



Using OpenSSL to boost Tomcat

Jean-Frederic Clere

What I will cover

- Who I am.
- Connectors
 - NIO, NIO2, APR
 - OpenSSLImplementation
 - HTTP/2 and ALPN in Tomcat.
- Performances tests
 - With ab and h2load as client load generator.
- Questions?

Who I am

Jean-Frederic Clere

Red Hat

Years writing JAVA code and server software

Tomcat committer since 2001

Doing OpenSource since 1999

Cyclist/Runner etc

Lived 15 years in Spain (Barcelona)

Now in Neuchâtel (CH)

Tomcat



What is a Connector?

- Tomcat's interface to the world
- Binds to a port
- Understands a protocol and possible upgrades.
- Dispatches requests (example)
 - `protocol="org.apache.coyote.http11.Http11AprProtocol"`
 - `protocol="org.apache.coyote.http11.Http11NioProtocol"`
 - `protocol="org.apache.coyote.http11.Http11Nio2Protocol"`

Tomcat Connectors

- Java Non-blocking I/O (NIO)
- Native / Apache Portable Runtime (APR)
- Java NIO.2

Technically, there are combinations of all of the above with HTTP and AJP protocols.

The presentation focuses on HTTP and on NIO/NIO2.

What is new in Tomcat 9 / 8.5

- Property sslImplementationName
 - Allows replacement of the SSL code
 - OpenSSLImplementation (use OpenSSL)
 - JSSEImplementation (use JSSE)
- UpgradeProtocol
 - Allows protocol upgrade from HTTP/1.1
 - HTTP/2 (yes)
 - WebSocket/Speedy (cool).

