Monitoring Systems Comparison 2020

Status	IN PROGRESS
Stakeholders	Matt Rowley David Jeffers Robert Bellante Michael Crossin Michael O'Neill Eric Parton Garth Dubin Maureen Seiler
Outcome	
Due date	19 Dec 2019
Owner	Steven M. Bambling

Goals

- Determine a replacement for the current Sensu 1.x version

Legend

Cost: The products yearly cost for licensing and/or support

Support: The products Community and/or Enterprise support options

Platform: The platform (Coding Language) on which the tool was developed/written.

Access Controls: Features user-level security, allowing an administrator to prevent access to certain parts of the product on a per-user or per-role basis

Distributed Architecture: The product is able to leverage more than one server to distribute the load of network monitoring across multiple site /silos

Latest Release: The release date and version of the latest published **stable** release

Simplistic Architecture: Maintainability / Ease of upgrading for all required components that make up the whole of the system (database, message bus, web front end, platform/main app)

Data Storage Method: Main method used to store the network data it monitors

Client Server Communication: Client/App and method for communication between the host being monitored and server instance

UI Server Communication: Method for communication between the UI(web) and product server

Logical Grouping: Supports arranging the hosts or devices it monitors into user-defined groups.

Configuration Management: What configuration methods are available, most frequently used and up to date.

Functional UI: The ease of use, modularity and key feature of the most frequently used Web UI

Single pane of glass: Product provide a single pane of glass Web UI/Dashboard that details status of all monitored assets in a distributed architecture.

Host/Service Dependencies: Product provides an intelligent/easy method for both host (parent/child) and service dependencies.

Including but not limited to dependency linking with keep-alive checks and network devices to prevent page fatigue from a flood of alerts

Parent/Child Reachability Logic: Logic to determine host/check states for down vs unreachable, by performing parallel checks of the parents and children of the affected host.

Check Aggregation: Aggregates make it possible to treat the results of multiple disparate check results, executed across multiple disparate system as a single result

Time/Day Based Alerting: Function to suppress/filter events based on timestamp for a host, hostgroup, service, or service group. AKA business hour paging

Business Logic: Products ability to provide intelligent grouping / dependencies to monitor hierarchical business processes

Notification Integrations: Products level of integration with additional notification platforms, notably Slack and Pager Duty

Cross-Platform Clients: Operating Systems/Platforms that the product provides clients or an alternative method for monitoring

API Available: Product provides a robust API to interface either/or with the WebUI or each distributed server instance

CLI Available: Product provides a robust CLI to interface either/or with the WebUI or each distributed server instance

Inventory: Keeps a record of hardware and/or software inventory for the hosts and devices it monitors

Comparison Table

	Nagios (core)	Sensu Go	Zabbix	Icinga	Prometheus	NetData	Monit	CheckMK	LogicMonitor	SignalFX	DataDog
Cost	Free / $2,495/yr	16,000/yr	Free	Free	Free	Free		Standard: ~3,000/yr (18,000 service/ ~600 hosts) Managed Services: ~3,000/yr (18,000 service/ ~600 hosts)	Need to request a quote	Standard: 9,000/yr (600 hosts) Enterprise: 15,000/yr (600 hosts)	Pro: 9,000 /yr (600 hosts) Enterprise: 3,800/yr (600 hosts)
Support	Community	Enterprise	Community	Community	Community	Community	Community /Enterprise	Enterprise (**+ 7399/yr**)	Enterprise	Enterprise	Enterprise
Platform	C	Go	C	C++	Go	C			N/A	Agent: Go	N/A
Access Controls	Yes	Yes	Yes	Yes	Maybe	No	Maybe	Yes (Managed Services Edition)		Enterprise Only	Enterprise Only ??
Distributed Architecture	Yes	Yes	Yes	Yes*	Yes	Yes	No*	Yes	Yes*		
Latest Release	2019-08-29	2019-12-18	2019-12-19	2019-10-24	2019-12-25	2019-11-27	2019-07-06	2019-11-12			
Simplistic Architecture	Nagios + Web Server + Nagios Fusion	Sensu Go Backend (embedded etcd)	Zabbix + Web Server + IDO Database	Icinga Master /Satellite + Icinga Web + IDO Database	Prometheus (Includes TSDB)	NETDATA	M/Monit + IDO Database	CheckMX \ Apache Web Server (Included /Optional)	LogicMonitor Collector		
Data Storage Method	Flat-File	etcd	MySQL /PGSQL	MySQL /PGSQL	Flat-File / Volume	Database Engine	SQLite / MySQL / PGSQL	Flat-File			
Client Server Communication	NRPE /NCPA + SSL	WebSockets (wss)	Certificate / PSK	TLS certificates, secure connection handling	HTTPS	HTTPS	HTTPS	HTTP / TCP-6556	SNMP		
UI Server Communication	HTTPS	HTTPS	PDO (PHP)	IDO SSL	SSL via Proxy	HTTPS	HTTPS	Livestatus			
Logical Grouping	Yes	Yes	Yes	Yes	No	No	Yes	Yes			
Configuration Management	Yes (Community)	Yes	Yes	Yes	Yes	Yes (Community)	Yes (Community)	Yes (Community)			
Functional UI	Yes	Yes	Yes	Yes	Maybe w/ Grafana	Yes	Maybe w/ Grafana	Yes			
Single pane of glass	Nagios Fusion $	Yes (Enterprise only)	Yes*	Yes*	Maybe w/ Grafana	Maybe*	Maybe w/ Grafana	Yes*			
Host/Service Dependencies	Yes	No	Yes	Yes	Maybe	No	Yes	Maybe*			
Parent/Child Reachability Logic	Yes	No	Maybe	Maybe	No	No	No	Yes			
Business Logic	Plugin	Maybe		Plugin	Maybe	No	No	Yes			
Check Aggregation	Maybe	Plugin	Yes	Maybe	Yes	No	No	Maybe			
Time/Day Based Alerting	Yes	Yes	Yes	Yes	Maybe	No	No	Yes			
Notification Integrations	Slack / PagerDuty	Slack / PagerDuty	Slack / PagerDuty	Slack / PagerDuty	Slack / PagerDuty	Slack / PagerDuty	Slack / PagerDuty * (custom)	Slack / PagerDuty			

