Misusing Open Services on the Internet

Jelte Fennema    Ben de Graaff

University of Amsterdam
Supervisor: Rick van Galen (KPMG)

February 3, 2016
Introduction

Open service: no authentication or default credentials

Relevant: more than **35,000** open MongoDB instances
Exposing **685 TB** (last December [1])

More than just data leaks – example: botnet command and control
A problem for devops and software developers

“Memcached does not spend much, if any, effort in ensuring its defensibility from random Internet connections. So you must not expose Memcached directly to the Internet.”

— Memcached documentation

“Everybody has privileges to do anything. Neat.”

— CouchDB security documentation
Research goals

- What are settings that lead to exploitable services?
- What are the operations required when exploiting an open service as a command & control server?
- What are best practices for default configurations and authentication?
Approach

For various software packages...

- Examine configuration (weaknesses?)
- Tool to scan level of access
- Proof of concept: botnet command & control
**Approach**

For various software packages...

- Examine configuration (weaknesses?)
- Tool to scan level of access
- Proof of concept: botnet command & control

**Scanning the Internet**

- Shodan
- ZMap and our own scan tool
Software classes

- **Relational databases**: MySQL, MariaDB, PostgreSQL
Software classes

- **Relational databases**: MySQL, MariaDB, PostgreSQL
- **NoSQL databases**: MongoDB, CouchDB
Software classes

- **Relational databases:** MySQL, MariaDB, PostgreSQL
- **NoSQL databases:** MongoDB, CouchDB
- **Key-value store:** Redis, Memcached
Software classes

- **Relational databases**: MySQL, MariaDB, PostgreSQL
- **NoSQL databases**: MongoDB, CouchDB
- **Key-value store**: Redis, Memcached
- **Message queue**: RabbitMQ
Software classes

- **Relational databases:** MySQL, MariaDB, PostgreSQL
- **NoSQL databases:** MongoDB, CouchDB
- **Key-value store:** Redis, Memcached
- **Message queue:** RabbitMQ
- **Printing protocols:** CUPS (and IPP printers)
Proof of concept

Simple botnet simulation (communication channel):

- Botnet operator sends signed *commands* to one bot or all bots
- Bots execute commands, write back encrypted *results*

Operator → Setup → Write command → Read command → Read result → Write result
Impact on the Internet

What is the impact on the Internet?

Is configuration security a factor?
**Configuration security**

<table>
<thead>
<tr>
<th></th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

**Table 1**: Comparison of security settings for the software packages
## Configuration security

<table>
<thead>
<tr>
<th>Feature</th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication by default</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>No public creds or anon</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

**Table 1**: Comparison of security settings for the software packages
## Configuration security

<table>
<thead>
<tr>
<th></th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication by default</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>No public creds or anon</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Host-based access control</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

**Table 1:** Comparison of security settings for the software packages
## Configuration security

<table>
<thead>
<tr>
<th>Feature</th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication by default</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>No public creds or anon</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Host-based access control</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication always on</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

**Table 1:** Comparison of security settings for the software packages
## Configuration security

<table>
<thead>
<tr>
<th></th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication by default</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>No public creds or anon</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Host-based access control</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication always on</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Minimal steps to make open</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 1:** Comparison of security settings for the software packages
## Configuration security

<table>
<thead>
<tr>
<th></th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication by default</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>No public creds or anon</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Host-based access control</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication always on</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Minimal steps to make open</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Steps to make public/secure</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**Table 1:** Comparison of security settings for the software packages
Percentage of open services exposed to the Internet

**Figure 1:** Percentages of open services.
## Combined results

<table>
<thead>
<tr>
<th></th>
<th>PostgreSQL</th>
<th>CUPS</th>
<th>RabbitMQ</th>
<th>CouchDB</th>
<th>MongoDB</th>
<th>Redis</th>
<th>Memcached</th>
<th>IPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localhost (in config)</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Not public (default)</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication by default</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>No public creds or anon</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Host-based access control</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Authentication always on</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Minimal steps to make open</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Steps to make public/secure</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Percentage open</td>
<td>2%</td>
<td>21%</td>
<td>22%</td>
<td>72%</td>
<td>71%</td>
<td>37%</td>
<td>98%</td>
<td>81%</td>
</tr>
<tr>
<td>With full access</td>
<td>2%</td>
<td>1%</td>
<td>22%</td>
<td>18%</td>
<td>71%</td>
<td>37%</td>
<td>98%</td>
<td>–</td>
</tr>
</tbody>
</table>

**Table 2:** Combined comparison of software packages
First attempt: Shodan

Shodan:

- Scans the Internet
- Sends command related to that service
- Stores result in plain text

Full text search interface to find open services

Example: `port:5984 couchdb !unauthorized`
### Shodan results

<table>
<thead>
<tr>
<th>Package</th>
<th>Open</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcached</td>
<td>100,044</td>
<td>–</td>
</tr>
<tr>
<td>MongoDB</td>
<td>47,351</td>
<td>–</td>
</tr>
<tr>
<td>Redis</td>
<td>13,455</td>
<td>23,174</td>
</tr>
<tr>
<td>RabbitMQ</td>
<td>6,487</td>
<td>23,121</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>6,391</td>
<td>293,481</td>
</tr>
</tbody>
</table>

