ActiveMQ and RadioVIS

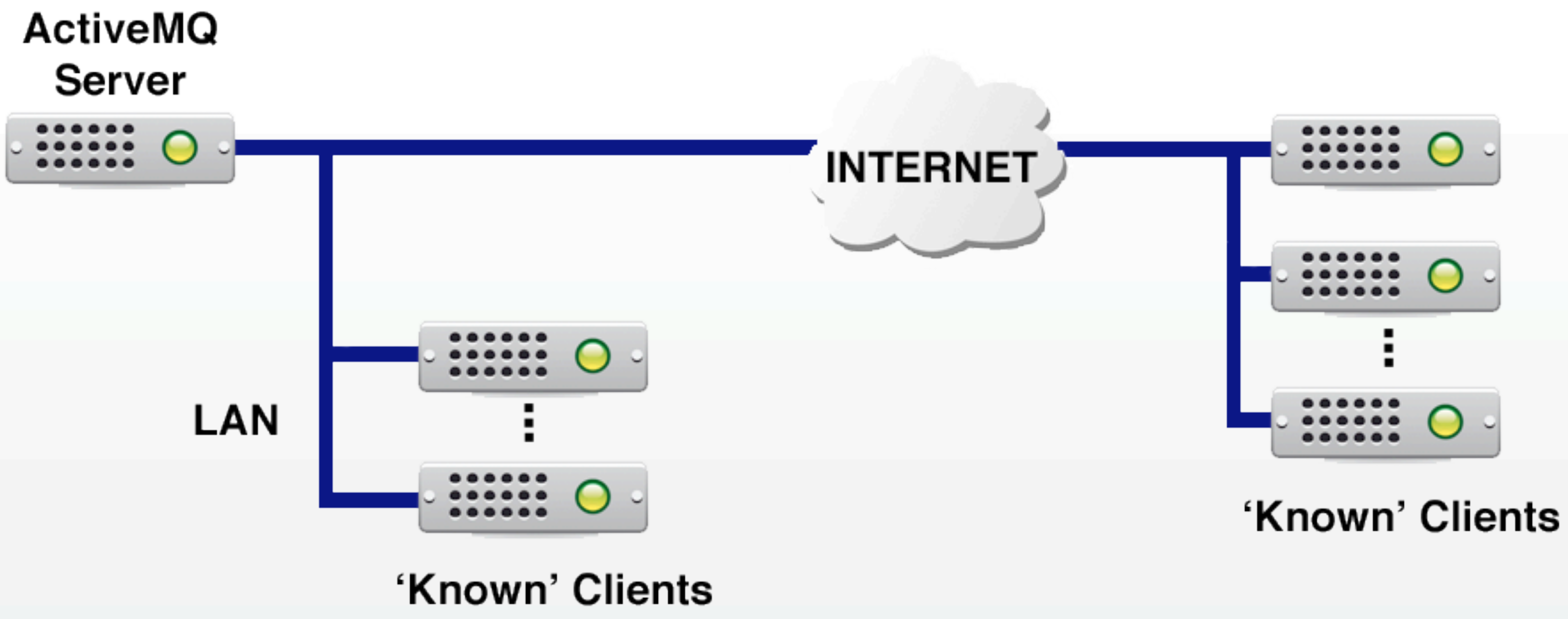
Some of the useful things I have learnt from two years of fighting to keep a server running

