Stephen D. Cope

sdc.org.nz - +65 9231 1703

Financial domain knowledge: FX market making, pricing, trading, and hedging.

Infrastructure knowledge of Linux, system administration, software release cycle.

Vast breadth of experience across IT: networking, load balancers, TLS, Solace, Oracle, SAN.

Communications formats on the wire: FIX, TLS, IMIX, Webshere MQ
Big Data / Business Intelligence: Splunk, Tableau, R, Logstash, ElasticSearch, Kibana
System administration: scripting { bash, Ansible, puppet }, monitoring { Splunk, Logstash, ElasticSearch, SNMP, Ganglia, MRTG, sysstat, sar }, scheduling { AutoSys, at, cron }, networking { tcpdump, nc, nmap, ss, route, ip[6]tables, IPv6, r[t]advd }, rsync, git, svn, ...
Linux engineering and administration: RHCSA, CentOS / Redhat, Mint / Ubuntu, OpenBSD
Programming languages: bash, Ruby, R, Java, Python, C, C++, ASP.NET, PHP, make
Maintenance programming: perl, C#, JavaScript (jQuery)
Middleware: Solace, Webshere MQ
Representational encoding formats: XML, JSON, XHTML, HTML, CSS, vCalendar
SQL: Oracle, MS SQL, MySQL; BerkeleyDB, Sybase and SQLite also
Regular expression: grep, sed, Perl-compatible
Distributed computing environments, incl. compute clusters: Oracle/Sun Grid Engine
Daemons: Apache httpd, nginx w/- LibreSSL, Postfix; formerly qmail, Mailman, squid
Standards: HTTP/1.1, POP3, SMTP, mail (RFC2822), MIME, DNS, NPN, ...
Cryptography: regular user of GnuPG and S/MIME, personal website gets A+ from Qualys, managed a certificate authority (using OpenSSL), former Thawte Web of Trust member, find me on https://keybase.io/tipene 40DD 95E4 4778 0BA2 8B0E 7C08 6E1F 88E4 B3B4 BBF5

Career History

2014 – Present: Standard Chartered Bank, Singapore - S2BX Production Support

- Provide technical and function support for high frequency, low latency eComm trade capture system: Straight2Bank Exchange (S2BX) and downstream to DealHub.
- Infrastructure maintenance and diagnostics from application level, protocol, transport, server hardware, networking, firewalls, cabling. If the application touched it, I examined it.
- Interaction with traders, sales and external clients (API clients, ECNs, difficult GUI clients) including site visits with Sales.
- Handled business issues including diagnosing rate publishing by traders, spreads added by sales, through to explaining ultimate prices blended and presented to customers.
- Assisted with addition of new Liquidity Providers: tested network connectivity, ensured processes were running and monitored, checked liquidity, arranged test trades with counterparties and internal traders and confirmed STP.
- System monitoring with ITRS Geneos, Splunk, and BMC Patrol. Built multiple dashboards in ITRS Geneos and Splunk, as well as dashboards, reports, and alerts in Splunk. Interacted with Splunk Professional Services to fine tune and improve our Splunk setup. Helped push a Geneos version upgrade for Netprobes, Gateways, and Active Console.
- Maintenance and creation of support tools: scripts for managing common tasks.
- Tracking trades from capture in S2BX down to OCL DealHub and on to Murex, etc.
- Capacity management for existing servers. Requesting budget and meddling in hardware intake process to get new servers delivered with the correct specifications. User acceptance tests of new server builds to ensure the application can run correctly.
- Handling client certificates for our STP and FIX connections.
- Support for GUI clients: Java issues, user queries that couldn't be handled by L1; TLS negotiation.
- Internally, the face of GUI delivery to clients: Apache httpd, Netscaler load balancers,

SSL offload and client certificate verification, examing messages captured and diagnosing network issues.
- Network connectivity issues: firewalls, proxies, routing issues, hardware failures and replacement.
- Provided help with FIX client and ECN onboarding and go live.