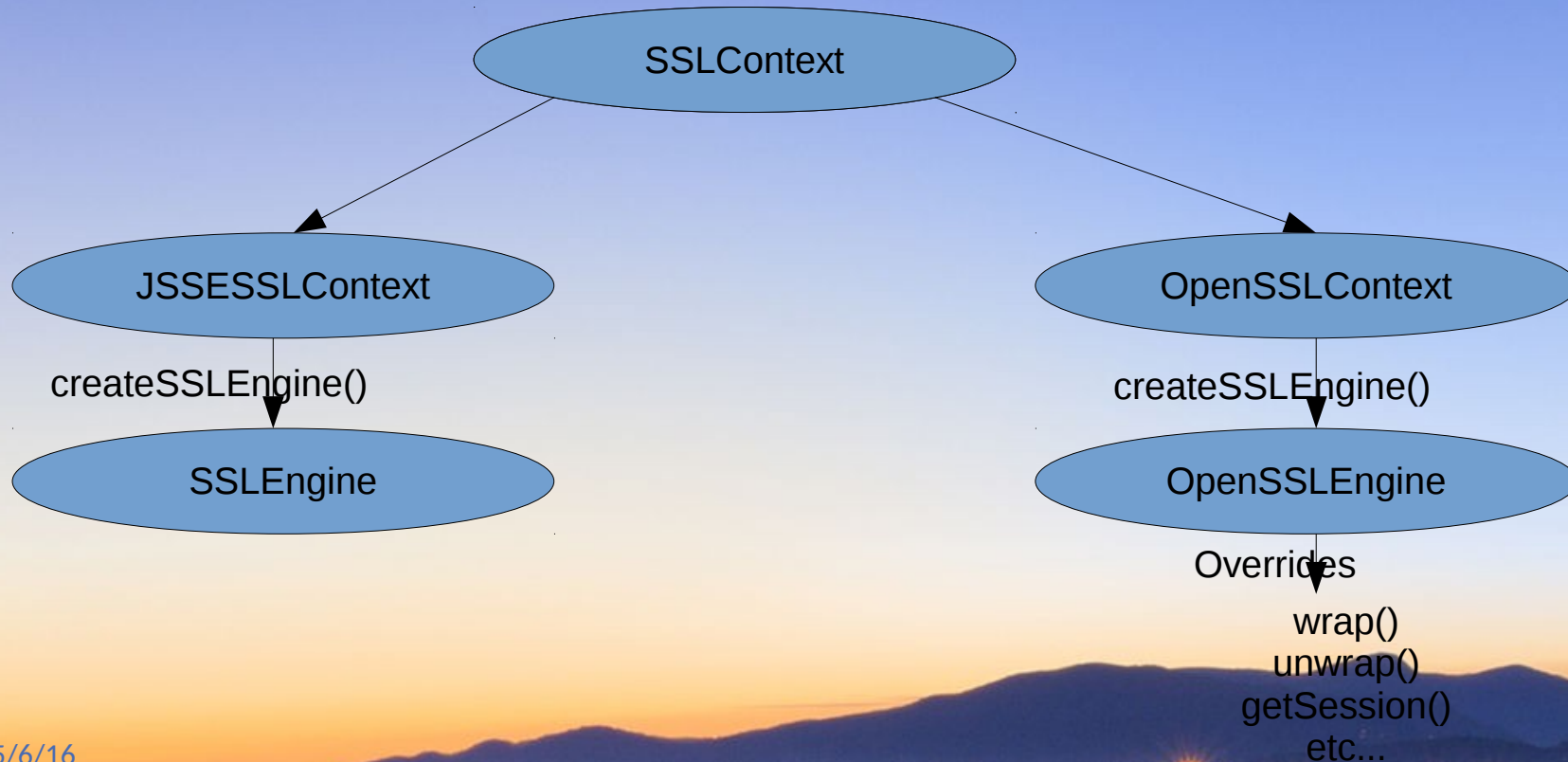
Why a new SSLImplementation

- JSSE:
 - Very slow
 - Missing features: like ALPN (JEP 244: TLS Application-Layer Protocol Negotiation)
 - Hardware acceleration very partial (like AES in java8)
- Native connector:
 - Fast but a lot of native code
 - Use OpenSSL for SSL/TLS.
- New OpenSSL implemetation:
 - Fast.
 - Uses only a OpenSSL for native code (no native socket, poller etc).
 - Works with NIO and NIO2.
 - Uses OpenSSL for SSL/TLS. (warp, unwarp, handshake etc).

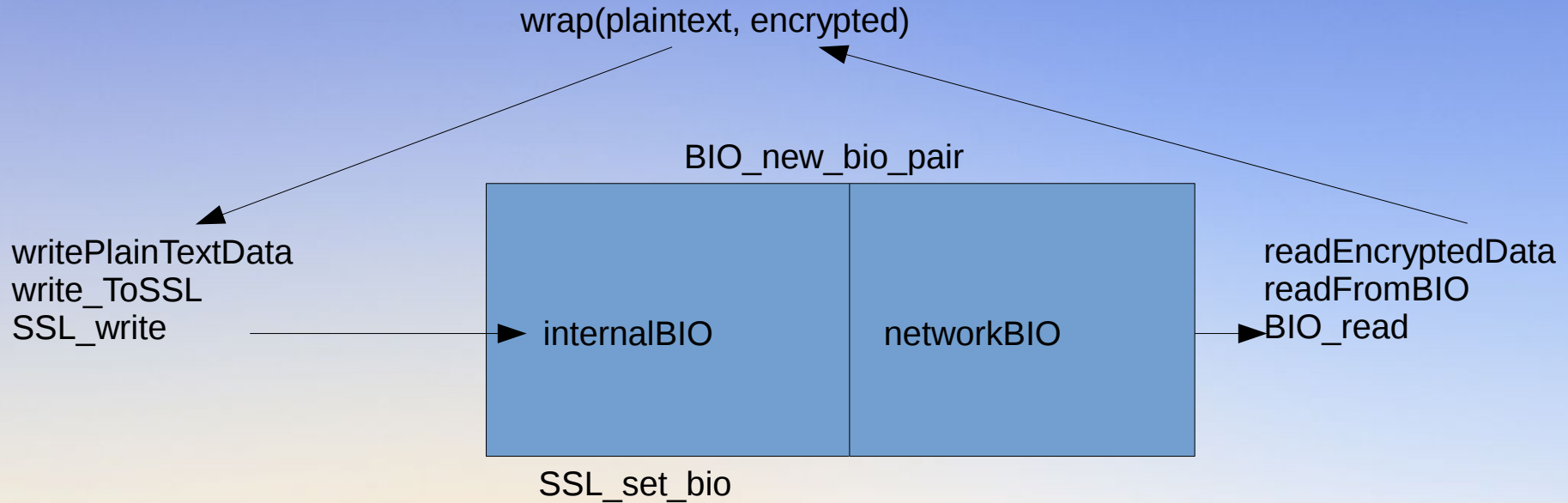
OpenSSL Implementation

- Code originates from netty-tcnative a forked Tomcat Native
- Prototype (last year):
 - Done with the BeFriNe University
 - Tested and ported to tc_trunk last summer
- SSL Configuration compatible with the JSSE connection (*)
- Uses keystores (*)
- Uses SSL BIO to wrap/unwrap, handshake
- Uses java NIO or NIO2 Sockets for the reads and writes
- Automatically enabled when TC native is installed/enabled (*)

