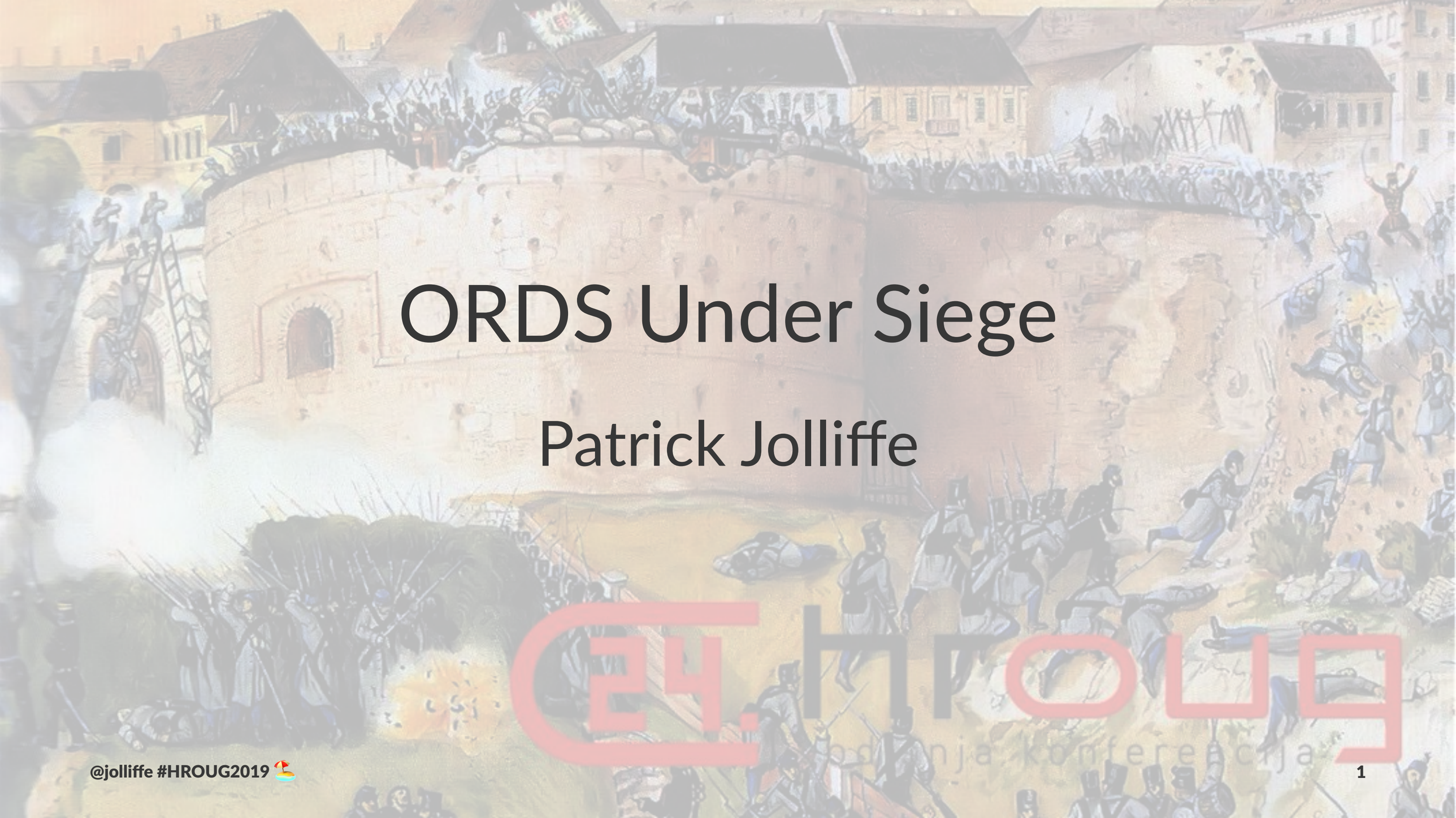


ORDS Under Siege

Patrick Jolliffe

Patrick Jolliffe

Working at Li & Fung as Oracle DBA

Co-Founder of Hong Kong Oracle Usergroup

Oracle ACE

https://jolliffe.hk

Agenda

Introduction

Testing Environment & Benchmarking Tools (Siege)

Tomcat & ORDS under Siege

Reverse Proxies under Siege

Starting Point

- JSP dashboard makes HTTP calls to **ORDS**
- ORDS on Apache Tomcat exposes data from Oracle Database
- **Reverse Proxy** (Apache HTTP Server) in front of Tomcat
- Browser⇔JSP⇔Reverse Proxy⇔ORDS⇔DB

