong**
- **The system must cope**
  - identify errors and failing sensors
  - choose alternative sensors or services
- **Sometimes systems will come under attack**
  - identify compromised systems
  - route around damage

Summary

- Security and privacy are major requirements for a successful IoT
- So far, there have been plenty of examples of early IoT systems susceptible to attacks
- Clearly, this will have to change
- Industry standards and/or government regulations
- Who owns the data?
  - the generator of the data?
  - the provider of the service?

Overview

- The state of IoT security
- Securing and sharing the Web of Things

The elements of a secure Thing

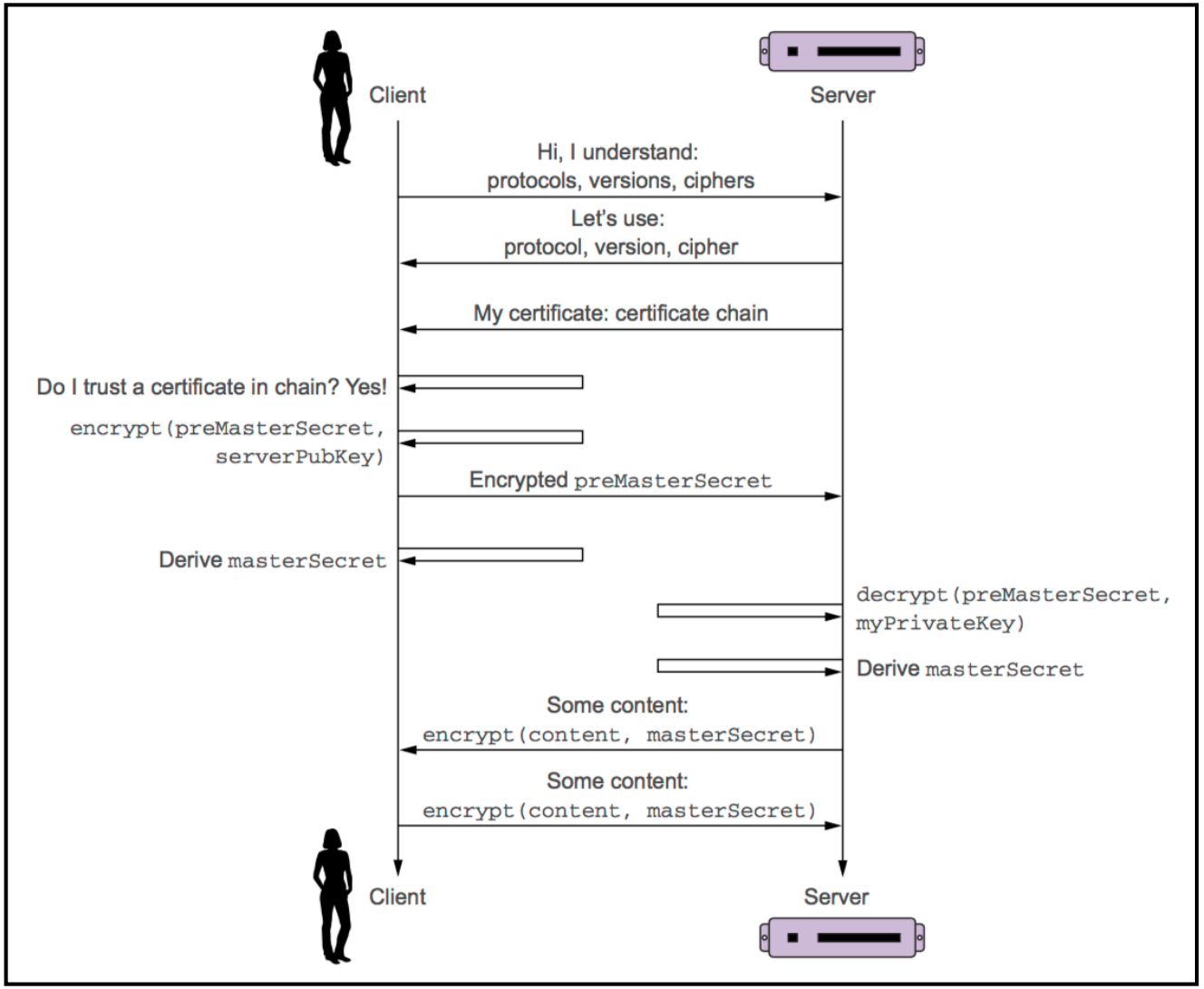
- Encrypted communications
- Authenticated servers
- Authenticated clients
- Secure access control
- Secure software updates

