DA-ITSI-TELEGRAF-OS Documentation

Release 1

Guilhem Marchand

Apr 04, 2019

Contents

1 Overview: 3
- 1.1 About . . . 3
- 1.2 Compatibility . . . 3
- 1.3 Known Issues . . . 4
- 1.4 Support . . . 4
- 1.5 Download . . . 4

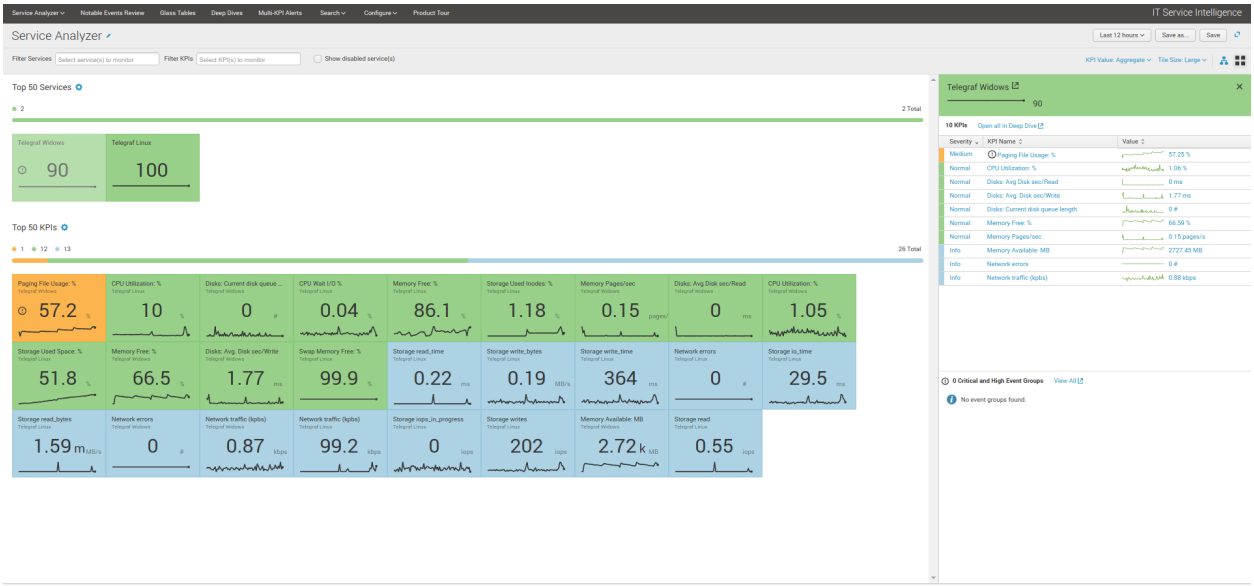
2 Deployment and configuration: 7
- 2.1 Deployment & Upgrades . . . 7
- 2.2 Telegraf metrics ingestion . . . 7
- 2.3 Entities discovery . . . 18
- 2.4 Services creation . . . 19
- 2.5 Telegraf Health views . . . 22

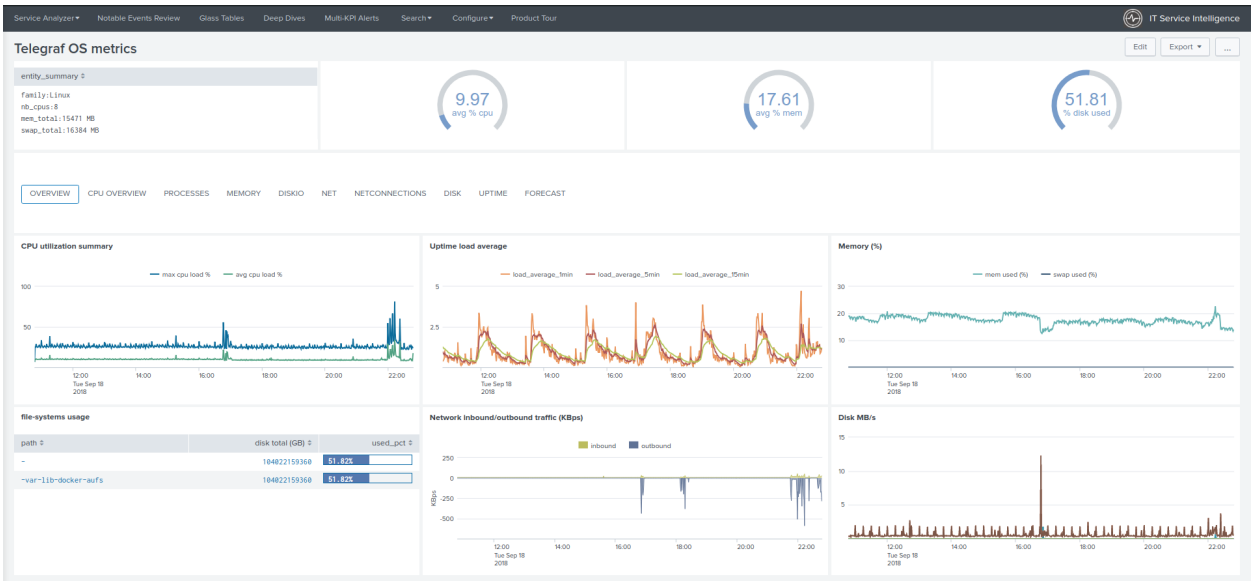
3 Troubleshoot: 35
- 3.1 Troubleshoot & FAQ . . . 35