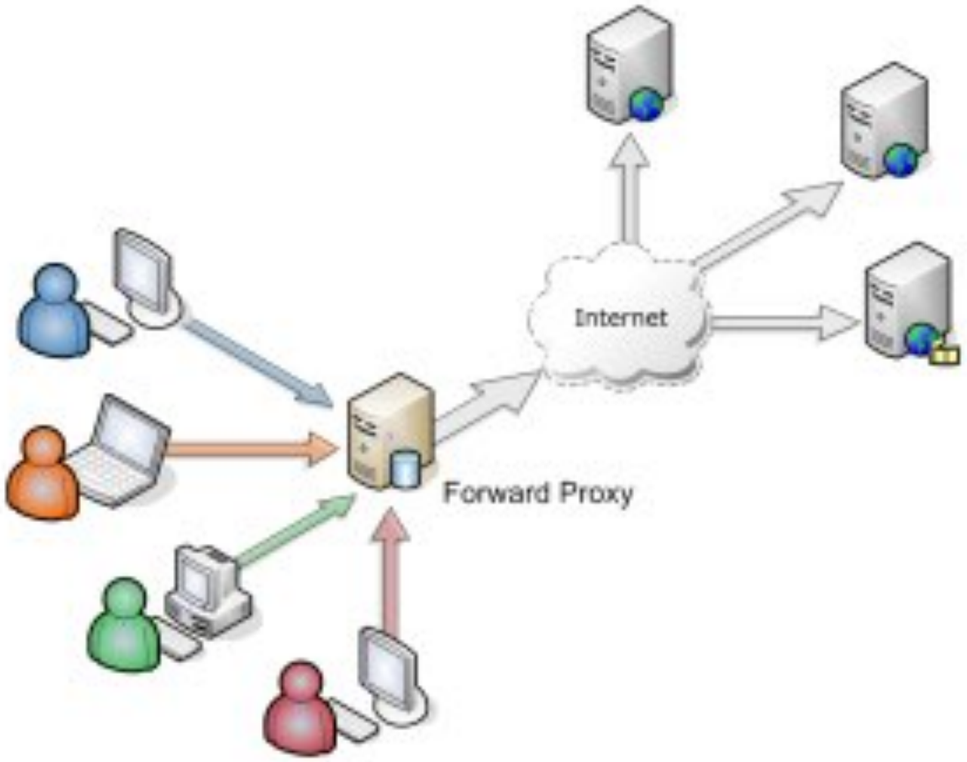
ORDS: Auto REST

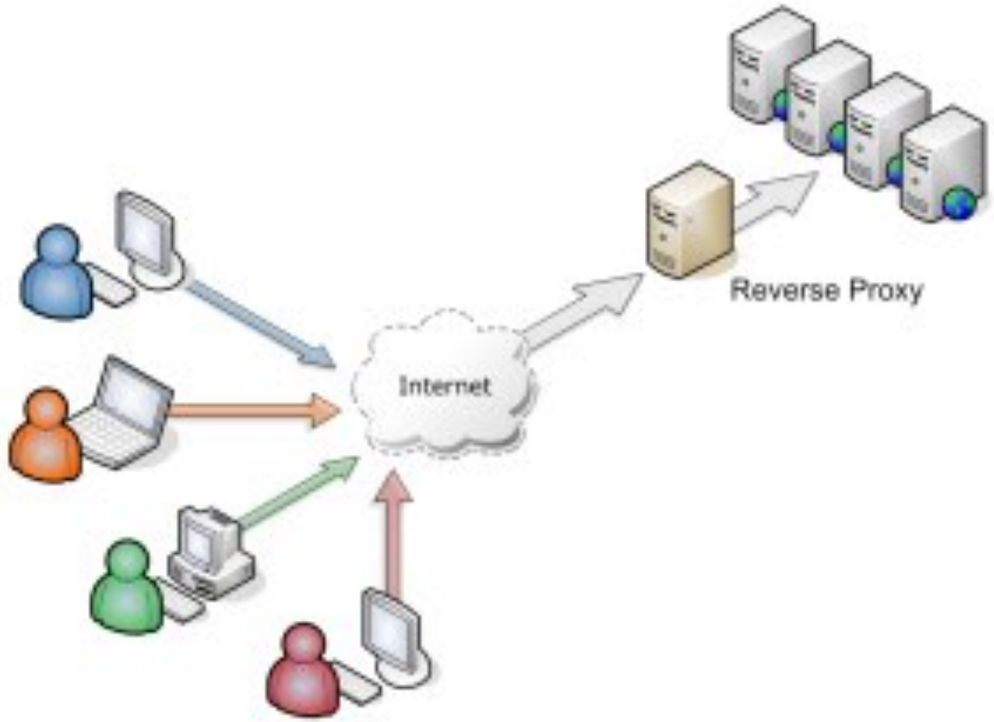
```
BEGIN
   ords.enable_schema(
          p_schema          => 'HR',
          p_auto_rest_auth => FALSE);

   ords.enable_object(
      p_schema          => 'HR',
      p_object          => 'EMPLOYEES',
      p_object_alias    => 'auto_rest_employee',
      p_auto_rest_auth => FALSE);

   COMMIT;
END;
/
```

ORDS: Auto REST

```
[vagrant@ords ~]$ wget http://ords:1110/ords/hr/auto_rest_employee/194    \
>                      -qO- | jq
{
  "employee_id": 194,
  "first_name": "Samuel",
  "last_name": "McCain",
  ...
  "links": [
    {
      "rel": "self",
      "href": "http://ords:1110/ords/hr/auto_rest_employee/194"
    },
    ...
  ]
}
```

HTTP Proxies

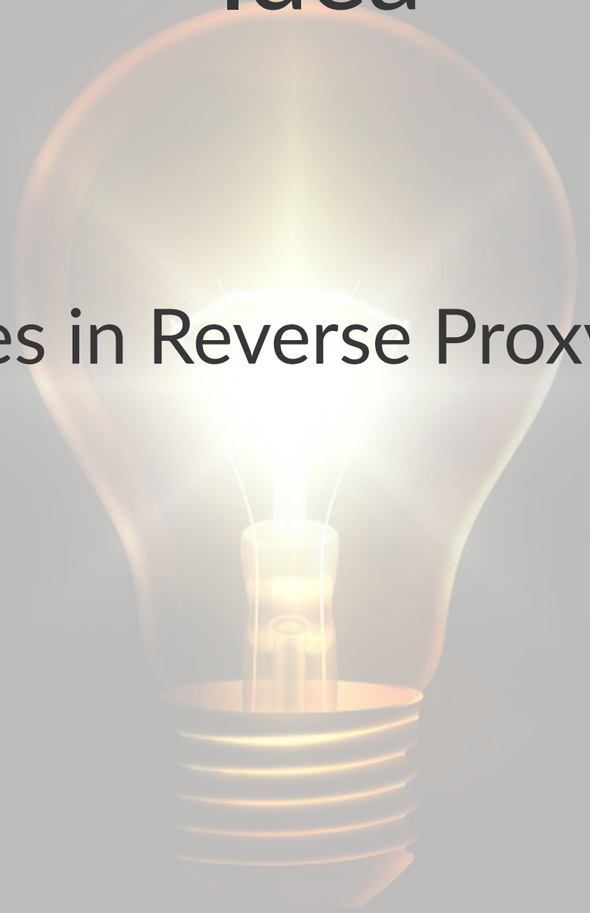




3 Benefits of Reverse Proxies

1. Load balancing
2. Web acceleration
3. Security and anonymity

Idea

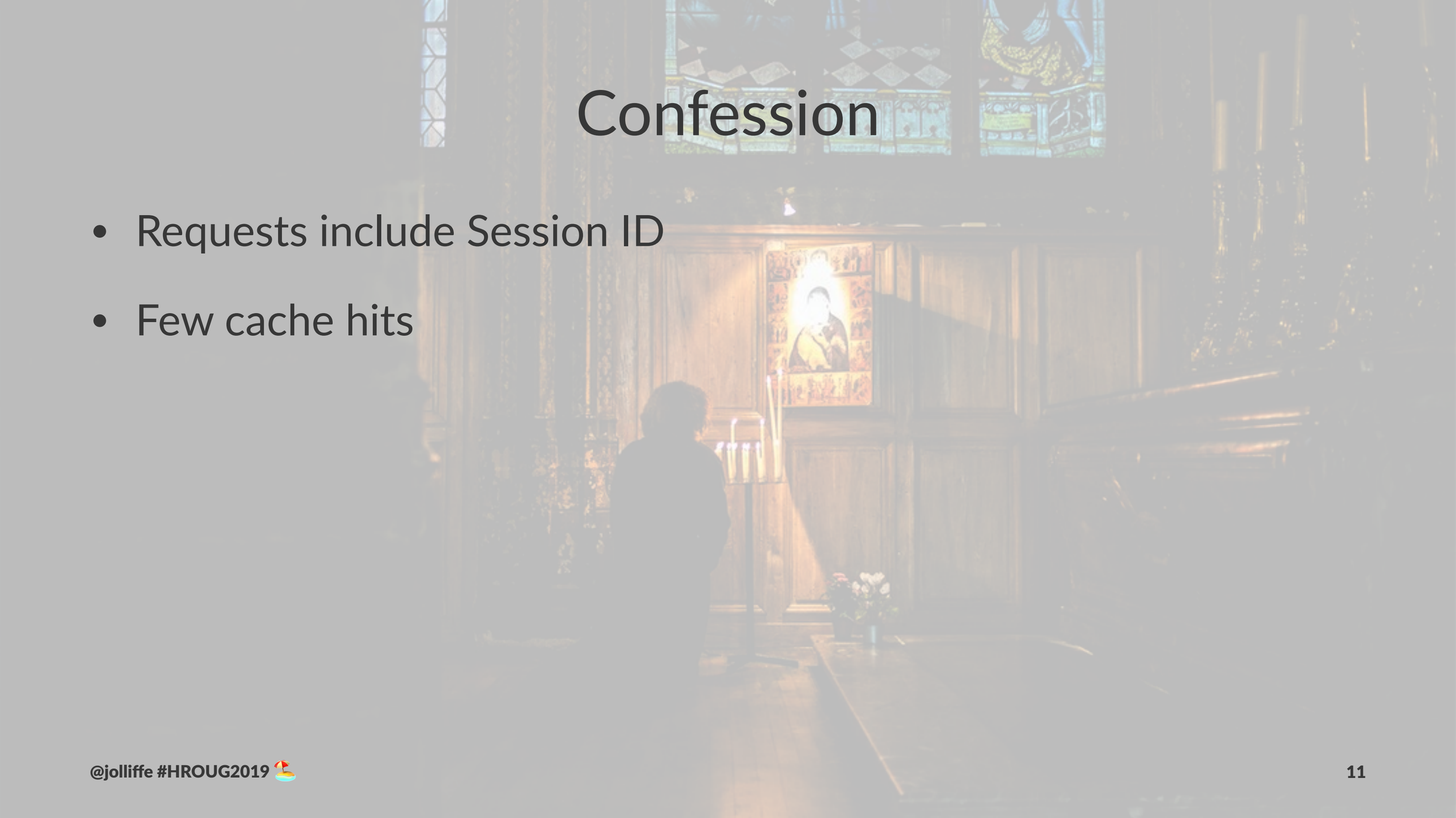


- Nightly data refresh
- Cache HTTP responses in Reverse Proxy

Complication

- Though only reading data, all requests are POSTs
- Apache HTTP Server can only cache GETs