Encrypted communications

- The basic requirement
- Provided that keys are exchanged securely, this should ensure no eavesdropping
- Asymmetric and symmetric encryption combined

Authenticated server

- We need to know the Thing is the *actual* Thing
  - HTTPS and TLS to the rescue
- Keys can be generated locally, and HTTPS support is in Node.js
- https://letsencrypt.org/ provides free certificates



Authenticated client/user

- **Some IoT devices may not need user authentication**
  - weather stations, public sensors, . . .
- **Others certainly require it**
  - cameras, anything with an actuator, anything privacy sensitive

Letting users in

- **Simplest approach: create user profiles with**
  - user names
  - passwords
  - privileges
  - etc
- **Require authentication over secure connection before access is granted**
- **Access, once granted, handled through a token**
  - generated by the server
  - exchanged in headers between client and server

OAuth

- Rather than having users remember yet another password (and having to store that securely), let users connect using preexisting identities
- The user authenticates themselves to a known service, and that service then authorises access to their API for that user from your server
- Not a perfect system—users have been fooled by phishing attacks with sites purporting to be, e.g., Google Docs requesting authorisation
  - though this is no different than ordinary phishing attacks

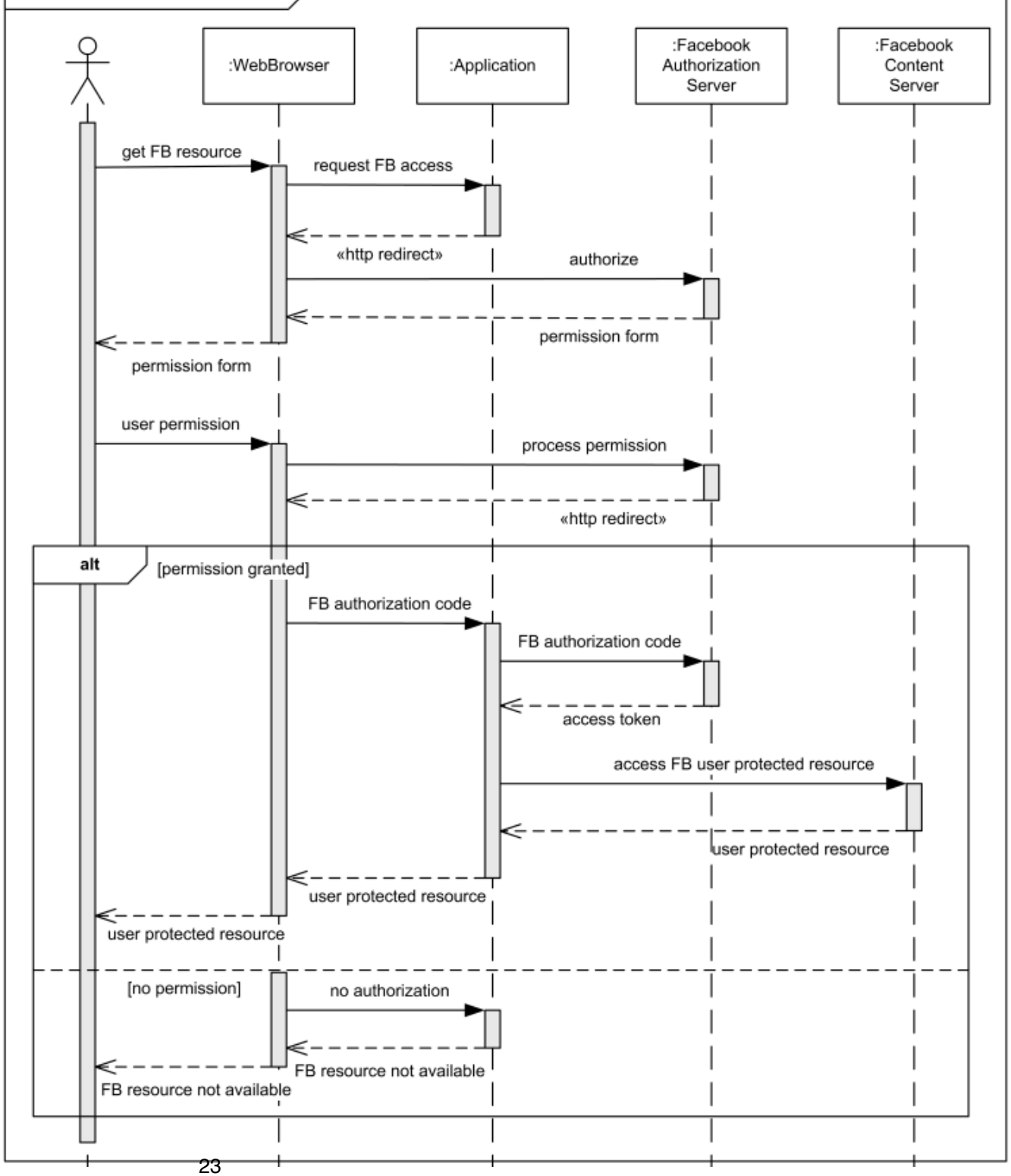
Roles in OAuth

- **The application**
  - the application needs access to some part of the user's account for, e.g., identity
- **The resource server**
  - provides the API for accessing the user's account
- **The authorisation server**
  - handles the interaction with the user granting/denying access (i.e., login to Twitter)
- **The resource owner**
  - the user, who is granting/denying access to part of their account at the resource server

Requisites

- **The application must be registered with the authorisation server, which provides**
  - client id (this can be public information)
  - client secret (this cannot)
- **The application must provide an redirection URL**
  - which must be secure, e.g., HTTPS

- The interaction between a user, an application, and Facebook authorisation & content servers
- Tokens exchanged through redirects
- Given the access token, the application can access the Facebook API

