
# OpStack Discussion Document
# Cloud-Integrated Trading Infrastructure

By Evan Bauer & Jared Trog
August 12, 2020

# Cloud-Integrated Trading Infrastructure

## Discussion Document - 2020-08-09



New York ▪ Toronto ▪ Dallas
email: info@opstack.com / tel +1-416-806-3711

## Leveraging Cloud Technology as Trading Infrastructure

Trading systems have long exploited new technologies to give market participants advantages in understanding, strategy, and timing to improve outcomes. The increasing exploitation of latency reductions for market advantage has made co-location with major venues critical -- leading market participants to make increasing capital expenses for on-premise and co-lo infrastructure.

For much of IT, the major trend in the second decade of the century is the move to leverage major cloud service providers to provide flexibility, innovation, and capital efficiency in providing services. Regulatory, control, and latency requirements for trading systems have made them one of the last application areas to move to the cloud. OpStack believes that with thoughtful architecture it can be remarkably effective to build and operate trading systems using a hybrid cloud approach.

In consultation with the lead architect at a financial services firm, OpStack developed this reference infrastructure to allow leveraging public cloud services to substantially reduce the capital investment needed to enter a new market without sacrificing any competitive advantage to other participants fully co-located with the electronic trading venues. In this architecture, the presumption was to use cloud neutral technologies (Docker, Kubernetes, et al) with the possibility of replacing some of the less latency sensitive elements with PaaS elements -- we envisioned the trade history databases leveraging a cloud DBaaS (Database as a Service) offering rather than building and maintaining dedicated clusters from scratch on cloud IaaS (Infrastructure as a Service).

OpStack uses discussion documents, like this one, as a means of creating conversations with partners that lead to business-specific solutions. We specified this candidate architecture in a fair amount of detail in order to illustrate its viability and cost-effectiveness, with the understanding that a final solution for any client and market will be at least adjusted if not redirected by that ongoing discussion.

## Requirements and Best Practices

| Requirement | Benefit |
|---|---|
| Low-latency access to Electronic Communication Networks (ECN) | Competitive trading posture in time sensitive markets |
| Hardware infrastructure (servers, switches, and firewalls) that support the exploitation of low-latency data access through co-location with ECNs | The ability to translate algorithms and strategies into profitable transactions executed at the speed of the market |
| Infrastructure does not dictate limits of application capabilities | Infrastructure needs to support what the business and market require and be able to adapt as those requirements evolve |
| Automated CI/CD pipeline fully integrated with infrastructure automation | Removing manual tasks for code deployment and infrastructure changes reduces the opportunity for error and lowers time-to-market for new capabilities |
| Consistent environments across both production and pre-production | Minimizes code promotion costs and risks, critical for rapidly changing applications |
| Ephemeral infrastructure with servers, storage, network, and (optionally) containers fully defined in software | The application isn't tied to any hardware provider or cloud vendor and can be moved or scaled up or down as requirements change |
| Parallel independent environments, both production and non-production | Parallel new version development and production support -- plus blue/green testing with production data |
| Patched and compliant with stateful enforcement | Provides infrastructure that is both dynamic and auditably compliant |
| High availability and disaster recovery should be inherent to the operational platform | Failure of a server, service, or site should not take the desk out of the market |