Confession



- Requests include Session ID
- Few cache hits

Test Environment: Software & Versions

- VirtualBox VM with 4 Cores & 6 GB RAM
- Oracle Linux 7.6
- Oracle Database 18c XE (HR sample schema)
- Apache Tomcat 7 (different ports for different configs)
- ORDS 19.2 (query HR.employees by employee_id)

Test Environment: Vagrant Box[1]

```
Patricks-MacBook-Pro:vagrant-ords patrick$ pwd
/Users/patrick/src/vagrant-ords

Patricks-MacBook-Pro:vagrant-ords patrick$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'bento/oracle-7.6'...
...
    default: Complete!

Patricks-MacBook-Pro:vagrant-ords patrick$ vagrant ssh
Welcome to the ORDS Vagrant Machine
ords-demo will run the test suite

[vagrant@ords ~]$ ords-demo
ords-connections DURATION=1M
Demonstrate differences between 1 and 255 connections
** SIEGE 4.0.2
** Preparing 1 concurrent users for battle.
The server is now under siege...
...
```

[1] https://github.com/pjolliffe/vagrant-ords

Siege[2]: Introduction

HTTP load testing & benchmarking utility

Free & Open Source ✓

Lightweight ✓

Simple ✓

[2] https://www.joedog.org/siege-home/

Siege: Configuration: `urls.txt`

```
[vagrant@ords ~]$ cat /etc/siege/urls.txt
http://ords:1110/ords/hr/manual_rest/employee/100
http://ords:1110/ords/hr/manual_rest/employee/101
...
http://ords:1110/ords/hr/manual_rest/employee/198
http://ords:1110/ords/hr/manual_rest/employee/199

[vagrant@ords ~]$ ords-urls.py --reverse_proxy httpd --cache on --method post

[vagrant@ords ~]$ cat /etc/siege/urls.txt
http://ords:2130/ords/hr/manual_rest/employee POST employee_id=100
http://ords:2130/ords/hr/manual_rest/employee POST employee_id=101
...
http://ords:2130/ords/hr/manual_rest/employee POST employee_id=198
http://ords:2130/ords/hr/manual_rest/employee POST employee_id=199
```

Siege: Latency vs Throughput

```
[vagrant@ords ~]$ ords-connections
```

Siege: Latency vs Throughput

```
[vagrant@ords ~]$ siege -c 1 -t 1M
** SIEGE 4.0.2
** Preparing 1 concurrent users for battle.
The server is now under siege...

Lifting the server siege...
Transactions:                    189 hits
Availability:                 100.00 %
Elapsed time:                  59.23 secs
Data transferred:               0.11 MB
Response time:                  0.03 secs
Transaction rate:               3.19 trans/sec
Throughput:                     0.00 MB/sec
Concurrency:                    0.11
Successful transactions:         189
Failed transactions:               0
Longest transaction:            0.06
Shortest transaction:           0.02
```

```
[vagrant@ords ~]$ siege -c 255 -t 1M
** SIEGE 4.0.2
** Preparing 255 concurrent users for battle.
The server is now under siege...

Lifting the server siege...
Transactions:                   7339 hits
Availability:                 100.00 %
Elapsed time:                  59.15 secs
Data transferred:               4.39 MB
Response time:                  1.77 secs
Transaction rate:             124.07 trans/sec
Throughput:                     0.07 MB/sec
Concurrency:                  220.09
Successful transactions:        7339
Failed transactions:               0
Longest transaction:            3.13
Shortest transaction:           0.08
```

Tomcat & ORDS under Siege

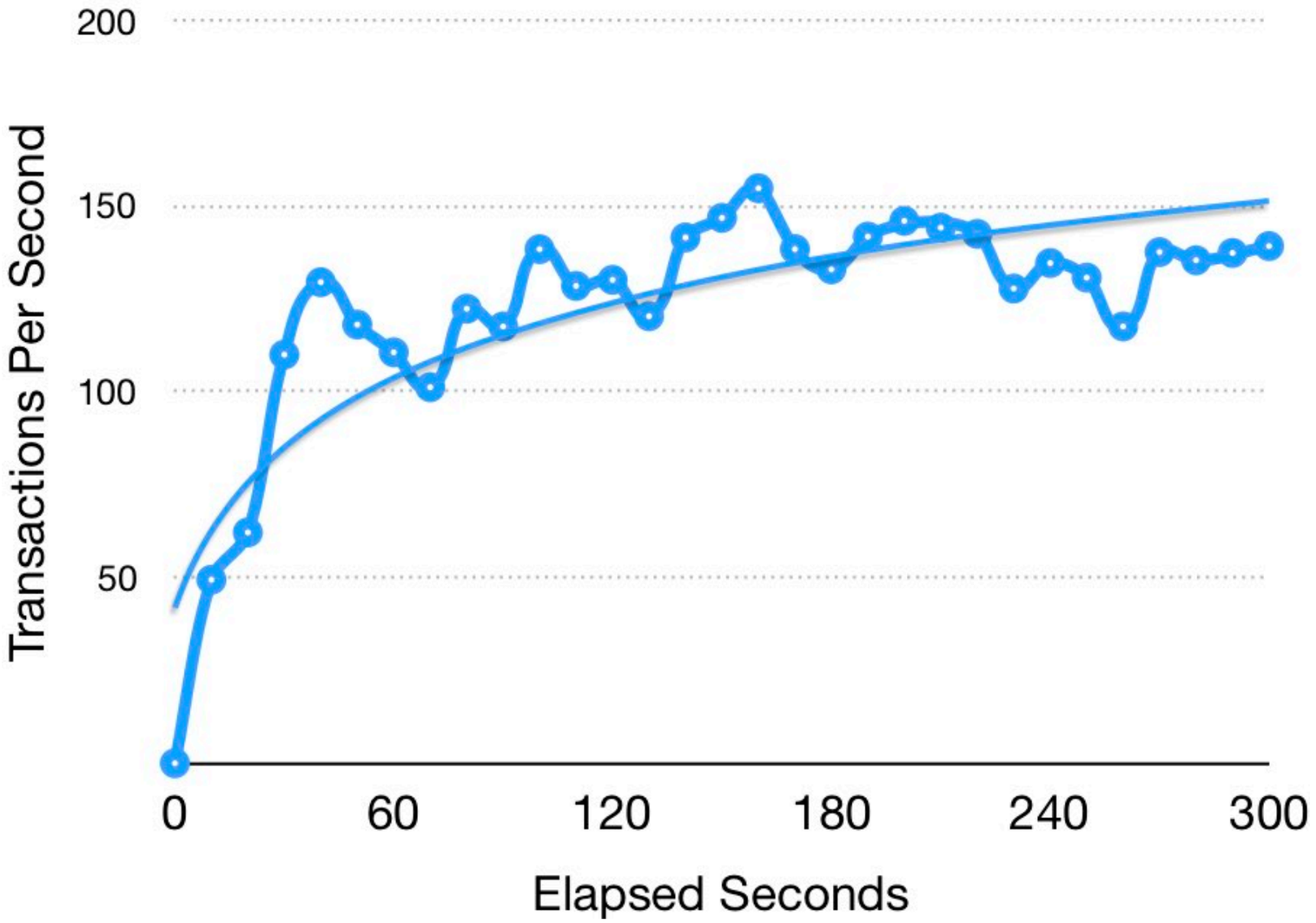
Tomcat: Slow Startup

```
Jul 18, 2019 12:26:44 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 148812 ms

/etc/tomcat/tomcat.conf
JAVA_OPTS="-Djava.security.egd=file:/dev/urandom"

Jul 18, 2019 12:30:47 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 5420 ms
```