Notable achievements;

(1) Expired quotes for client D

A client experienced trade rejects due to latency. Investigation showed that the latency appeared to be within bounds and the prices should have been able to be consumed in a timely fashion, but the latency was neither consistent nor the same as advertised for the link.

I examined the packet capture of the connection in WireShark as well as application logs in Splunk. From the application logs the round trip time (calculated by examining our sending time for a TestRequest FIX message to receipt of a corresponding HeartBeat) was some 40 – 50ms higher than the advertised line latency from the vendor. Cross-referencing with the packet capture I could see that the ACKs for the TestRequest packets were received in a timely fashion but the inbound HeartBeat did not arrive until 40 – 50ms later according to the application log.

From this I concluded that the client's application was suffering from CPU shortage on the remote side and unable to consume the prices in time. I asked the vendor to please move their FIX engine to a less loaded host. They admitted there was little done in the way of monitoring CPU utilisation and they would shift it to a less loaded host. After this was done the round trip latency for the application decreased.

(2) Strange disk I/O patterns uncovered

Something of a paradox exists within our application in that it both writes a lot of log files but also attempts to be low latency. How could this co-exist?

I configured additional samplers in Splunk so that we collected more data on the underlying performance of the Linux servers where our application runs. I frequently sampled /proc/meminfo, /proc/interrupts, iostat, and vmstat – amongst others.

From Splunk I parsed and extracted the output to collect memory and I/O statistics for a host of interest. A 24 hour period of data was dumped into multiple CSV files. This I read into R and began an exploratory analysis. Individually each of the metrics (free, cached, buffers, I/O writes) looked acceptable but by plotting the dirty memory against I/O writes I could see that there were regular spikes. There would be sudden large jumps in dirty memory, followed by peaks in I/O output that would decrease until the dirty memory was low.

When presented to the developers they identified that particular mmap()ed files would be closed at regular intervals, leading to this behaviour.

(3) Splunk for infrastructure monitoring

Amongst the Splunk samplers is one I added to collect the NIC state. Splunk sends its data out via a management network interface. In the event of failure or failover of the bonded application network interface a Splunk Alert will generate an email to indicate that a NIC requires attention. Not really the best tool for the job, but it's what we had on hand and it works better than the existing tools. A similar setup is used to monitor directory ACLs to check for consistency and changes.

(4) Latency for client O

A client complained of latency issues when streaming large numbers of quotes. An initial packet capture showed that there would be periods during which the client would send duplicate ACKs causing us to retransmit data, causing delays as the quoting connection caught up. This indicated packet loss. I suggested we split the quoting streams into four sessions so that packet loss on one session would not cause the other streams to be backlogged.

Comparing packet captures from the client and from our side showed we still had issues with duplicate ACKs and retransmits, and loss on one stream would happen at the same time as on

other streams. BT checked their settings and discovered that one traffic shaper (?) was incorrectly configured, causing packet loss.

When this was corrected service resumed. I used the amount of unacknowledged data in the FIX engine's TCP Send Queue, divided by the number of streams the FIX engine was pricing, the time between quotes, and the known round trip time of the line to calculate whether the number of in-flight prices was within expected bounds. It was. Now the issue had shifted up the OSI stack to the business layer, which I also analysed – including a simulation in R to model different amounts of latency – and presented my findings for, but they were less well received.