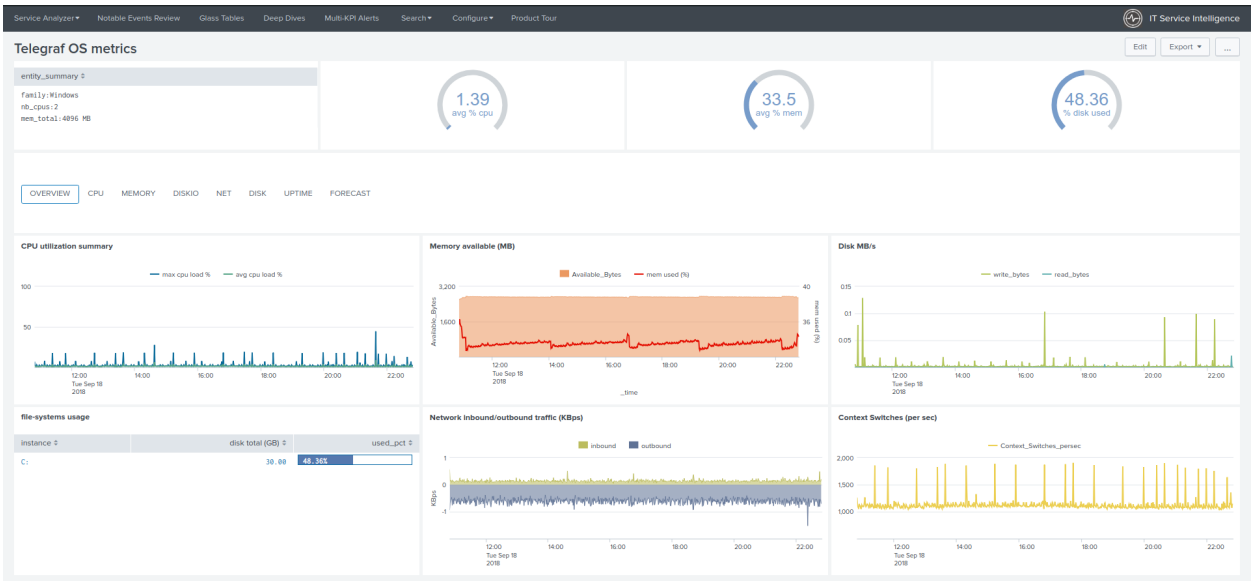
4 Versioniong and build history: 37
- 4.1 Release notes . . . 37

Telegraf is the famous and powerful plugin-driven server agent for collecting and reporting metrics by Influxdata.

The ITSI module for Telegraf OS brings the power of Telegraf to ITSI and provides builtin entities discovery, services templates and KPI base searches, and rich entity health views.






Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.39
avg % cpu
33.5
avg % mem
48.36
% disk used
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
CPU utilization summary
max cpu load %
avg cpu load %
Memory available (MB)
Available_Bytes
mem used (%)
Disk MB/s
write_bytes
read_bytes
file-systems usage
instance
disk total (GB)
used_pct
C:
30.00
48.36%
Network inbound/outbound traffic (KBps)
inbound
outbound
Context Switches (per sec)
Context_Switches_persec

CHAPTER 1

Overview:

1.1 About

- Author: Guilhem Marchand
- First release published in September 2018
- Purposes:

The ITSI module for Telegraf OS leverages the rich and powerful plugin driver agent from Influxdata:

https://github.com/influxdata/telegraf

Telegraf is the famous and powerful plugin-driven server agent for collecting and reporting metrics by Influxdata

The ITSI module leverages Telegraf for Splunk and provides:

- Builtin entities discovery
- Services templates and KPI base searches
- Rich entity health views to manage Operating System metrics ingested in the Splunk metric store

1.2 Compatibility

1.2.1 Splunk compatibility

Especially because the ITSI module relies on, and leverages the Splunk metric store, Splunk 7.0.x or or higher is required.

1.2.2 ITSI compatibility

The ITSI module has been tested and qualified against reasonably fresh versions of ITSI, recommended version is 3.1.0 and higher, previous versions may work as well although it has not and will not be tested.

1.2.3 OS metrics collection compatibility

Telegraf supports various operating systems and process architectures including any version of Linux and Windows.

For more information:

- https://portal.influxdata.com/downloads

1.3 Known Issues

There are no known issues at the moment.

1.4 Support

The ITSI module for Telegraf OS is community supported.

To get support, use of one the following options:

1.4.1 Splunk Answers

Open a question in Splunk answers for the application:

- https://answers.splunk.com/app/questions/4194.html

1.4.2 Splunk community slack

Contact me on Splunk community slack, or even better, ask the community !

- https://splunk-usergroups.slack.com

1.4.3 Open a issue in Git

To report an issue, request a feature change or improvement, please open an issue in Github:

- https://github.com/guilhemmarchand/DA-ITSI-TELEGRAF-OS/issues

1.4.4 Email support

- guilhem.marchand@gmail.com

However, previous options are far betters, and will give you all the chances to get a quick support from the community of fellow Splunkers.

1.5 Download

1.5.1 ITSI Module for Telegraf OS

The ITSI Module for Telegraf OS can be downloaded from:

Splunk base

- https://splunkbase.splunk.com/app/4194

GitHub

- https://github.com/guilhemmarchand/DA-ITSI-TELEGRAF-OS

1.5.2 Technology Addon for Telegraf metrics

The Technology Addon for Telegraf metrics can be downloaded from:

notes: The TA is only required for indexing time parsing if you index data in Graphite output format with Splunk TCP inputs or Kafka

Starting version 1.8 of Telegraf, data can be sent directly to Splunk HTTP Event Collector from Telegraf hosts

Splunk base

- https://splunkbase.splunk.com/app/4193

GitHub

- https://github.com/guilhemmarchand/TA-influxdata-telegraf

CHAPTER 2

Deployment and configuration:

2.1 Deployment & Upgrades

2.1.1 Initial deployment

The deployment of the ITSI module for Telegraf OS is extremely straight forward.

Deploy the ITSI module using one of the following options:

- Using the application manager in Splunk Web (Settings / Manages apps)
- Extracting the content of the tgz archive in the "apps" directory of Splunk
- For SHC configurations (Search Head Cluster), extract the tgz content in the SHC deployer and publish the SHC bundle

2.1.2 Upgrades

Upgrading the ITSI module is pretty much the same operation, use one of the techniques that matches your conditions / requirements.

2.2 Telegraf metrics ingestion

Implementing Telegraf and sending its metrics in Splunk is simple, and efficient.

It is not the purpose of this documentation to expose every piece of the installation and configuration of Telegraf or Splunk.

2.2.1 Telegraf installation and configuration

Telegraf standard installation (standalone and independant process)

The installation of Telegraf is really straightforward, consult:

- https://github.com/influxdata/telegraf

If you wish to deploy Telegraf as a Splunk TA application instead, consult the next step.

Telegraf deployment as Splunk application deployed by Splunk (TA)

For Linux OS only (this does not work on Windows), you can publish Telegraf through a Splunk application that you push to your clients using a Splunk deployment server.

This means that you can create a custom Technology Addon (TA) that contains both the Telegraf binary and the telegraf.conf configuraton files.