Tomcat: Warm Up & Fluctuation

```
[vagrant@ords ~]$ ords-warmup
```

Tomcat: Warm Up & Fluctuation



Tomcat

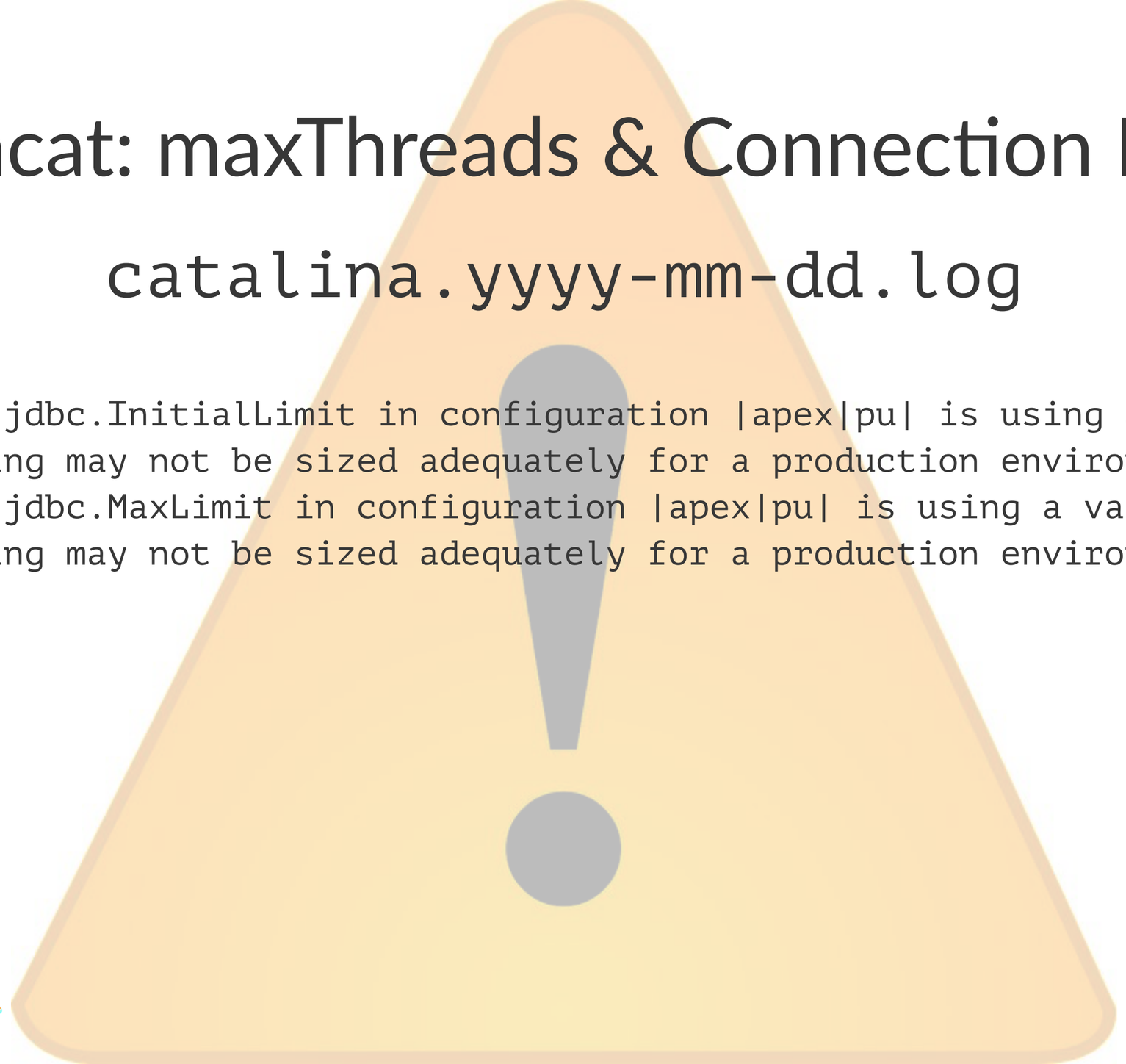
maxThreads & Connection Pool

```
[vagrant@ords ~]$ ords-threads
```

Tomcat: maxThreads & Connection Pool

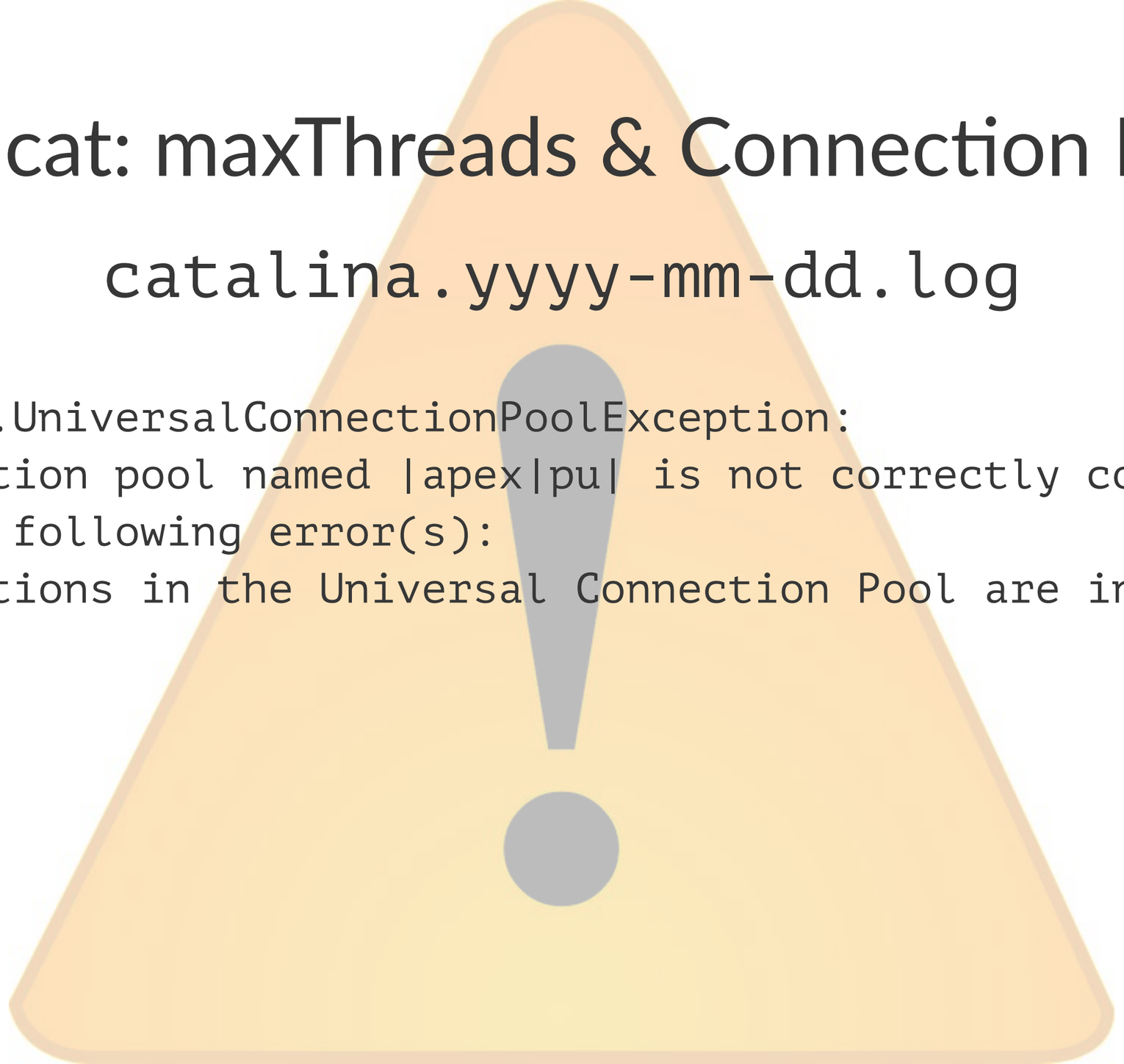
`catalina.yyyy-mm-dd.log`

```
WARNING: *** jdbc.InitialLimit in configuration |apex|pu| is using a value of 3,
   this setting may not be sized adequately for a production environment ***
WARNING: *** jdbc.MaxLimit in configuration |apex|pu| is using a value of 10,
   this setting may not be sized adequately for a production environment ***
```