Cross-Platform Clients	Linux \ OpenBSD \ FreeBSD	Linux \ Docker \ FreeBSD \ OpenBSD*	Linux \ OpenBSD \ FreeBSD	Linux \ OpenBSD \ FreeBSD	Linux \ OpenBSD \ FreeBSD ***	Linux / FreeBSD	Linux / OpenBSD / FreeBSD	Linux / OpenBSD/ FreeBSD /	Linux / FreeBSD	Linux	Linux \ FreeBSD*
API Available	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes			
CLI Available	Plugin	Yes	Community	Yes	No	No	Yes	Yes			
Inventory	No	No	Yes	No	No	No	No	Yes			

Nagios

Notes

- Nagios (core) requires Nagios Fusion (Enterprise) to provide the desired distributed architecture, single pane of glass dashboard and many of the additional grouping features.
  - This cost $2,495/yr per instance
- The API seems limited but is available for additional custom tooling
- HostGroup,ServiceGroup and Service definitions can help to simplify configuration by 'subscribing' nodes to checks
- Check aggregation looks to be possible with the business process add-on and/or using the cluster_check plugin

Pros

- Provides correct host reachability logic along with predictive checks to prevent a flood of check alerts reducing alert fatigue
- The business process intelligence (BPI) plugin allows us you to set more complex dependencies to determine groups states. Example ARIN Online \ Whois Stacks
- Nagios Fusion provides AD authentication

Cons

- OpenBSD NRPE port is old 2.15 vs 3.2.1 though there have been some updates as recently as (2018-01-12) http://openports.se/net/nagios/nrpe
- Configuration management with Ansible is limited there is no provided module to configure Nagios or Checks.
- The Ansible Playbooks will likely need to be home-grown from whole cloth

---

Sensu-Go

Notes

- Pub/Sub model could allow for easier management and configuration for both new nodes and check.
  This is because the configuration would be at a single source and not need to be pushed out to every client
- With an etcd backend to create a clustered backend instance you would need a minimum of 3 nodes and for quorum would need to scale in odd numbers (3/5/7)
- From talk within the Slack channel (no official document or word from Devs) the keepalive checks will not suppress subscription checks
- Check dependancies may not be necessary. Because of the pub/sub model if a parent is down the agent
  would not be obtain subscribed messages from the queue to execute, this assumes a TTL is not being used
- Business process logic may
- **Events in Sensu Go are ALWAYS handled by default, whereas events in Sensu 1.x are only handled on failure conditions**
- Business process logic looks like it could be accomplished though filters, though these are all written in JavaScript now
  and require assets to be created to be deployed.
- The Sensu agent was able to be compiled on a fresh install of OpenBSD 6.6

```
pkg_add git
pkg_add go
git clone https://github.com/sensu/sensu-go
cd sensu-go
go build ./cmd/sensu-agent
go build ./cmd/sensuctl
```

Pros

- Up to date and what looks like thorough CM tools for Puppet/Ansible (Community supported)
- Robust CLI for management/configuration, this can make it easier to configure monitoring for immutable infrastructure
- Subscriptions help to simplify configuration applying them to groups of system rather then one-to-one mapping