DISCLAIMER

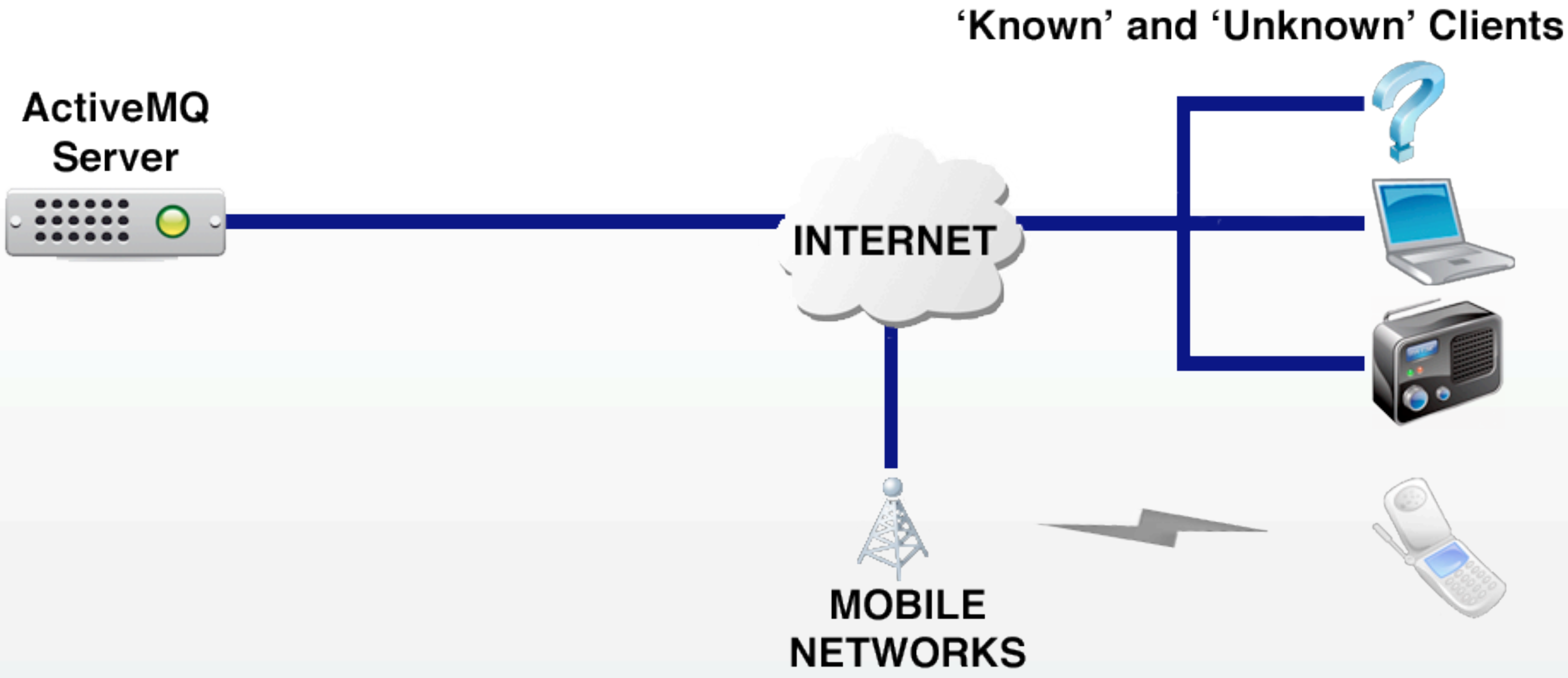
- This could all be completely wrong

Why the default configuration is not ideal for RadioVIS

Common Use Case



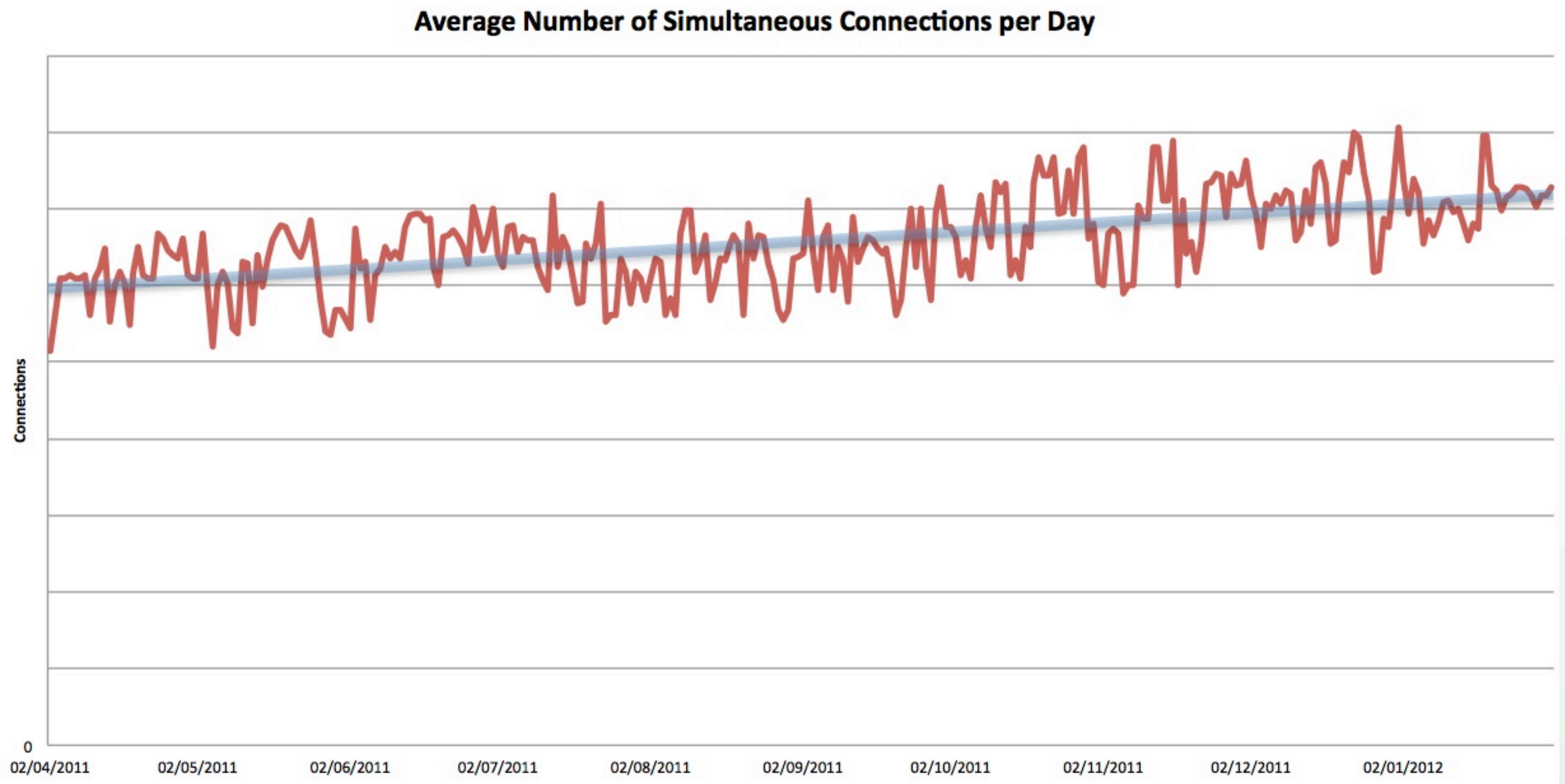
The RadioVIS Environment



Our Setup

- A single server running ActiveMQ
- One dual-core processor with 2 GB RAM
- Approximately 860 topics
- Large number of simultaneous connections
- UK Radio Player clients fed from a different server

Traffic Levels



Areas Affecting Performance

- Environmental
- Configuration
- Software Bugs

Environmental

- Operating System
- Virtual Machine

Operating System

- Other applications running on the same server
- Timing of CRON jobs
- Maximum Open File Descriptors

JAVA Virtual Machine

- Maximum Java Heap Size
- SurvivorRatio
- Use Parallel GC
- Disable Explicit GC
- MaxPermSize
- Stack Size

Active MQ Configuration

- Persistence
- Memory Limits and System Usage
- Producer Flow Control
- Message Eviction Strategy
- Pending Message Limit Strategy
- Pending Subscriber Strategy and VM Cursor
- Slow Consumer Strategy

Persistence

- Persistence is used to allow undelivered messages to be recovered in the event of a system failure.
- When persistence is enabled on the broker, messages are written to a database.
- In a RadioVIS environment, we do not care about persistence. We send messages every 10 seconds or so, so if we lose one it really isn't an issue.
- With 860 topics we save a significant number of database writes.

```
<broker  ...  persistent="false" ... >
```

Memory Limits and System Usage

- Memory limits are used to prevent the broker from running out of resources.
- In ActiveMQ there are 3 areas where messages are stored by the broker: in the broker's memory, in persistence store and in temp storage.

```
<systemUsage>
    <systemUsage sendFailIfNoSpace="true" >
        <memoryUsage>
            <memoryUsage limit="384mb" percentUsageMinDelta="10"/>
        </memoryUsage>
        <tempUsage>
            <tempUsage limit="256mb" percentUsageMinDelta="10"/>
        </tempUsage>
    </systemUsage>
</systemUsage>
```

Memory Limits and System Usage

- Can set memory limit per topic, but appears to have very little effect

```
2012-02-07 09:09:27,962 | DEBUG | default:memory:topic://ip/
http%3A%2F%2Fvis.media-ice.musicradio.com%2FChoiceFM/