Tomcat: maxThreads & Connection Pool

`catalina.yyyy-mm-dd.log`



```
oracle.ucp.UniversalConnectionPoolException:
The connection pool named |apex|pu| is not correctly configured,
due to the following error(s):
All connections in the Universal Connection Pool are in use
```

Tomcat: maxThreads & Connection Pool

apex_pu.xml

```
. . .
<entry key="jdbc.InitialLimit> 20 </entry>
<entry key="jdbc.MinLimit">    20 </entry>
<entry key="jdbc.MaxLimit">    20 </entry>
. . .
```

Tomcat: maxThreads & Connection Pool

`catalina.yyyy-mm-dd.log`

```
oracle.ucp.UniversalConnectionPoolException:
The connection pool named: |apex|pu| is not correctly configured,
due to the following error(s):
   All connections in the Universal Connection Pool are in use
```

Tomcat: maxThreads & Connection Pool

apex_pu.xml

```
. . .
<entry key="jdbc.InitialLimit> 40 </entry>
<entry key="jdbc.MinLimit">    40 </entry>
<entry key="jdbc.MaxLimit">    40 </entry>
. . .
```

Tomcat: maxThreads & Connection Pool

`catalina.yyyy-mm-dd.log`

```
oracle.ucp.UniversalConnectionPoolException:
The connection pool named: |apex|pu| is not correctly configured,
due to the following error(s):
   All connections in the Universal Connection Pool are in use
```

Tomcat: maxThreads & Connection Pool

apex_pu.xml

```
. . .
<entry key="jdbc.InitialLimit> 80 </entry>
<entry key="jdbc.MinLimit">    80 </entry>
<entry key="jdbc.MaxLimit">    80 </entry>
. . .
```

Tomcat: maxThreads & Connection Pool

`catalina.yyyy-mm-dd.log`

```
oracle.ucp.UniversalConnectionPoolException:
The connection pool named: |apex|pu| is not correctly configured,
due to the following error(s):
   All connections in the Universal Connection Pool are in use
```

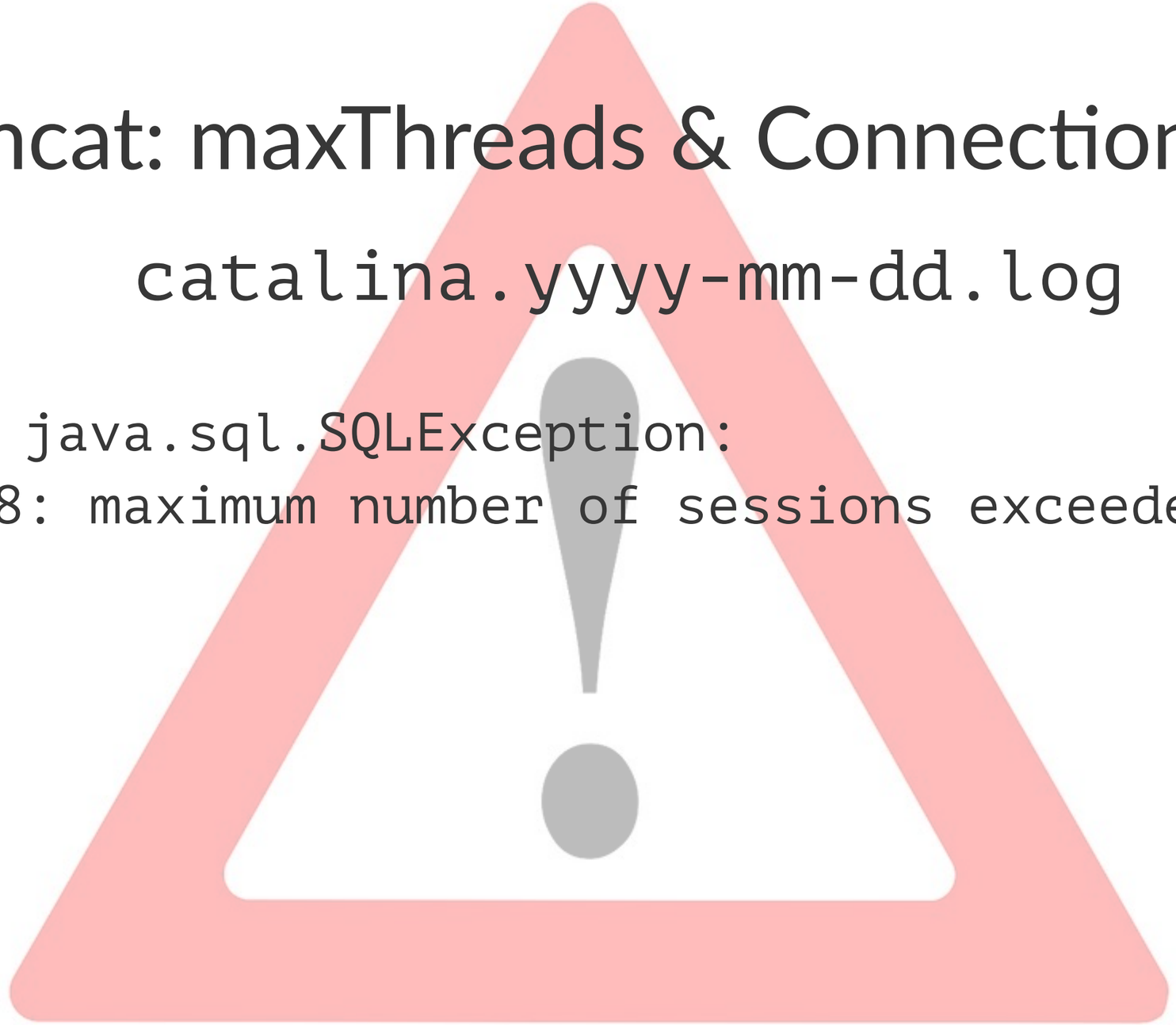
Tomcat: maxThreads & Connection Pool

apex_pu.xml

```
. . .
<entry key="jdbc.InitialLimit> 160 </entry>
<entry key="jdbc.MinLimit">    160 </entry>
<entry key="jdbc.MaxLimit">    160 </entry>
. . .
```