This method has several advantages:

- If you are a Splunk customer already, you may have the Splunk Universal Forwarder deployed on your servers, you will NOT need to deploy an additional collector independently from Splunk
- You get benefit from the Splunk centralisation and deploy massively Telegraf from Splunk
- You can maintain and upgrade Telegraf just as you do usually in Splunk, all from your Splunk Deployment Server. (DS)

If you wish to check for a read to use example, see:

- https://github.com/guilhemmarchand/TA-telegraf-amd64

To achieve this, you need to:

- Create a package in your Splunk Deployment Server, let's call it "TA-telegraf-linux-amd64", if you have more than this architecture to maintain, just reproduce the same steps for other processor architectures. (arm, etc)

```
$SPLUNK_HOME/etc/deployment-server/TA-telegraf-linux-amd64
                                                    /bin
                                                    /local/telegraf.conf
                                                    /metadata/default.meta
```

- The "bin" directory hosts the Telegraf binary
- The "local" directory hosts the telegraf.conf configuration file
- The "metadata" directory is a standard directory that should contain a default.meta which in the context of the TA will contain:

default.meta

```
# Application-level permissions
[]
owner = admin
access = read : [ * ], write : [ admin ]
export = system
```

- Download the last Telegraf version for your processor architecture (here amd64), and extract its contain in the "bin" directory, you will get:

```
bin/telegraf/etc
bin/telegraf/usr
bin/telegraf/var
```

- Telegraf provides an "init.sh" script that we will use to manage the state of the Telegraf process:

```
bin/telegraf/usr/lib/telegraf/scripts/init.sh
```

- Copy this "init.sh" script to the root of the "bin" directory:

```
cp -p bin/telegraf/usr/lib/telegraf/scripts/init.sh bin/
```

- Achieve some minor modifications related to patch customizations, basically:

Change:

```
USER=telegraf
GROUP=telegraf
```

By:

```
USER=$(whoami)
GROUP=$(id -g -n)
```

Notes: the configuration above will automatically use the owner of the Splunk processes

After this section, add: (adapt if TA name differs)

```
APP=TA-telegraf-amd64
```

Change:

```
STDERR=/var/log/telegraf/telegraf.log
```

By:

```
STDERR=$SPLUNK_HOME/etc/apps/$APP/bin/telegraf/var/log/telegraf/telegraf.log
```

Change:

```
daemon=/usr/bin/telegraf
```

By:

```
daemon=$SPLUNK_HOME/etc/apps/$APP/bin/telegraf/usr/bin/telegraf
```

Change:

```
config=/etc/telegraf/telegraf.conf
confdir=/etc/telegraf/telegraf.d
```

By:

```
config=$SPLUNK_HOME/etc/apps/$APP/local/telegraf.conf
confdir=$SPLUNK_HOME/etc/apps/$APP/bin/telegraf/etc/telegraf/telegraf.d
```

Change:

```
pidfile=/var/run/telegraf/telegraf.pid
```

By:

```
pidfile=$SPLUNK_HOME/var/run/telegraf/telegraf.pid
```

Finally, create a very simple local/inputs.conf configuration file:

local/inputs.conf

```
# start telegraf at Splunk start, and restart if Splunk is restarted. (which allows␣
↪upgrading easily Telegraf binaries shipped with the TA package)
[script://./bin/init.sh restart]
disabled = false
interval = -1
```

Et voila ! Deploy this TA to your Linux host and start receiving metrics in Splunk.

If you wish to check for a read to use example, see:

- https://github.com/guilhemmarchand/TA-telegraf-amd64

Upgrades:

To upgrade Telegraf binary to a new version, simply extract the new tgz release in the "bin" directory, and reload your Splunk Deployment server.

Splunk will automatically restart the Telegraf process after Splunk startup.

Telegraf minimal configuration

A minimal configuration to monitor Operating System metrics:

- https://docs.influxdata.com/chronograf/latest/guides/using-precreated-dashboards/#system

The output configuration depends on the deployment you choose to use to ingest metrics in Splunk, consult the next sections.