image:memory: usage change from: 486% of available memory, to:
487% of available memory | org.apache.activemq.usage.Usage |
ActiveMQ Transport: tcp:///10.15.81.142:41574
```

Producer Flow Control

- Flow control occurs when the broker detects that the memory limit for the topic or the temp store limit for the broker has been exceeded.
- The default settings will cause the producer to block when the memory limits are reached.
- Producers that use Async Sends do not wait for any acknowledgement from the broker; so, when a memory limit has been exceeded, the producer will not be notified.

```
<policyEntry topic=">" producerFlowControl="false" ...>
```

Message Eviction

- The MessageEvictionStrategy is used to decide which message should be evicted on a slow consumer.

```
<messageEvictionStrategy>
   <oldestMessageEvictionStrategy/>
</messageEvictionStrategy>
```

Pending Message Limit

- The strategy calculates the maximum number of pending messages to be kept in RAM for a consumer (above its prefetch size).
- A value of zero means keep no messages around other than the prefetch amount.
- A value greater than zero will keep up to that amount of messages around, discarding the older messages as new messages come in. A value of -1 disables the discarding of messages.
- Setting prefetch limit only appears to be possible for clients connecting via JMS

```
<pendingMessageLimitStrategy>
    <constantPendingMessageLimitStrategy limit="1"/>
</pendingMessageLimitStrategy>
```

Pending Subscriber

- The VM Cursors holds references to messages in memory, and passed to the dispatch queue when needed.
- The alternative is to use a file based cursor.
- The VM Cursor is very fast, but also has the downside of not being able to handle very slow consumers or consumers that have been inactive for a long time.

```
<pendingSubscriberPolicy>
    <vmCursor />
</pendingSubscriberPolicy>
```

Slow Consumers

- Slowness is a product of the prefetch and message production rate.
- Abort slow consumers when they reach the configured threshold of slowness.
- The default setting aborts clients that are slow for 30 seconds

```
<slowConsumerStrategy>
    <abortSlowConsumerStrategy/>
</slowConsumerStrategy>
```

Bugs

With JMX enabled, sockets may become stuck in the CLOSE_WAIT state.

Useful Resources

- Log files
- JStack
- Source
- Forums