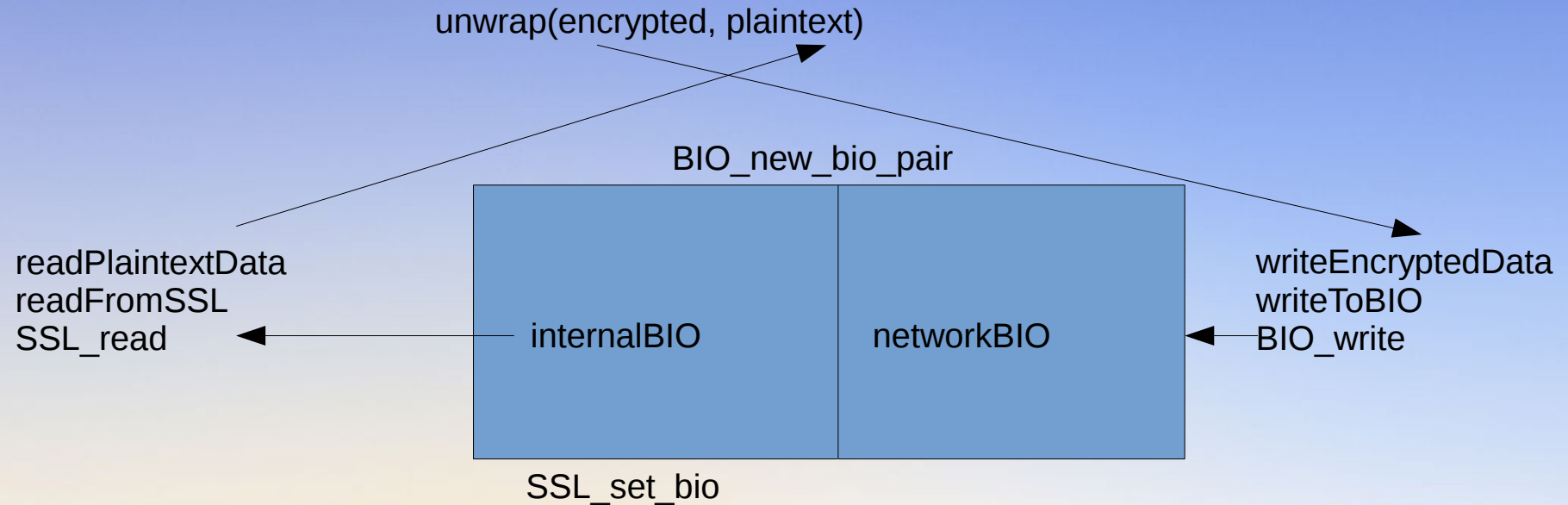
How does that works



How does wrap works



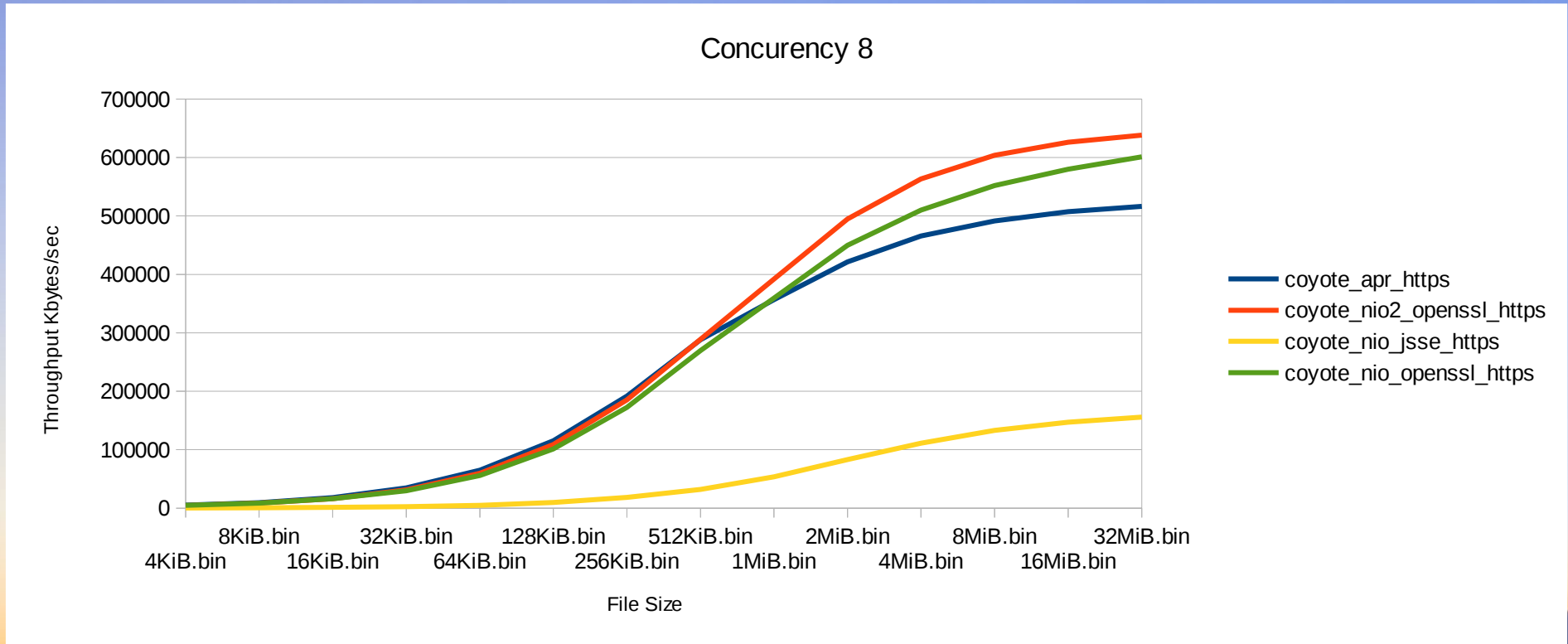
How does unwrap works



Connector Performance

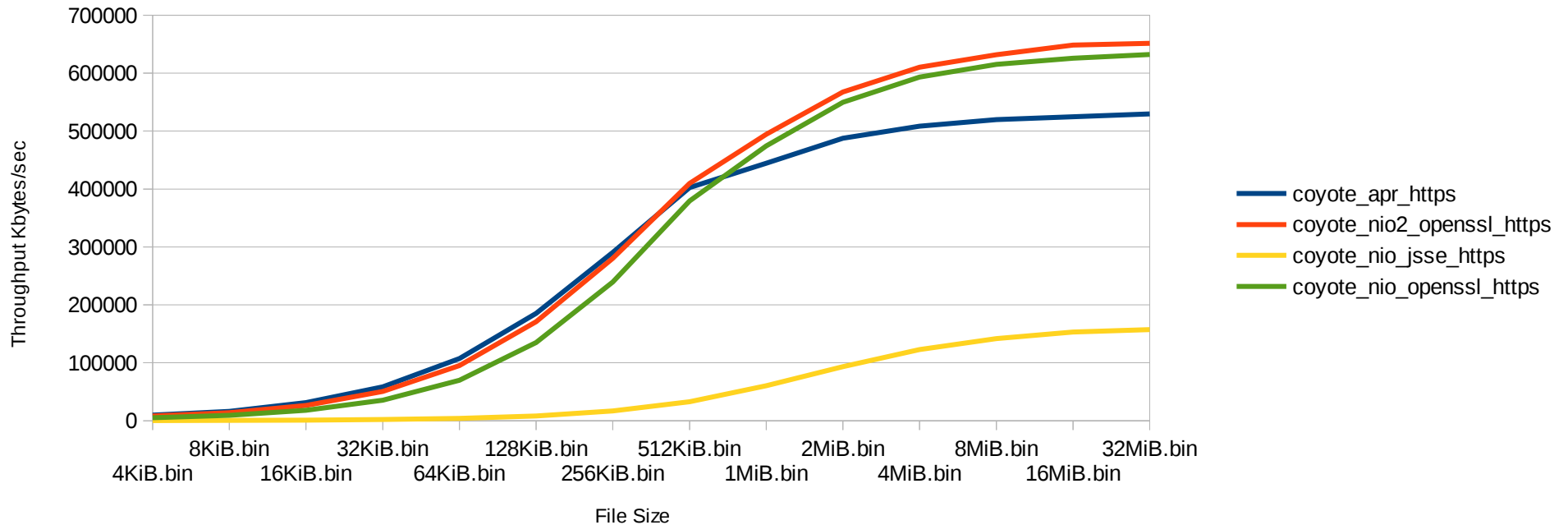
- Compare connectors throughput against each other
- Only static content was compared, varying file sizes
- Run on “fast” machines, 10 Gbps local network
- Tests:
 - Compare the connectors (trunk) NIO, NIO2 and APR
 - Using JSSE and OpenSSL
 - First without “sendfile”

Connector Throughput (c8)