2012 – 2014: Barclays Capital, Singapore - BARX FX Application Management

- Provide technical and functional L2 support for high frequency, low latency eComm trade capture system: BARX FX (ranked #3 in 2014 Euromoney survey).
- High visibility role looking after Criticality A application with numerous connected systems internally and externally.
- From market connectivity for pricing and orders, through to quoting for FIX and GUI clients, execution and booking.
- Member of second line team providing follow-the-sun support to business and internal users.
- Providing assistance with FIX connectivity: connection issues, latency tracking and resolution, firewalls and network tests, tracking FIX conversations to confirm trade status and origin.
- Support for GUI clients: tracking down sources of latency, Java issues, user problems.
- Maintenance and creation of support tools: scripts and canned SQL for common queries; development of web tool used by L1 support, trading, operations for querying trade data.
  Statistical analysis of system usage levels to monitor impact of outages.
- Big data log analytics using Logstash and ElasticSearch. Researched, implemented, and trained staff on system to instantly search across thousands of logs to allow for self-servicing queries.
- Booking assistance: tracking orders from the inbound FIX, GUI, market, or internal system through to message queues to downstream systems: Murex, Devon, other in house systems.
- Resolution of pricing queries: investigation and recalculation of static, spreads, order stack, etc.
- ITIL & Change Management: requesting, coordinating, or implementing changes for production system. Communication with multiple teams: Unix, DBA, Networks, business. Representation to change management boards.
- SME for Research tool used for technical trading.
- Joined Barclays Capital from the University of Auckland because I was looking to be part of a larger team with more challenging problems to solve.

Notable achievements;

(1) Introducing Logstash and ElasticSearch with Kibana (ELK)

BARX FX is a distributed application over hundreds of servers in multiple DCs. To track down a trade required logging in to multiple servers and examining many log files. Also due to retention policies particular items in the log files were rotated out to tape after 30 days, but investigations typically required data from 15 – 75 days earlier.

I located some unallocated servers in our estate as well as hardware from decomissioned servers and put together a Logstash, ElasticSearch, and Kibana setup.

So as not to run additional software on servers I wrote a log shipping setup which would pipe selected logs off the host to where the Logstash servers were running. This used tail and ssh (multiplexed) along with sophisticated safeguards in bash to catch up on missed files and prevent duplicate transfers. In normal operation the log would be spooled to disk, which Logstash would read. Logstash would parse the logs, extracting key fields, and categorising or discarding lines. This I incrementally approached, adding log lines with extractions as

requested by the support staff, to suit customer needs, or when Dev asked for specific monitoring. After the log file was read into Logstash – I compared the ingested size of the file as recorded in the Logstash inode cache against the size of the file on disk – it would be truncated. In this way the Logstash host would have a rotating selection of files and didn't require as much storage on the parsing hosts. Files could be re-read by using another script which would 'catch up' any piped files by using rsync. The entire pipeline was parallelised so that bottlenecks in parsing or submitting to ElasticSearch would not hold up other log files.

For ElasticSearch I setup custom retention policies so that complete data would be held for a period dependent on available disk space and then other data for a longer term. This was written in bash, used the ElasticSearch web calls, and would be called by AutoSys.

The Kibana front end was introduced to the teams that depended on the data and I set up different dashboards to report on or search for different data: trade ids for downstream reconciliation, database performance metrics for the developers, trade volumes, GUI login locations, FIX login times and trading behaviour, etc.

By adding analytics and cross-host log search, without placing additional load on the production hosts, we were able to more quickly identify log files of interest, see performance figures that were previously hidden or difficult to report on (the developers were surprised to see what our most popular stored procedures were), and allow some of our common queries from internal users to be self serviced without requiring a log restore from tape!

2003 - 2012: The University of Auckland - Department of Statistics

High Performance Computing (HPC) consulting for staff and postgraduate students. System administration, programming, management of Department’s multimedia resources,

In 2003, started with the job of maintaining the website, including programming the in-house content management system, and writing documentation. Grew to include keeping the web server and mail server working, system monitoring, and system administration. Documentation has grown to include tutorials, seminars, and evangelism. Have located and hired three additional programmers for work on individual projects.

Work on the CensusAtSchool project includes web-based visualisation tools that work in with the sampler. Coded the survey, sampler, registration, and visualisation tools.

High performance computing advice for students and staff, assisting them to rewrite their code to take advantage of parallelism (including OpenMPI and OpenMP). I configured and optimised the Sun Grid Engine scheduler to allow for allocating hosts and distributing jobs around the available compute servers. I also configured additional queues to allow researchers priority access to servers their grants had paid for.