## Solution Elements

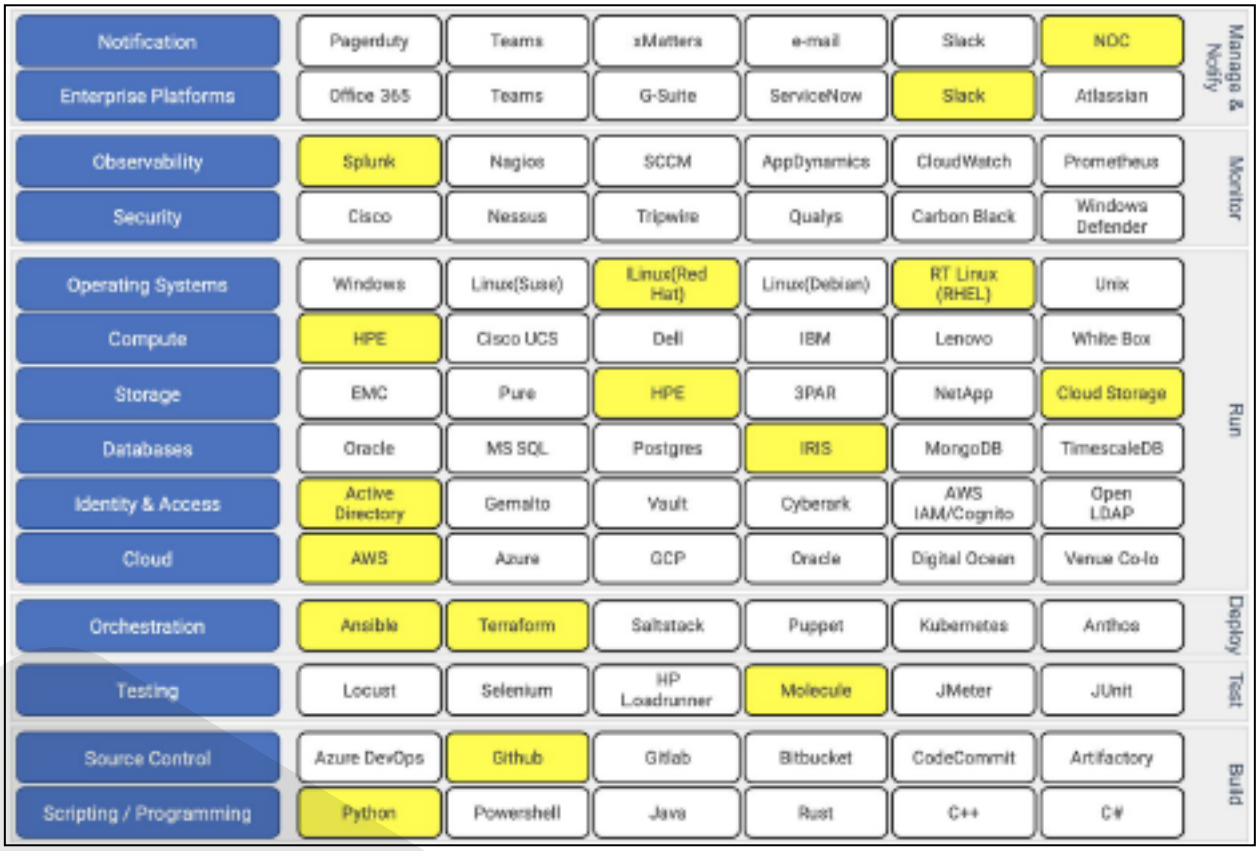
- Integrated CI/CD pipeline built to support the languages, platform, and architecture selected by the development team. Critical components include:
  - GIt-based source code management
  - Automated unit testing on check-ins
  - Integrated static code analysis
  - Scheduled, triggered and on-demand integration/regression testing
  - Automated security scanning - both static and of deployed pre-prod instances
  - ITIL change process automated in the implementation of the pipeline

- Fully Software Defined Infrastructure
  - Encompasses network, firewall rules, server configuration, operating system, layered products, security, and monitoring
  - Buildable on physical or virtual hardware, in a co-lo or on IaaS cloud
  - Under full source code management, making infrastructure repeatable, inspectable, and auditable
  - Applicable to physical servers, firewalls, and switches, to virtualized environments (VMware, KVM, ...), and both public and private (Aprenda, Open Stack...) clouds
  - ITIL change process automated in the implementation of the infrastructure deployment process

- Operations Automation
  - Operations orchestration technology centralizes control of all components with a stateful, scripted execution capability
  - Create all new non-prod and production instances with an automated process — cloud, virtual, and physical
  - Every server or container starts fully loaded and fully patched with all security, monitoring, and admin packages installed and tested — and is in the CMDB
  - Access controls, group privileges, auditing, firewall configurations are all set as part of the build and adjustable centrally with changes applied to any logical group of assets
  - Make every server image easily replaceable with a clean build at any time
  - Automation components log every action, providing real-time monitoring, anomaly detection, and corrective feedback loops for infrastructure and application changes
  - Choose to patch server images or simply redeploy fresh infrastructure with all application components
  - Orchestrate changes to allow for continuous availability

## OpStack Baseline Operations Stack for Trading Systems

The tools below have been proven and tested in complex, regulated infrastructure environments. They are a combination of commercially supported open source and best of breed. Substitutions to provide desired capabilities or to conform to existing corporate standards is always possible and often desirable. Capacity modeling, an important capability for trading systems are currently all proprietary but should be part of the solution stack.