Example of a minimal telegraf.conf configuration that monitors Operating System metrics for Linux:

```
# Read metrics about cpu usage
[[inputs.cpu]]
  ## Whether to report per-cpu stats or not
  percpu = true
  ## Whether to report total system cpu stats or not
  totalcpu = true
  ## If true, collect raw CPU time metrics.
  collect_cpu_time = false
  ## If true, compute and report the sum of all non-idle CPU states.
  report_active = false

# Read metrics about disk usage by mount point
[[inputs.disk]]
  ## By default stats will be gathered for all mount points.
  ## Set mount_points will restrict the stats to only the specified mount points.
  # mount_points = ["/"]

  ## Ignore mount points by filesystem type.
  ignore_fs = ["tmpfs", "devtmpfs", "devfs"]

# Read metrics about disk IO by device
[[inputs.diskio]]
  ## By default, telegraf will gather stats for all devices including
```

(continues on next page)

(continued from previous page)

```
  ## disk partitions.
  ## Setting devices will restrict the stats to the specified devices.
  # devices = ["sda", "sdb", "vd*"]
  ## Uncomment the following line if you need disk serial numbers.
  # skip_serial_number = false
  #
  ## On systems which support it, device metadata can be added in the form of
  ## tags.
  ## Currently only Linux is supported via udev properties. You can view
  ## available properties for a device by running:
  ## 'udevadm info -q property -n /dev/sda'
  # device_tags = ["ID_FS_TYPE", "ID_FS_USAGE"]
  #
  ## Using the same metadata source as device_tags, you can also customize the
  ## name of the device via templates.
  ## The 'name_templates' parameter is a list of templates to try and apply to
  ## the device. The template may contain variables in the form of '$PROPERTY' or
  ## '${PROPERTY}'. The first template which does not contain any variables not
  ## present for the device is used as the device name tag.
  ## The typical use case is for LVM volumes, to get the VG/LV name instead of
  ## the near-meaningless DM-0 name.
  # name_templates = ["$ID_FS_LABEL","$DM_VG_NAME/$DM_LV_NAME"]

# Get kernel statistics from /proc/stat
[[inputs.kernel]]
  # no configuration


# Read metrics about memory usage
[[inputs.mem]]
  # no configuration


# Get the number of processes and group them by status
[[inputs.processes]]
  # no configuration


# Read metrics about swap memory usage
[[inputs.swap]]
  # no configuration


# Read metrics about system load & uptime
[[inputs.system]]
  # no configuration

# # Read metrics about network interface usage
[[inputs.net]]
#   ## By default, telegraf gathers stats from any up interface (excluding loopback)
#   ## Setting interfaces will tell it to gather these explicit interfaces,
#   ## regardless of status.
#   ##
#   # interfaces = ["eth0"]
#   ##
#   ## On linux systems telegraf also collects protocol stats.
#   ## Setting ignore_protocol_stats to true will skip reporting of protocol metrics.
```

(continues on next page)

(continued from previous page)

```
#   ##
#   # ignore_protocol_stats = false
#   ##

# # Read TCP metrics such as established, time wait and sockets counts.
[[inputs.netstat]]
#   # no configuration

# # Monitor process cpu and memory usage
[[inputs.procstat]]
#   ## PID file to monitor process
#   pid_file = "/var/run/nginx.pid"
#   ## executable name (ie, pgrep <exe>)
#   # exe = "nginx"
#   ## pattern as argument for pgrep (ie, pgrep -f <pattern>)
#   # pattern = "nginx"
#   ## user as argument for pgrep (ie, pgrep -u <user>)
    user = "guilhem"
#   ## Systemd unit name
#   # systemd_unit = "nginx.service"
#   ## CGroup name or path
#   # cgroup = "systemd/system.slice/nginx.service"
#
#   ## override for process_name
#   ## This is optional; default is sourced from /proc/<pid>/status
#   # process_name = "bar"
#
#   ## Field name prefix
#   # prefix = ""
#
#   ## Add PID as a tag instead of a field; useful to differentiate between
#   ## processes whose tags are otherwise the same.  Can create a large number
#   ## of series, use judiciously.
#   # pid_tag = false
#
#   ## Method to use when finding process IDs.  Can be one of 'pgrep', or
#   ## 'native'.  The pgrep finder calls the pgrep executable in the PATH while
#   ## the native finder performs the search directly in a manor dependent on the
#   ## platform.  Default is 'pgrep'
#   # pid_finder = "pgrep"
```

Windows additional configuration (mem inputs)

For Windows memory management, the default win_mem inputs does not retrieve some of the metrics we need.