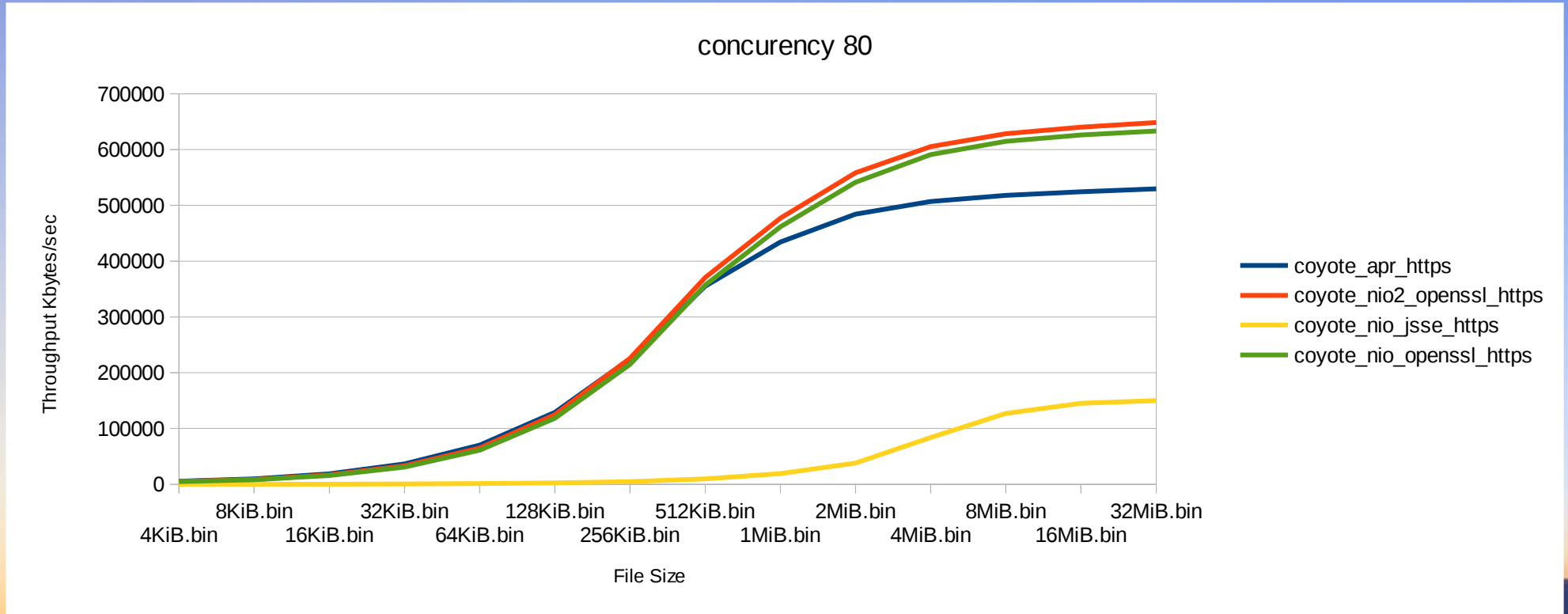


Connector Throughput (c40)

Concurrency 40

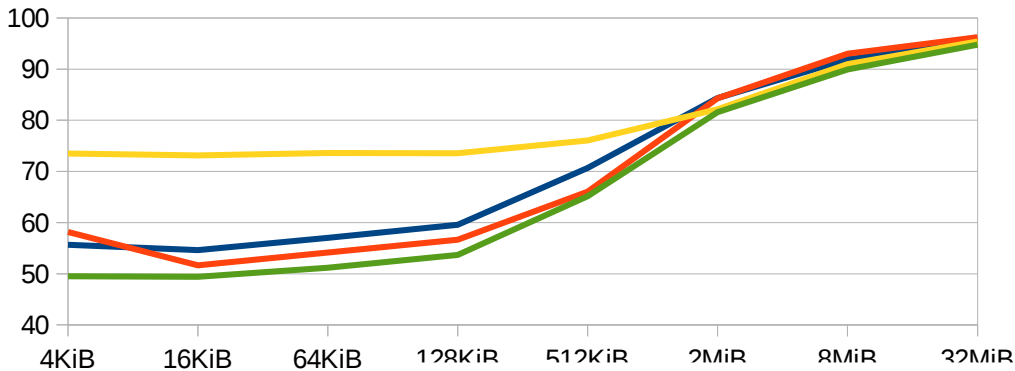


Connector Throughput (c80)