Tomcat: maxThreads & Connection Pool

`catalina.yyyy-mm-dd.log`



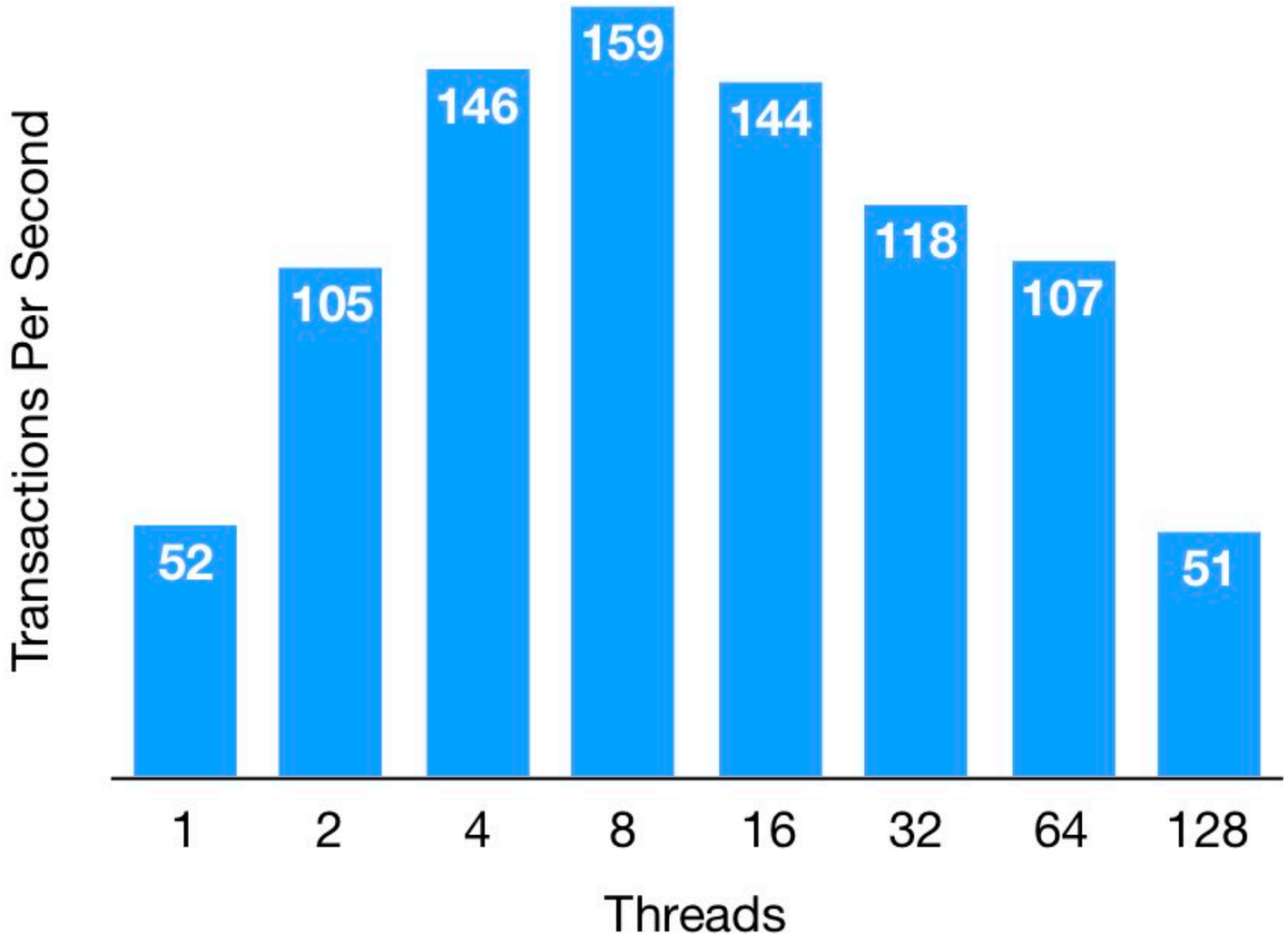
```
WARNING: java.sql.SQLException:
ORA-00018: maximum number of sessions exceeded
```

Tomcat: maxThreads & Connection Pool

`server.xml`

- `Connector` element is HTTP Handler
- `maxThreads` attribute is number request processing threads
- Default value is 200

Tomcat: Threads (on 4 Core VM)



Tomcat: HTTP Connector Protocols

```
[vagrant@ords ~]$ ords-protocols
```

Tomcat: HTTP Connector

`Connector` element: `protocol` attribute

Possible values are

- `Http11Protocol` Blocking Java (to Tomcat 8.0)
- `Http11NioProtocol` Non-blocking Java
- `Http11Nio2Protocol` New non-blocking (from Tomcat 8.0)
- `Http11AprProtocol` APR/Native requires tomcat-native

Tomcat: HTTP Connector Protocols

Default value: `HTTP/1.1`

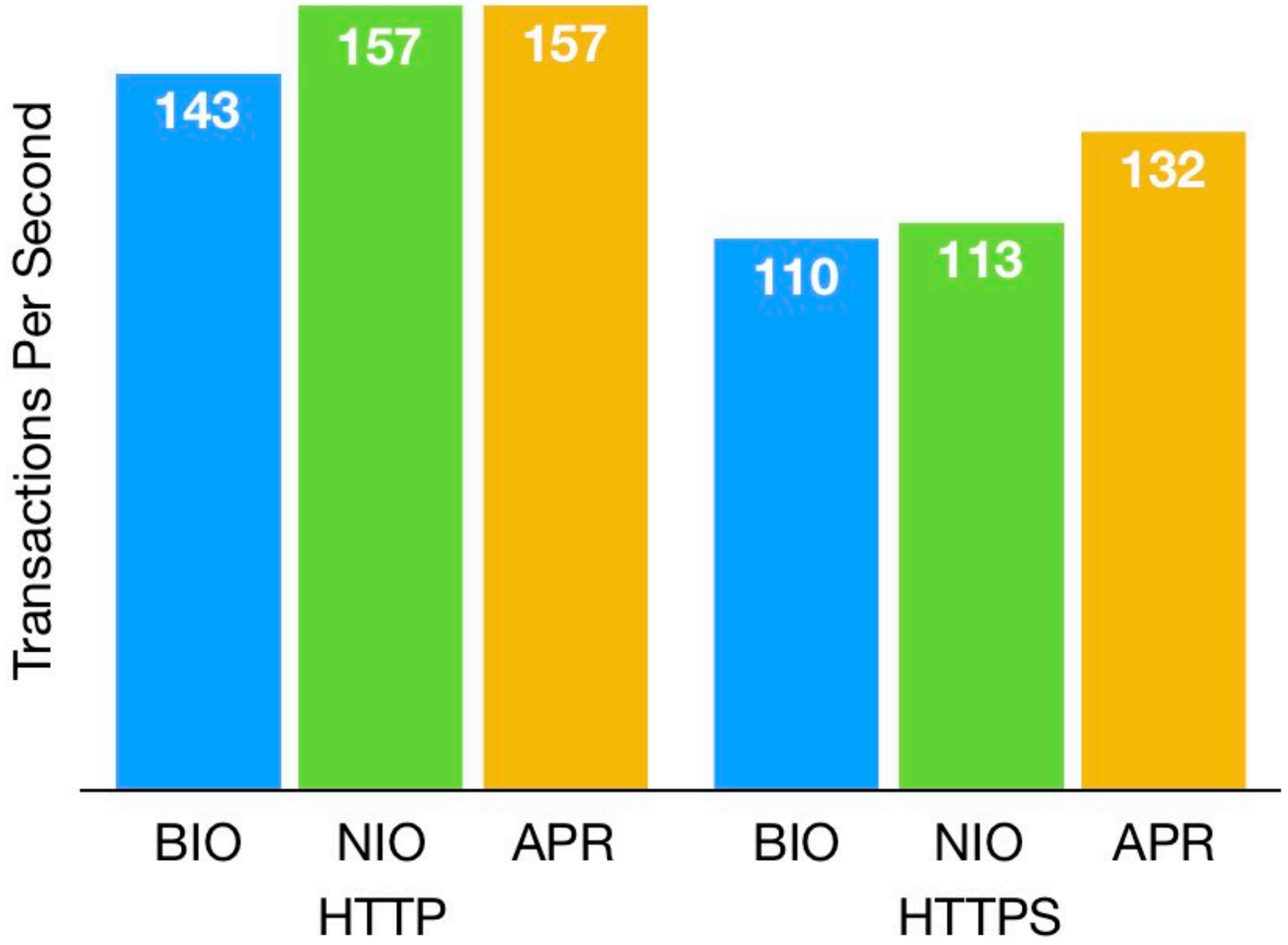
- `Http11AprProtocol` if tomcat-native installed (all versions)
- `Http11Protocol` (to Tomcat 7)
- `Http11NioProtocol` (from Tomcat 8)

Tomcat: HTTP Connector Protocols

SSL Handling

- Http11AprProtocol uses OpenSSL (.crt & .key files)
- Others use Java JSSE (.jks file)