You need to activate the memory inputs. (which on Windows uses WMI collection):

```
[[inputs.mem]]
# no configuration
```

Windows Active Directory Domain Controller

Follow instructions for "Active Directory Domain Controller":

- https://github.com/influxdata/telegraf/tree/master/plugins/inputs/win_perf_counters#active-directory-domain-controller

Windows DNS server

Follow instructions for "DNS Server + Domain Controllers":

- https://github.com/influxdata/telegraf/tree/master/plugins/inputs/win_perf_counters#dns-server–domain-controllers

Windows DFS server

For DFS Namespace, follow instructions for "DFS Namespace + Domain Controllers":

- https://github.com/influxdata/telegraf/tree/master/plugins/inputs/win_perf_counters#dfs-namespace–domain-controllers

For DFS Replication, follow instructions for "DFS Replication + Domain Controllers":

- https://github.com/influxdata/telegraf/tree/master/plugins/inputs/win_perf_counters#dfs-replication–domain-controllers

Microsoft IIS / ASP.NET

For IIS and ASP.NET, follow instructions for: IIS / ASP.NET

- https://github.com/influxdata/telegraf/tree/master/plugins/inputs/win_perf_counters#iis–aspnet

Linux processes monitoring (procstat)

In the linux views, the processes usage (both CPU and Memory) rely on the procstat inputs, which requires additional configuration depending on your context.

As for an example, the following configuration monitors all the processes owned by the "splunk" unix user:

```
[[inputs.procstat]]
#   ## PID file to monitor process
#   pid_file = "/var/run/nginx.pid"
#   ## executable name (ie, pgrep <exe>)
#   # exe = "nginx"
#   ## pattern as argument for pgrep (ie, pgrep -f <pattern>)
#   # pattern = "nginx"
#   ## user as argument for pgrep (ie, pgrep -u <user>)
    user = "splunk"
```

2.2.2 HTTP Events Collector (HEC)

Splunk deployment with HEC (available with Telegraf starting version 1.8)

Telegraf agents –> HTTP over SSL –> Splunk HEC inputs

With Telegraf starting version 1.8, you can send metrics directly from Telegraf to HTTP Events Collector using the excellent serializer leveraging the http Telegraf output.

This is extremely simple, scalable and reliable.

Example of an HEC input definition:

Splunk inputs.conf:

```
[http://Telegraf]
disabled = 0
index = telegraf
indexes = telegraf
token = c386d4c8-8b50-4178-be76-508dca2f19e2
```

Telegraf configuration:

The Telegraf configuration is really simple and relies on defining your ouput:

Example:

```
[[outputs.http]]
   ## URL is the address to send metrics to
   url = "https://mysplunk.domain.com:8088/services/collector"
    ## Timeout for HTTP message
   # timeout = "5s"
    ## Optional TLS Config
   # tls_ca = "/etc/telegraf/ca.pem"
   # tls_cert = "/etc/telegraf/cert.pem"
   # tls_key = "/etc/telegraf/key.pem"
   ## Use TLS but skip chain & host verification
   insecure_skip_verify = true
    ## Data format to output.
   ## Each data format has it's own unique set of configuration options, read
   ## more about them here:
   ## https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_OUTPUT.md
   data_format = "splunkmetric"
    ## Provides time, index, source overrides for the HEC
   splunkmetric_hec_routing = true
    ## Additional HTTP headers
    [outputs.http.headers]
   # Should be set manually to "application/json" for json data_format
      Content-Type = "application/json"
      Authorization = "Splunk c386d4c8-8b50-4178-be76-508dca2f19e2"
      X-Splunk-Request-Channel = "c386d4c8-8b50-4178-be76-508dca2f19e2"
```

Push this configuration to your Telegraf agents, et voila.

Check data availability in Splunk:

```
| mcatalog values(_dims) as dimensions values(metric_name) as metric_name where␣
↪index=telegraf metric_name=*
```

2.2.3 TCP / TCP-SSL Inputs

Splunk deployment with TCP inputs.

This deployment requires additional indexing time parsing configuration:

- https://github.com/guilhemmarchand/TA-influxdata-telegraf

The deployment is very simple and can be described as:

Telegraf agents –> TCP or TCP over SSL –> Splunk TCP inputs

In addition and to provide resiliency, it is fairly simple to add a load balancer in front of Splunk, such that you service continues to ingest metrics depending on Splunk components availability. (HAProxy, Nginx, F5, whatsoever...)

The data output format used by Telegraf agents is the "Graphite" format with tag support enable. This is simple, beautiful, accurate and allows the management of any number of dimensions.