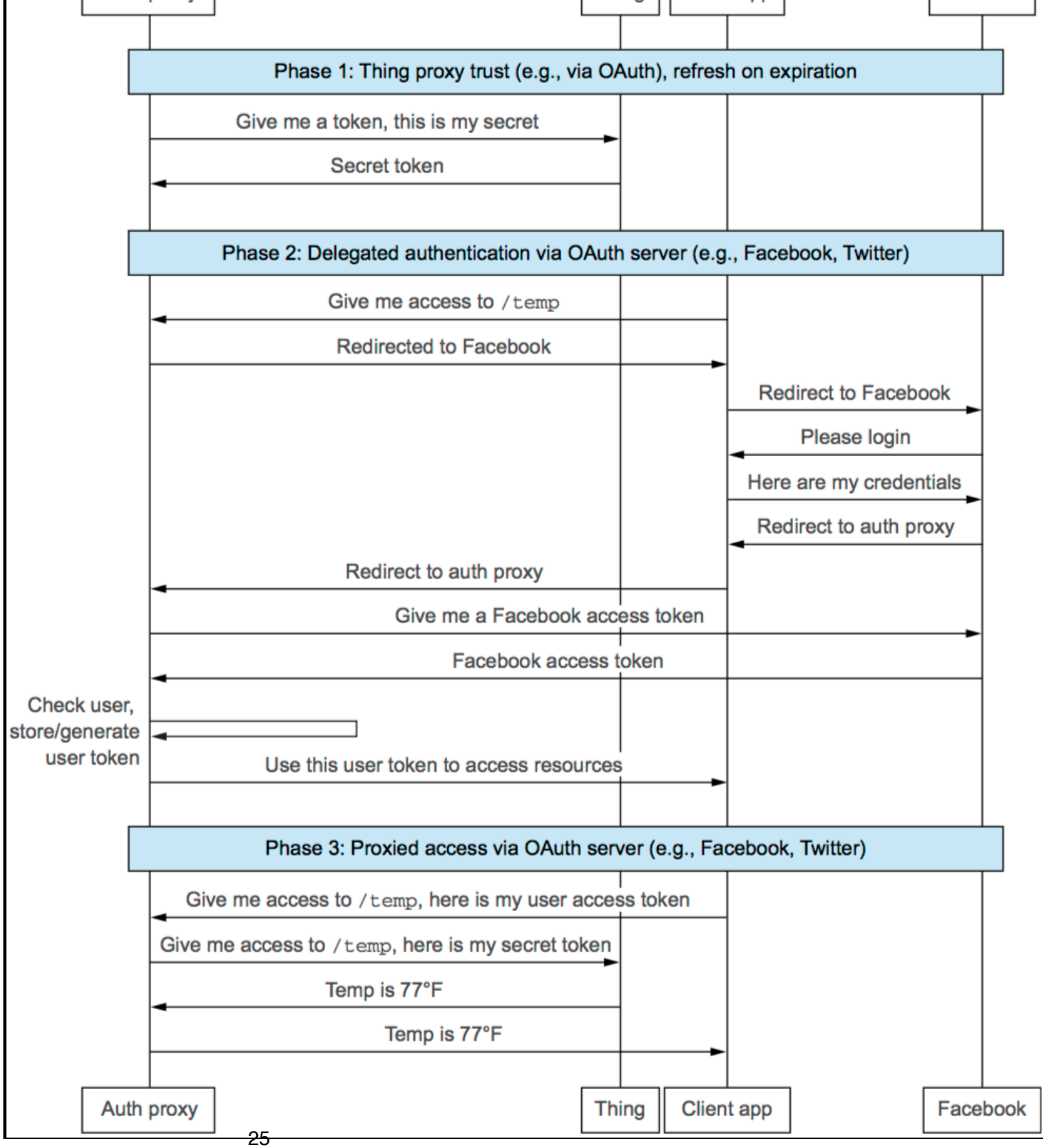


The Social Web of Things

- **Using OAuth helps with authorisation and identities of users**
- **Having each and every Thing maintain lists of users is a bit cumbersome though…**
- **An Authentication Proxy could handle this interaction**
  - registering all the Things
  - handling access to the authorisation servers through OAuth
- **One Authentication Proxy for, e.g., a building, a company, or a home**

SWoT flow

- Using the token generated, the user can then access the protected resources
  - in *Fahrenheit?*



User identity is not enough

- User roles must also be defined, also known as Access Control Lists
- Some users can be administrators, others cannot or should not

So… what if there is a bug?

- **Software contains bugs, and so will IoT devices**
- **Providing a secure mechanism to push software updates out to devices becomes crucial**
  - if the software is not updated, security holes may not be patched, and new features cannot be added
  - if the update mechanism is compromised, the device can be loaded with malware by criminals
  - one solution is an App Store with automatic downloads
- **resin.io provides such a service**
  - based on git, node.js and/or Docker
  - and it's free, if you have no more than 10 devices

Security is tricky

- **Security is not an end goal—it is a process**
  - all software contains bugs, including security software, so we must adapt over time
  - the web stack is the most used in the world, so its security will receive much scrutiny
- **Many IoT devices provide terrible security, which puts only the devices and their users at risk, but the entire Internet, if these devices are weaponised into botnets**
- **Therefore: do better, be careful, follow best practices**