The accompanying Stack Builder is solution tailored, but it includes many of the technologies and tools used or incorporated into recent OpStack Engagements to meet the needs of our clients.Not all of the tools in any category are required to automate infrastructure for any single environment or application, OpStack will select the smallest set that fully meets the requirements of the application suite being hosted.

## Co-Lo versus Cloud or Co-Lo with Cloud Deployment

The approach proposed allows for development, test, and production environments to be deployed on in the cloud or in the desired trading-advanted co-lo (e.g. the Equinix NY4 data center.) There are cost, complexity, and opportunity trade-offs to be made in the proximity of hosting of trading engines to the ECNs (NYSE Bonds, Bloomberg, Instinet, …). Only as the trading strategies and algorithms are developed and the engine deployed to execute them within the trading application will it be definitive as to the advantage or disadvantage of relative differences in latency. The number of firms that have chosen to co-locate with trading venues in order to effectively pursue low-latency and high-frequency trading strategies argues for opting to minimize latency.

The OpStack approach to defining, deploying, and operating the trading infrastructure is equally applicable to any mix of cloud and co-lo deployments, the table below lays out some of those options.

## Hosting Options

| Option | Example | Pro | Con |
|---|---|---|---|
| Cloud VIrtual | AWS EC2 Virtual | Low-startup costs<br><br>Expandable in minutes<br><br>Can use AWS Direct Connect | Least control of consistent performance<br><br>Network is shared, firewalls are virtual and shared<br><br>Highest latency option - WAN, not fibre-LAN, latencies |
| Cloud Physical | AWS EC2 Physical | Higher cost, all OpEx<br><br>Expandable in days<br><br>Can use AWS Direct Connect<br><br>Allows option for Real-TIme Linux OS | Network is shared, firewalls are virtual and shared<br><br>High latency - WAN, not fibre-LAN, latencies |

| | | | |
|---|---|---|---|
| NY4 Co-Lo Virtual | HPE Gen 10 Servers<br><br>Arista Switches<br><br>Palo Alto Firewalls | Sub-millisecond latency to ECNs<br><br>Fully independent network and firewalls<br><br>Ephemeral deployment of resources across<br><br>Ability to distribute load across servers<br><br>CapEx option | Capacity upgrades require hardware purchase and installs<br><br>Additional hypervisor latency<br><br>Requires standard Linux or WIndows OS, real-time option irrelevant<br><br>Network and firewalls dedicated |
| NY4 Co-Lo Physical | HPE Gen 10 Servers<br><br>Arista Switches<br><br>Palo Alto Firewalls | Sub-millisecond latency to ECNs<br><br>Real-Time Linux option for fully predictable performance<br><br>Lowest latency option with COTS hardware and software<br><br>Fully independent resources | Capacity upgrades require hardware purchase and installs<br><br>Workloads partitionable at the server level (though virtual and physical can be mixed in cage) |

The optimal solution is to mix and match the above options as the application is developed. The strength of the design is that infrastructure is fungible with little or no change required to the trading applications.

Development and functional QA environments can be provisioned in cloud or on-prem virtual environments that are software identical to production. As the number of these environments will be variable with the development and testing cycle (and multiple parallel development streams are common amongst trading shops) they can be created, spun up, spun down, and released at any time with nominal effort. Algorithm testing against real-time data or capture/playback facilities would benefit from environments identical to production -- whether cloud or co-lo, but on high performance hardware and production latencies.

A hybrid-cloud approach, using:

- AWS for development, monitoring, logging, and post-trade analytics
- with all low-latency trading components running in containers on dedicated hardware co-located with the ECNs

is OpStack's recommendation as the optimal solution in 2020's technical and trading environment that would be both market competitive and cost-effective.

## Representative Co-Lo Configuration

This hardware configuration provides a representative hardware and system software configuration for the Co-Lo Physical and CoLo Virtual options. The equipment proposed is state-of-the-practice and the configurations based on the assumptions laid out by our client prior to the final definition and creation of the platform. The pricing provided was quoted by the manufacturers and is not reflective of competitive procurement practices or negotiated discounts.