Tomcat: Connector Protocols



Auto & Manual REST

```
[vagrant@ords ~]$ ords-auto-manual
```

Manual REST

```
CREATE OR REPLACE FUNCTION get_employee (
    employee_id IN NUMBER
) RETURN SYS_REFCURSOR AS
    employee_record SYS_REFCURSOR;
BEGIN
   OPEN employee_record FOR
      SELECT * FROM employees WHERE
         employee_id = get_employee.employee_id;

    RETURN employee_record;
END;
/
```

Manual REST: GET

```
BEGIN
   ords.define_template(  p_module_name => 'manual_rest',
                          p_pattern     => 'get_employee/:employee_id');

   ords.define_handler(   p_module_name => 'manual_rest',
                          p_pattern     => 'get_employee/:employee_id',
                          p_method      => 'GET',
                          p_source_type => ORDS.source_type_plsql,
                          p_source      =>
                            'BEGIN :record := get_employee(:employee_id); END;');

   ords.define_parameter( p_module_name => 'manual_rest',
...
```

Manual Rest: GET

```
[vagrant@ords ~]$ wget http://ords:1110/ords/hr/manual_rest/employee/195 \
>                  -qO- | jq
{
  "record": [
    {
      "employee_id": 195,
      "first_name": "Vance",
      "last_name": "Jones",
      ...
    }
  ]
}
```

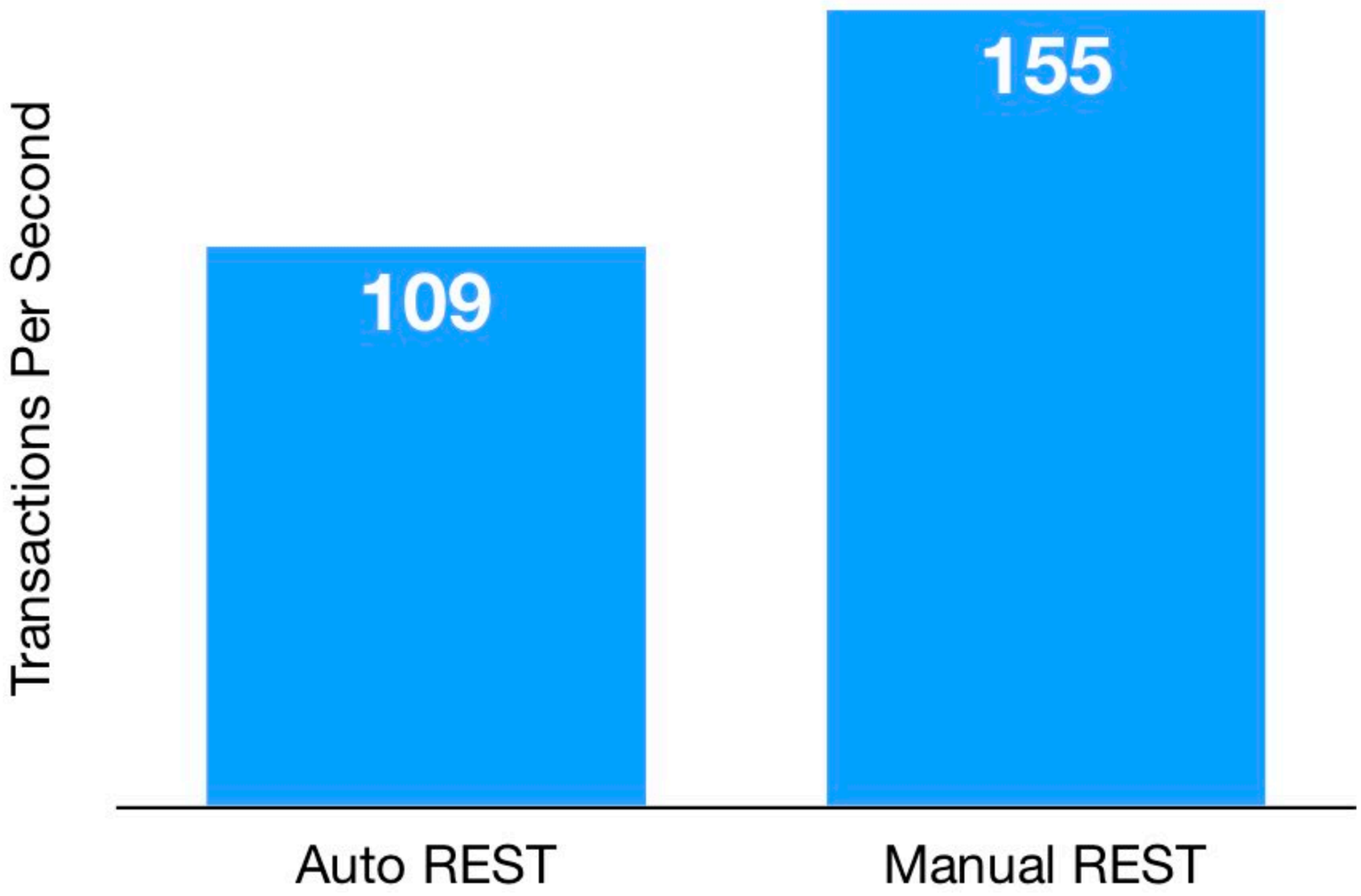
Manual REST: POST

```
BEGIN
   ords.define_template(  p_module_name => 'manual_rest',
                          p_pattern     => 'get_employee');

   ords.define_handler(   p_module_name => 'manual_rest',
                          p_pattern     => 'get_employee',
                          p_method      => 'POST',
                          p_source_type => ORDS.source_type_plsql,
                          p_source      =>
                            'BEGIN :record := get_employee(:employee_id); END;');

   ords.define_parameter( p_module_name => 'manual_rest',
...
/
```

Manual REST: POST

```
[vagrant@ords ~]$ wget http://ords:1110/ords/hr/manual_rest/employee
                      --post-data 'employee_id=191' -qO-            | jq
{
  "record": [
    {
      "employee_id": 191,
      "first_name": "Randall",
      "last_name": "Perkins",
      ...
    }
  ]
}
```

ORDS: Auto & Manual REST



Reverse Proxies under Siege

```
[vagrant@ords ~]$ ords-rev-proxy
```

Apache HTTP Server (HTTPD)

httpd.conf: HTTPS Cache GETs

```
...
CacheDetailHeader on

Listen 2220 https
<VirtualHost *:2220>
    SSLEngine on
    SSLCertificateFile    /usr/local/ssl/ords.crt
    SSLCertificateKeyFile /usr/local/ssl/ords.key
    ProxyPass             / http://ords:1110/
    ProxyPassReverse      / http://ords:1110/
    CacheEnable           disk  /
    CacheRoot             /var/cache/httpd/
    CacheDefaultExpire    3600
</VirtualHost>
...
```

nginx

"Apache has a million options but you only need 6. Nginx does those 6 things and does 5 of them 50 times faster than Apache."