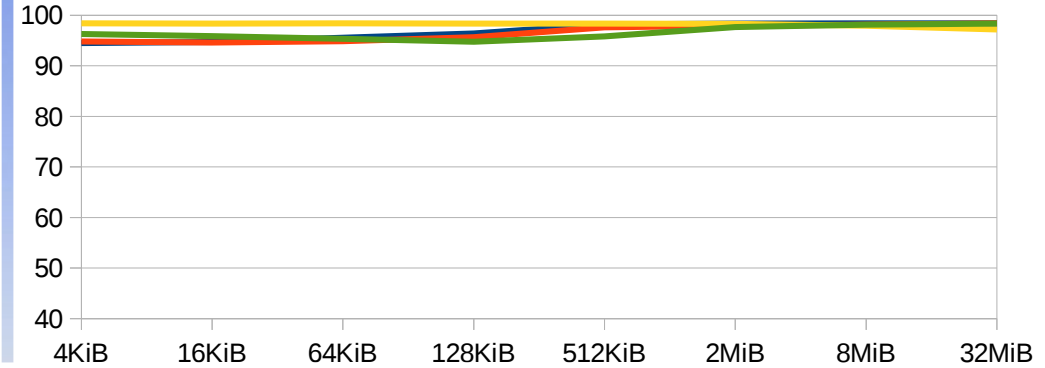


Connector CPU Use

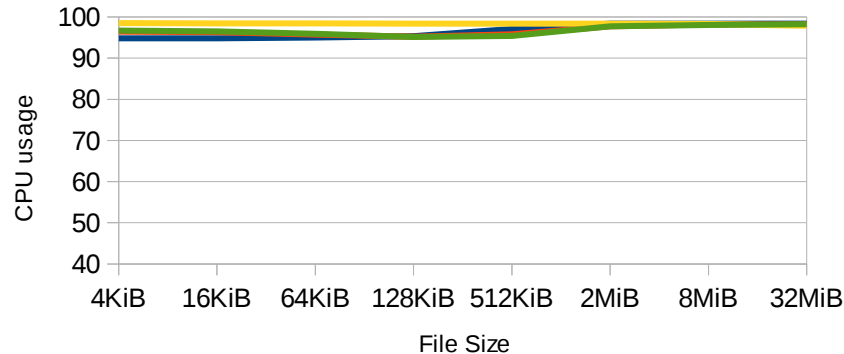
Concurrency 8



concurrency 40



Concurrency 80



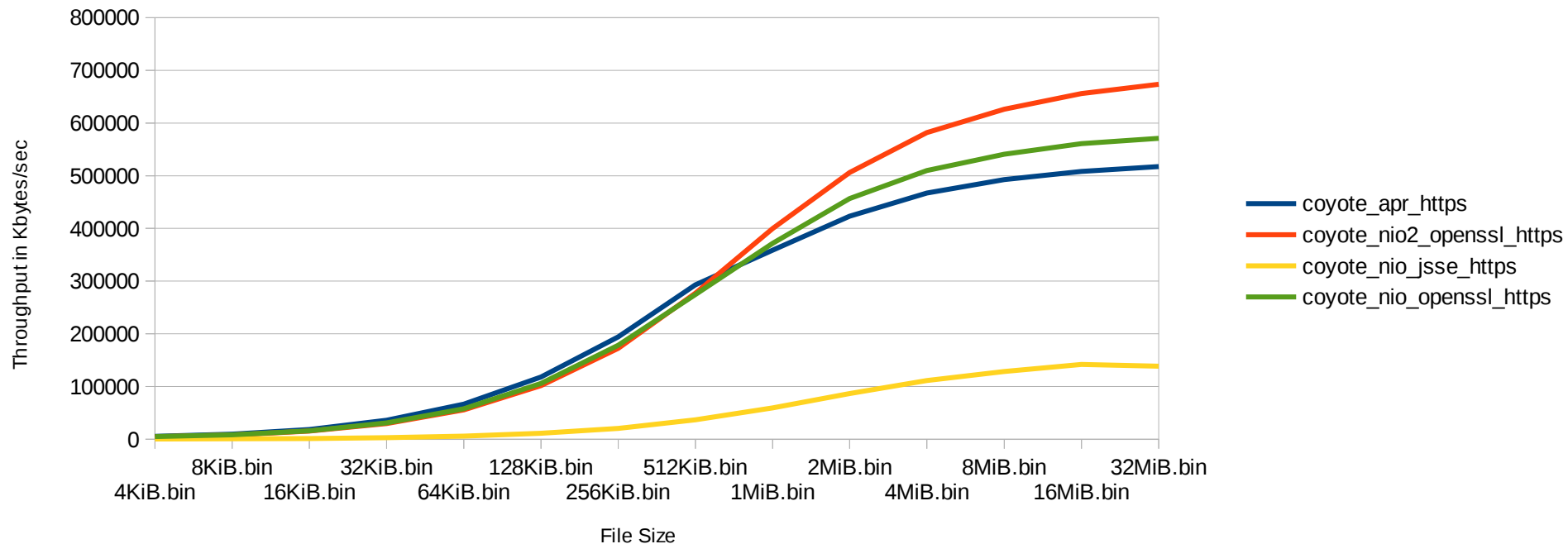
- coyote_apr_https
- coyote_nio2_openssl_https
- coyote_nio_jsse_https
- coyote_nio_openssl_https