## Design Approach

The infrastructure is designed to:

1. Provide high-performance execution engines to be housed in a single rack within the Equinix NY-4 data center.
2. Redundant UPS-protected power distribution is provided in rack as standard at Equinix.
3. High-performance, high reliability, industry-standard components have been selected. No GPU co-processors have been specified, but they could easily be added to the strawman design if advantageous.
4. The design assumes connectivity to both AWS (Direct Connect) infrastructure and to dedicated fibre drops to ECNs co-located at NY-4 or any other latency-advantaged data center.
5. All hardware is redundant and configured so that the loss of one component of any type or of any one connection would not disrupt operations.
6. AWS or another cloud provider will be used to house administrative systems, provide a storage target for backups, and host development as well as monitoring and control infrastructure -- only the low-latency components need premium hardware and minimum latency location.
7. Our assumption is that the ECN connections will terminate at the Arista switches (for lowest latency) rather than at the Palo Alto firewalls. This is a design decision to revisit as the application topology is finalized.

8. The network design provides for external connections at the standard speeds coming from those providers with a 40 gigabit internal network to provide for low latency, DMA-capable, communications between all elements of the trading systems.
9. There is expansion space both in the rack and in the networking components (including ports on the servers) to support the addition of additional servers, specialized GPU compute hardware, or a SAN storage array.

## Hardware Components

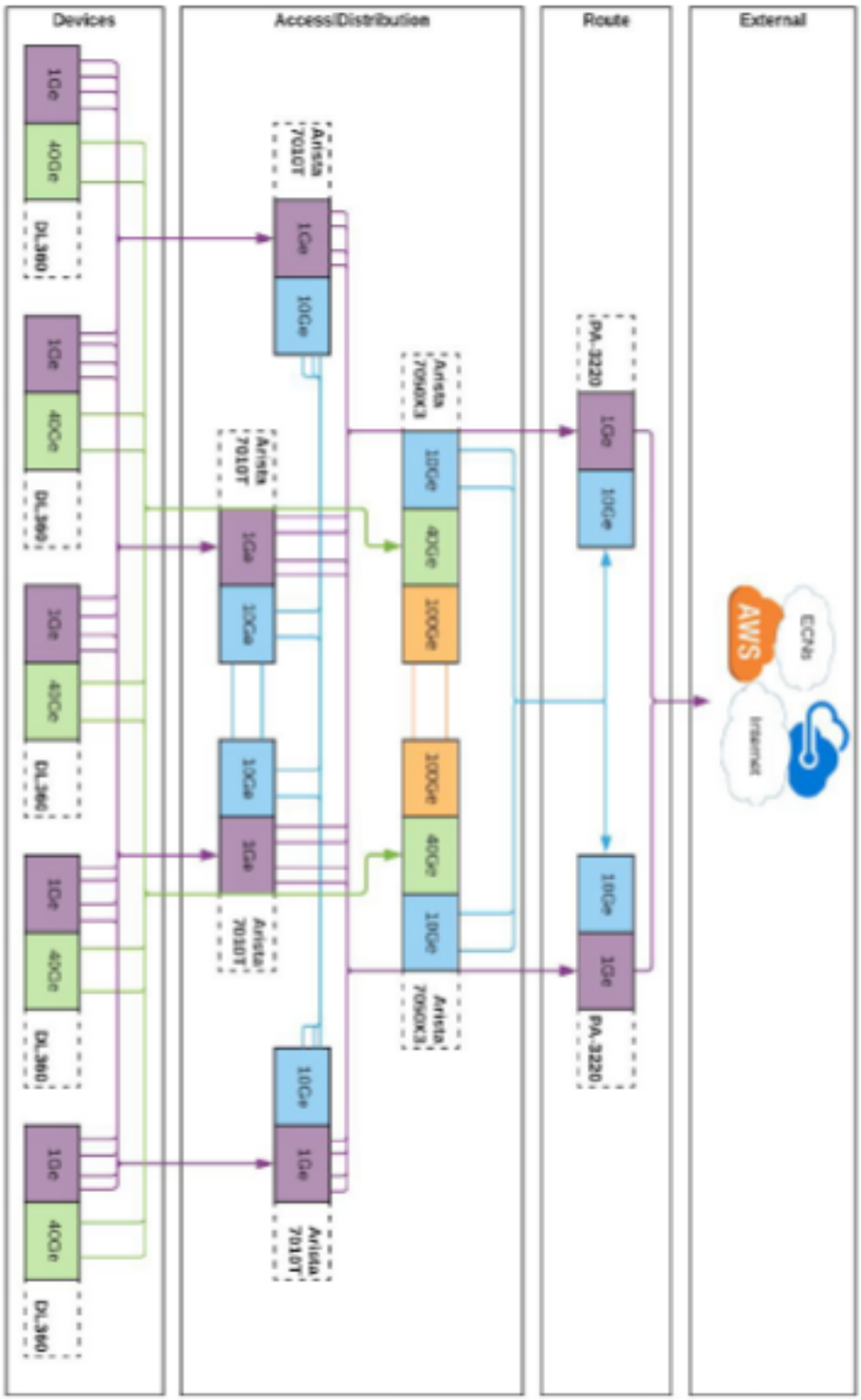
| Role | Selection | Notes |
|---|---|---|
| Server Hardware | 5 x HPE DL360 Gen10 with 512 GB of memory, and 2.5 TB of useable internal RAID-5 SSD storage | With the small server count in the initial request, we've specified high-quality, high-reliability server hardware with excellent out-of-band management capability |
| Network Switching | 4 x Arista 7010 and 2 x 7050 switches | Fully software configurable 40 gigabit fiber optic core distribution switching and gigabit copper switching in front of the firewalls. |
| Firewalls | 2 x Palo Alto PA-3220 | Full software configurable threat prevention firewalls with anti-malware and URL filtering |
| Total Hardware | | |