Cons

- No OpenBSD agent support (at the time of this comparison)
- No check dependancies for either host/service
- No parent/child reachability logic

Zabbix

Notes

- Zabbix does not support host dependencies directly, but a similar effect can be accomplished via trigger dependencies
- For service dependencies service.adddependancies can be used.
- Host reachability logic looks like its could be set with using Unavailable and Unreachable parameters for a host setting the network devices to a lower value
- Time based alerting can be accomplished via time periods
- Zabbix share provides a nice location for shared plugins and checks, etc
- Interesting Grafana integration
- https://tech.virtualminds.de/a-bit-of-everything/zabbix-is-awesome/
- Logical group can be done with Services and host groups

Pros

Cons

- The use of Zabbix proxy servers are required in order to achieve a distributed setup

---

Icinga

Notes

- The single pane of glass only works because all data is feed into a master
- Its unclear how the parent/child reachability logic functions and if parallel checks of the parents and children are executed upon state change
- Check aggregation looks like its might be possible by using Cluster checks
- Business hour alerting looks like it might be possible by using time dependent thresholds
- Notification periods can be assigned for business hour paging
- Check aggregation looks to be possible with the business process add-on and/or using something like a cluster check

Pros

- Icinga Exchange provides a nice location of shared plugins and checks such as the Grafana plugin to display graphs within Icinga Web

Cons

- Its distributed monitoring uses a Master/Satellite architecture which would require all data to be stored on a shared database at a single location
  This would require allowing external sites to send data into the core, would need to see about setting up some SSH tunnels similar to ElasticStack

---

Prometheus

Notes

- This use a different model then any of our legacy system where metrics are gathered from system and the evaluated based on rule expressions for alerting
- Grafana would become a more important part of the monitoring infrastructure and would need to be setup for high availability
- AlertManager doesn't look very robust and may have a steep learning curve, Grafana alerting maybe useful as an alternative but may require additional tooling for silences/pauses of alerts
- Dashboard will need to use Grafana (Prometheus AlertManager, Alerts - Overview)
- Awesome Prometheus Alerts

Pros

- Will combine the function of Graphite allowing us to manage less systems

Cons

- The node_exporter is only for Linux, so other methods would need to be used for OpenBSD such as the collectd_exporter or Prometheus plugin
  I can't see how to configure TLS for some exporters which would also require a proxy setup
- TLS is not currently implemented but on the road map, but can be done via Proxy (Nginx) for server endpoints
- Host/Service relationships look to be complex using inhibitions
- Time of day based alerting seems complex and error prone (GH)

NetData

Notes

- There is no functionality for access controls, simple basic auth can be setup with a web server (Nginx, Apache) used for authentication
- Metics can be shipped to a rental NetData server with the use of NetData proxies then all the alerts/alarms will be triggered from a 'central' server - All this data can also be shipped an external source such as prometheus / graphite
- Kali Linux article on NetData

- Bind Stats
- Interactive Infographic

Pros

- Minimal client setup
- Will combine the function of Graphite/Collectd allowing us to manage less systems

Cons

- No OpenBSD support, looks like there is an open issue for this work but it seems to have stalled
- Alerting rules don't allow for parent/child relationships to prevent alert storms
- No time/day based alerting for business hour paging
- No check aggregation ability

Monit

Notes

- Monit is different from traditional monitoring systems, where each client monitors itself and report alarm vs sending to a 'master' node to process the data similar to Netdata.

Pros

- monit agent is very nice and powerful for getting current status and auto remediation as needed
- The syntax for writing check is very pragmatic and easily human readable.

Cons

- In order to get a distributed architecture you will need to use m/monit to help aggregate alerts/status
- There is no concept of warning, an alert is either pass or fail
- You will need to configure all the checks on the node themselves but the alerts may or may not need to be configured on the m/monit node
- If alerts on configured on each node, some kind of proxy will need to be used in order to push alarms to PD and Slack
- Limited local check dependancies
- No option for business logic or parent child dependancies

CheckMX

Notes

- Inventory looks pretty basic, but provides an option to update agents

Pros

- Supports centralized management of all the distributed monitoring nodes via **distributed WATO**
- CheckMK is using Livestatus to poll the other monitoring servers for status of remote hosts

Cons

- Both the OpenBSD and FreeBSD agents look old but are still maintained, latest commit was ~3 weeks ago as of the time of this comparison. Will need to demo the amount of functionality they provide, as documentation is sparse on the topics
- There is no documentation of mention of secure communication between the CheckMK-agent and the monitoring host. Some googling suggests using stunnel for secure comms
- Configuration Management (Ansible/Others) looks very sparse and incomplete
- I can find no clear documentation on setting up service dependancies but it maybe possible via business intelligence

Action items

- [ ]