Connector Performance

- With sendfile
 - In fact with TLS/SSL sendfile is emulated

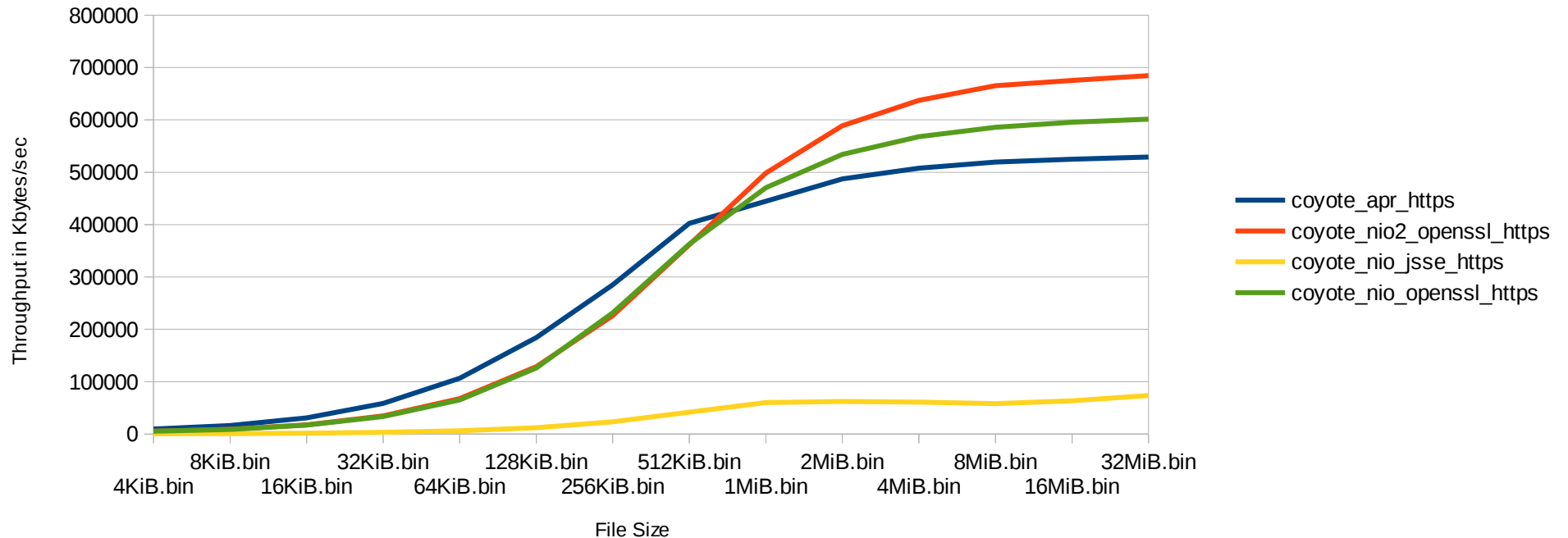
Connector Throughput (c8)

Concurrency 8

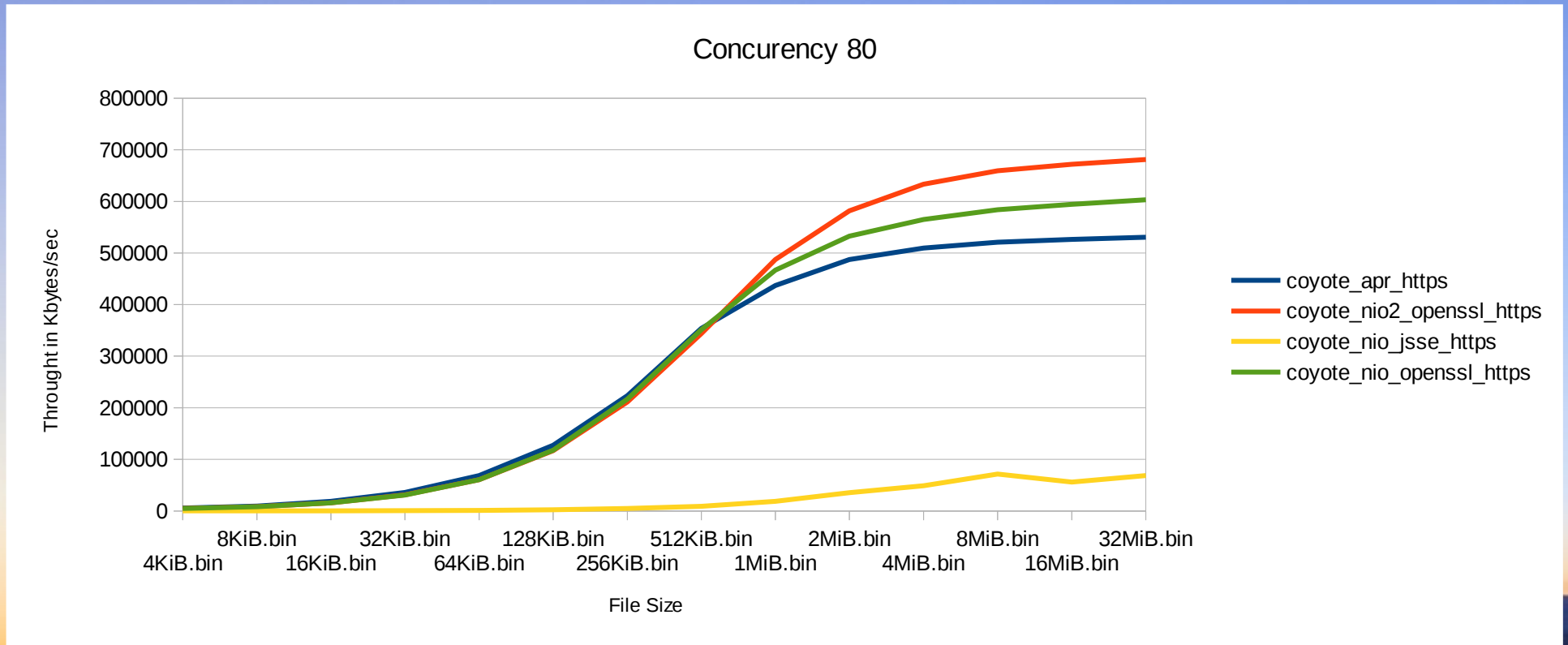


Connector Throughput (c40)