## System Software Components

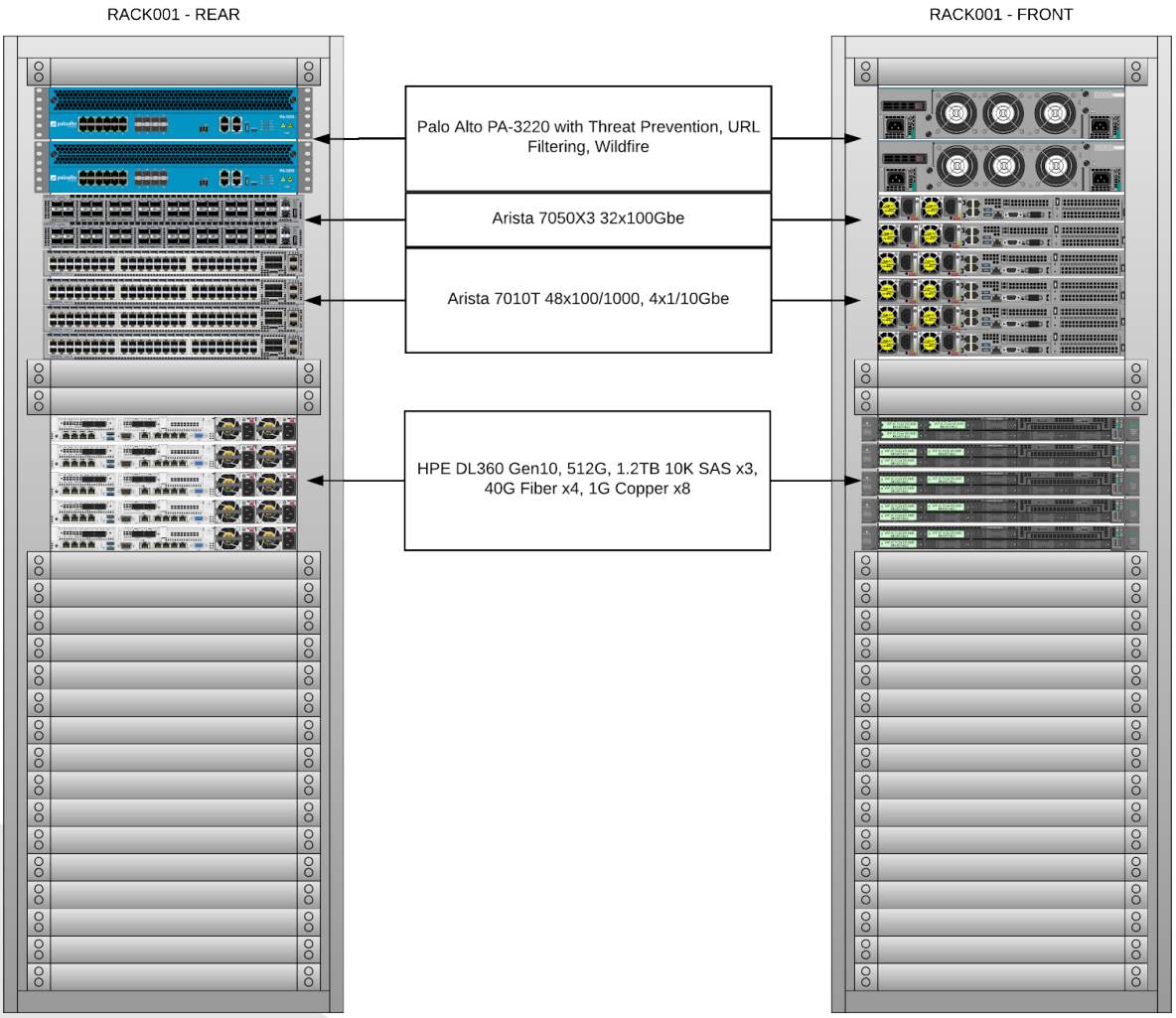
In addition to the relevant operations configuration and management components selected from the Operations Stack, the licensing and support for a container-based low-latency trading application include the following two items. Both Real-Time Linux and Kubernetes are open source products and there is a very wide range of list-prices for support and commercial licensing. The choice of distribution for these products will have little to no impact on performance and little impact on the overall cost of environment setup.