**Table 3:** Worldwide statistics based on Shodan
### Shodan inconclusive results

<table>
<thead>
<tr>
<th>Package</th>
<th>Unknown</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL/MariaDB</td>
<td>1,767,930</td>
<td>2,231,132</td>
</tr>
<tr>
<td>Non-CUPS IPP</td>
<td>23,948</td>
<td>1,664</td>
</tr>
<tr>
<td>CouchDB</td>
<td>2,783</td>
<td>513</td>
</tr>
<tr>
<td>CUPS</td>
<td>5,591</td>
<td>29,387</td>
</tr>
</tbody>
</table>

**Table 4:** Inconclusive statistics based on Shodan
Shodan unknowns

- **United States**: 1,581,550
- **China**: 413,865
- **Germany**: 229,203
- **Poland**: 194,094
- **United Kingdom**: 108,840

**TOP ORGANIZATIONS**
- **Hangzhou Alibaba Adv.**: 169,354
- **home.pl webhosting fa...**: 102,771
- **GoDaddy.com, LLC**: 94,186
- **Psycha Networks**: 79,542
- **Unified Layer**: 78,478

**TOP OPERATING SYSTEMS**
- **Linux 3.x**: 90,360
- **Windows XP**: 29,765
- **Linux 2.6.x**: 25,713
- **Windows 7 or R**: 15,503
- **Linux 2.4.2.6**: 577

**TOP PRODUCTS**
- **MySQL**: 3,012,182

**Total results**: 3,681,933

- **198.65.225.18**
  - Host: **www.fivansnorth.com**
  - IP: **5.1.69-log**
  - Details

- **212.223.92.159**
  - Host: **www.evergreen-tolllys.de**
  - IP: **4.0.23-nt**
  - Details

- **107.172.161.16**
  - Host: **ColoCrossing**
  - IP: **\x04Host '\xxx.xxx.xxx.xxx\' is not allowed to connect to this MySQL server**
  - Details

- **174.123.38.59**
  - Host: **ThePlanet.com Internet Services**
  - IP: **\x04Host '\xxx.xxx.xxx.xxx\' is not allowed to connect to this MySQL Server**
  - Details

- **211.177.164.35**
  - Host: **5K Broadband**
  - IP: **5.0.22**
  - Details
Second attempt: Our own scan

Shodan incomplete for some services

Setup

- Permission to scan all Dutch IPs
- Not allowed to log in to any service (required for MySQL, PostgreSQL)
Second attempt: Our own scan

Shodan incomplete for some services

Setup

- Permission to scan all Dutch IPs
- Not allowed to log in to any service (required for MySQL, PostgreSQL)

How

- ZMap to port scan (5.5 minutes for 4.6 million IPs)
- ~20,000 hits per port
- Scanner modified for concurrency (7 minutes at 500 concurrent requests)
Our scan results

<table>
<thead>
<tr>
<th>Service</th>
<th>Open</th>
<th>Closed</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memcached</td>
<td>98%</td>
<td>3,725</td>
<td>70</td>
</tr>
<tr>
<td>IPP</td>
<td>81%</td>
<td>260</td>
<td>61</td>
</tr>
<tr>
<td>CouchDB</td>
<td>72%</td>
<td>190</td>
<td>73</td>
</tr>
<tr>
<td>MongoDB</td>
<td>71%</td>
<td>1,859</td>
<td>753</td>
</tr>
<tr>
<td>CUPS</td>
<td>21%</td>
<td>474</td>
<td>1,824</td>
</tr>
</tbody>
</table>

Table 5: Dutch statistics based on our scan
Type of IP range per service

CUPS/IPP

- Consumer networks
- Some universities
- Businesses

Memcached, MongoDB, CouchDB

- Mostly hosting services
Other quirks we found

**CUPS:**
Print jobs can contain arbitrary attributes

... turning a printing job into a key-value store
Other quirks we found

**CouchDB**: Default security policy empty (writable by anyone)

... including database containing user credentials
Other quirks we found

**Memcached:**
Authentication requires *different*, binary protocol

... not even supported by all clients
Conclusion

- Open services still an issue

- Exploitation is *incredibly easy*: any service you can write data to and read it back later

- Bad (default) configuration can lead to exploitable services
Best practices

- Prefer `localhost` access, `require` authentication for remote access
Best practices

- Prefer **localhost** access, **require** authentication for remote access

- Secure **defaults**!
Best practices

- Prefer **localhost** access, **require** authentication for remote access
- Secure **defaults**!
- Clear **documentation** and warnings (also in configuration)
Best practices

- Prefer **localhost** access, **require** authentication for remote access

- Secure **defaults**!

- Clear **documentation** and warnings (also in configuration)

- **Simplify** configuration of authentication (good configs, tools)
References


*Shodan: the world’s first search engine for Internet-connected devices.* 2009. URL: https://www.shodan.io/ (visited on 04/01/2016).