Concurrency 40

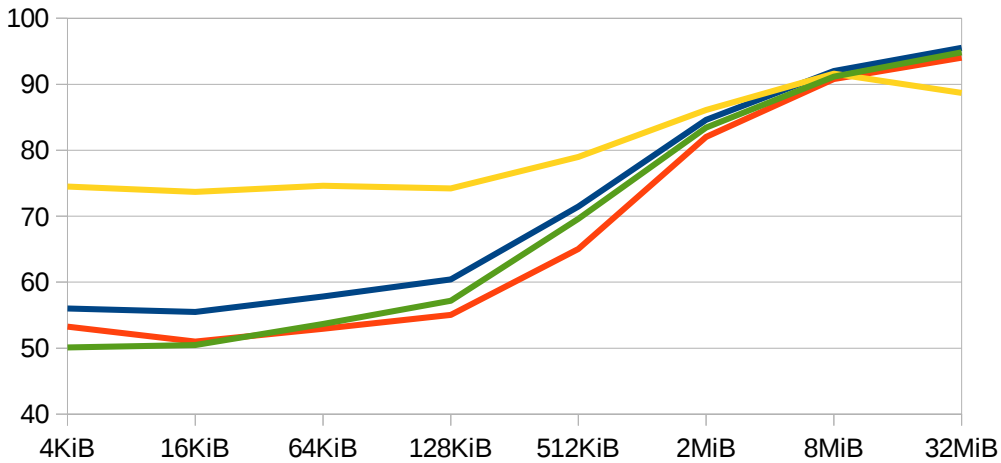


Connector Throughput (c80)

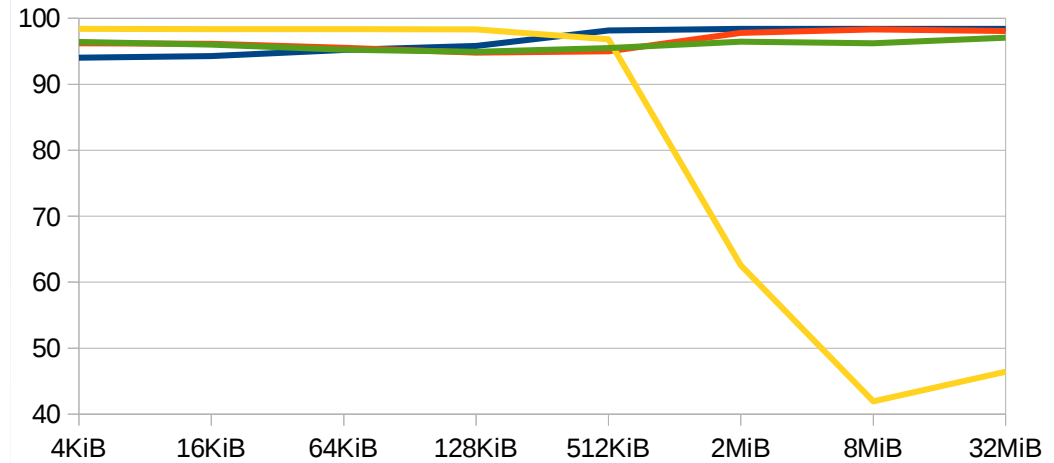


Connector CPU Use

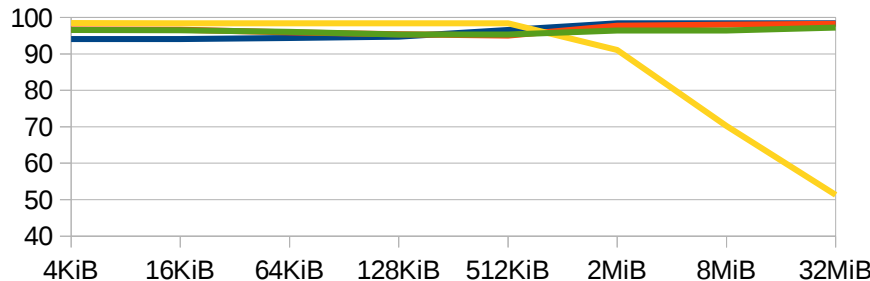
Concurreny 8



Concurency 40



Concurency 80



- coyote_apr_https
- coyote_nio2_openssl_https
- coyote_nio_jsse_https
- coyote_nio_openssl_https

Connector Performance

- Conclusion:
 - OpenSSL performs better than JSSE
 - NIO and NIO(2) give similar results
 - Emulated sendfile doesn't help a lot (bigger files better).
 - APR isn't needed
 - Until Java9 is released OpenSSL is needed for HTTP/2

Questions? Thank you!

- jfclere@gmail.com
- users@tomcat.apache.org
- Repo with the scripts for the tests:
 - <https://github.com/jfclere/AC2014scripts>

Jean-Frederic Clere

@jfclere

jfclere@gmail.com



APACHECON
NORTH AMERICA