| Role | Selection | Notes |
|---|---|---|
| Operating System | Real-Time Linux | We presume that the real-time kernel will be preferred in order to provide predictable performance on time-sensitive compute functions. |
| Container Orchestration | Kubernetes | Includes Docker containers and a full Kubernetes orchestration capability with management console. |

## Logical Connectivity Network Diagram



## Rack Elevation Diagram



## OpStack Services

### Project Phases - Months not Years

| Phase | Objective | Key Milestones | Effort |
|---|---|---|---|
| Dev and Test Environment Scripted Builds in AWS | Enable developers to actively work on the pre-production product | • Basic environments deployed in AWS on demand tied into CI/CD tooling<br>• Supporting infrastructure deployed<br>  ○ Satellite<br>  ○ Gitlab<br>  ○ Ansible | 480 hours (4 weeks, 3 named resources) |
| Physical Environment Build with Scripted OS, Network, and Security Configuration | Deploy and configure physical compute, network, and security infrastructure for production environment | • All hardware installed and configured<br>• Container platform deployed<br>• Environment securely connected to required tools and services | 360 hours (3 weeks, 3 named resources) |
| Production Application Deployment Integration and Automation | Deploy and configure production application components, monitored | • Integrate production environment into CI/CD environment<br>• Deploy and configure Splunk integration<br>• Integrate CI/CD deployment process into Change Management | 600 hours (5 weeks, 3 named resources) |
| Ongoing Support | Ensure operational excellence | • Monitor infrastructure<br>• Break/Fix<br>• Code releases<br>• Maintenance<br>• Patching | SLA - 10 minutes to callback 8-6 ET |

## Assumptions and Notes

- Splunk and other monitoring hardware has not been specified. The design forwards both log records and SNMP traps to monitoring software in the AWS cloud, Splunk/SignalFX is state-of-the-practice but OpStack has had excellent results with Elastic, and Prometheus/Graphana as well.
- All systems will be built locked-down to minimize security threats and will be scanned for relevant compliance testing. No additional security monitoring or mitigation software (e.g Carbon Black) or malware detection and prevention (e.g. Symantec Endpoint Protection) has been specified, but can be added at the customer's discretion.
- No DR (Disaster Recovery) environment has been specified or priced. The OpStack SDI implementation can be extended to define such an environment should it be desired. The key additional information needed to estimate those costs would be RTO (Recovery Time Objective), RPO (Recovery point Objective), and desired location of capacity based on the DR facilities of the connected trading venues.
- Additional server capacity for the production co-lo compute complex can be extrapolated from the server hardware and system software prices for the initial five servers.
  Any deployment costs (other than onsite time to rack, stack, wire, and light) are nominal.
- A current best practice is to have sufficient capacity to run two instances of production with the second available for A/B testing of new code, trading strategies, system software, or firmware. Should A/B capacity be desired it can be specified once the actual application hardware requirements are better understood.
- Trading simulation (tick capture and replay) is a capability that is likely to be desirable and may require additional CPU and storage capacity.

For more information or to continue the discussion with the team at OpStack,

**Call or email us at:**

+1.416.806.3711 or john@opstack.com | +1.646.641.2973 or evan@opstack.com