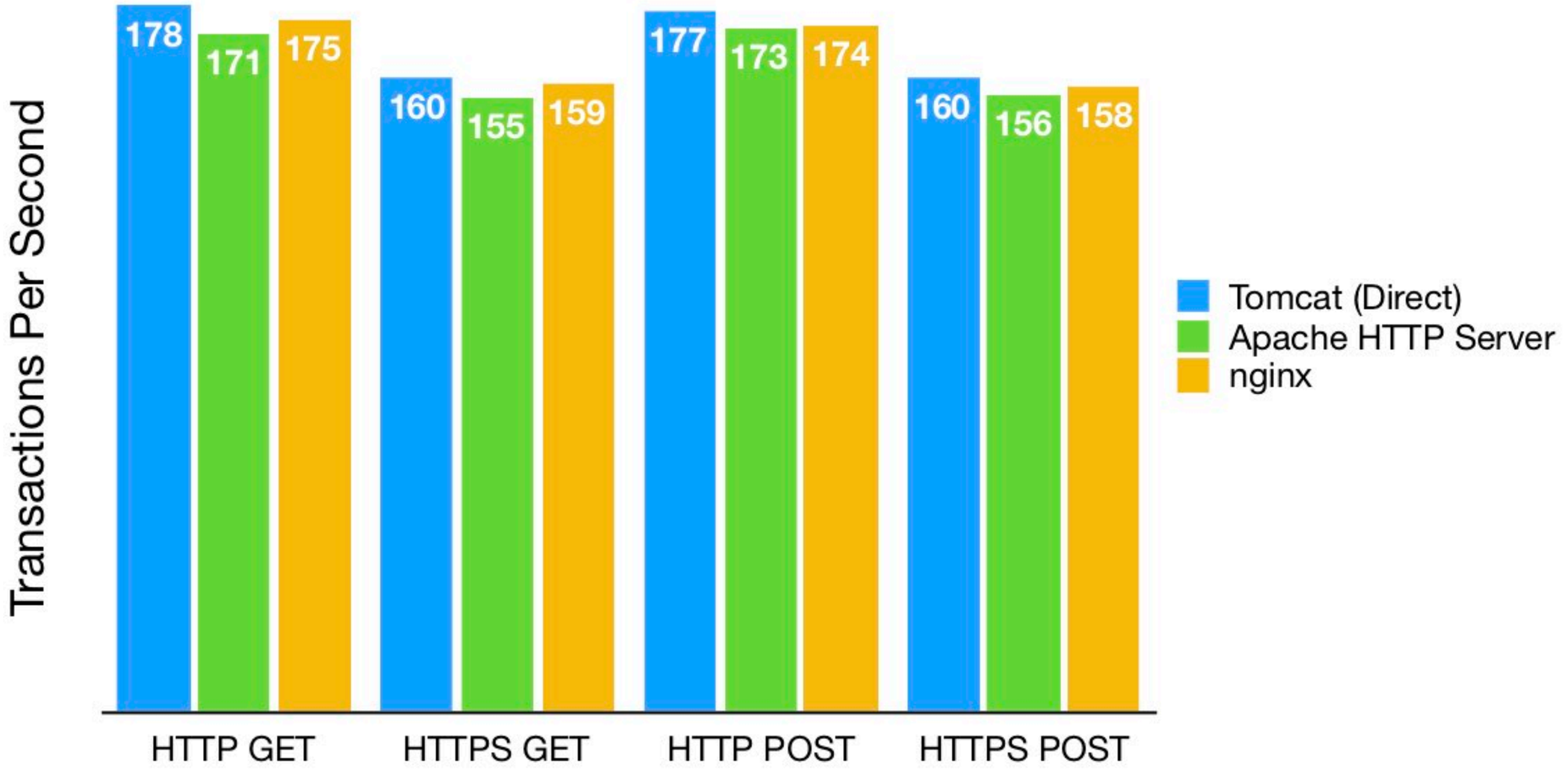
nginx.conf: HTTPS Cache GETs

```
http {
   proxy_cache_path /var/cache/nginx keys_zone=ORDS-CACHE:128m;
   add_header X-Proxy-Cache $upstream_cache_status;

   server {
      listen               3220 ssl;
      server_name          ords;
      ssl_certificate      /usr/local/ssl/ords.crt;
      ssl_certificate_key /usr/local/ssl/ords.key;
      location / {
         proxy_pass         http://ords:1110/;
         proxy_cache        ORDS-CACHE;
         proxy_cache_valid 60m;
      }
   }
...
```

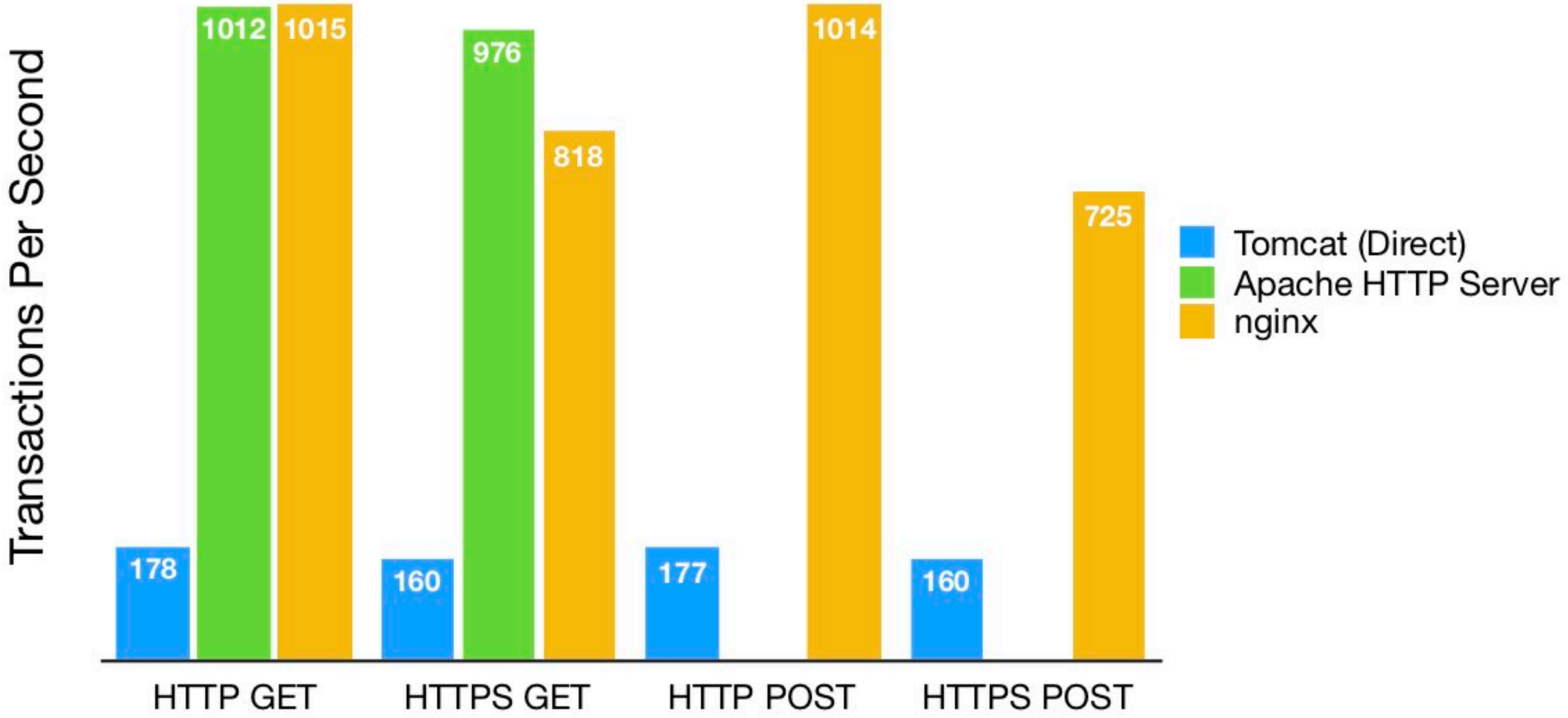
nginx.conf: HTTP Cache POSTs

```
server {
    listen 3130;
    location / {
        proxy_pass              http://ords:1110/;
        proxy_cache             ORDS-CACHE;
        proxy_cache_valid       60m;
        proxy_cache_methods     POST;
        proxy_cache_key         "$uri|$request_body";
    }
}
```

Reverse Proxies: Pass-Through



Reverse Proxies: Caching



Summary: Tomcat & ORDS

- Use tomcat-native
- Use manual (not auto) REST
- Use correct `maxThreads` & Connection Pool settings

Summary: Reverse Proxies & Caching

- Cache as early as possible for
  - Better throughput & response times ✓
  - Less load on database & app servers ✓
- Reverse Proxies have pros & cons, pick one that suits your needs

Thanks for listening!

Any questions?