Example of a tcp input definition:

Splunk inputs.conf:

```
[tcp://2003]
connection_host = dns
index = telegraf
sourcetype = tcp:telegraf:graphite
```

Telegraf configuration:

The Telegraf configuration is really simple and relies on defining your ouput:

Example:

```
[[outputs.graphite]]
  ## TCP endpoint for your graphite instance.
  ## If multiple endpoints are configured, the output will be load balanced.
  ## Only one of the endpoints will be written to with each iteration.
  servers = ["mysplunk.domain.com:2003"]
  ## Prefix metrics name
  prefix = ""
  ## Graphite output template
  ## see https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_OUTPUT.
↪md
  # template = "host.tags.measurement.field"

  ## Enable Graphite tags support
  graphite_tag_support = true

  ## timeout in seconds for the write connection to graphite
  timeout = 2
```

Push this configuration to your Telegraf agents, et voila.

Check data availability in Splunk::

> mcatalog values(_dims) as dimensions values(metric_name) as metric_name where index=telegraf metric_name=*

2.2.4 SPLUNK file monitoring Ingestion

Splunk deployment with Splunk file monitoring.

This deployment requires additional indexing time parsing configuration:

- https://github.com/guilhemmarchand/TA-influxdata-telegraf

Telegraf has a "file" output plugin that allows writing metrics to a local file on the file-system, don't say more that is much more than enough to Splunk it.

The Telegraf configuration is really simple and relies on defining your ouput:

Example:

```
[[outputs.file]]
  ## Files to write to, "stdout" is a specially handled file.
  files = ["/tmp/metrics.out"]

  ## Data format to output.
  ## Each data format has its own unique set of configuration options, read
  ## more about them here:
  ## https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_OUTPUT.md
  data_format = "graphite"
  graphite_tag_support = true
```

Notes: this is a simplistic example, in real condition do not forget to manage the file rotation using a simple logrorate configuration for Linux, and relevant solution for other OS.

Splunk file input configuration:

I cannot say more, this is simple, very simple. Add the following configuration to any inputs.conf configuration file of your choice:

Example:

```
[monitor::/tmp/metrics.out]
disabled = false
index = telegraf
sourcetype = file:telegraf:graphite
```

Apply this simple input.conf, if you deploy thought the Splunk deployment server ensure splunkd is configured to restart in your serverclass configuration.

Et voila, Splunk ingests the metrics continously and metrics are forwarded to the indexing layer using your Splunk infrastructure, be on-premise, private or Splunk Cloud.

2.2.5 KAFKA Ingestion

Splunk deployment with Kafka.

This deployment requires additional indexing time parsing configuration:

- https://github.com/guilhemmarchand/TA-influxdata-telegraf

If you are using Kafka, or consider using it, producing Telegraf metrics to Kafka makes a lot of sense.

First, Telegraf has a native output plugin that produces to a Kafka topic, Telegraf will send the metrics directly to one or more Kafka brokers providing scaling and resiliency.

Then, Splunk becomes one consumer of the metrics using the scalable and resilient Kafka connect infrastructure and the Splunk Kafka connect sink connector. By using Kafka as the mainstream for your metrics, you preserve the possibility of having multiple technologies consuming these data in addition with Splunk, while implementing a massively scalable and resilient environment.

On the final step that streams data to Splunk, the Kafka sink connector for Splunk sends data to Splunk HEC, which makes it resilient, scalable and eligible to all Splunk on-premise or Splunk Cloud platforms easily.

The deployment with Kafka can be described the following way:

Telegraf agents –> Kafka brokers <– Kafka connect running Splunk sink connector –> Splunk HTTP Event Collector (HEC)

Configuring Kafka connect:

- The Kafka connect properties needs to use the "String" converter, the following example start Kafka connect with the relevant configuration:

connect-distributed.properties:

```
# These are defaults. This file just demonstrates how to override some settings.
bootstrap.servers=kafka-1:9092,kafka-2:9092,kafka-3:9092
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.storage.StringConverter
key.converter.schemas.enable=false
value.converter.schemas.enable=false
internal.key.converter=org.apache.kafka.connect.json.JsonConverter
internal.value.converter=org.apache.kafka.connect.json.JsonConverter
internal.key.converter.schemas.enable=false
internal.value.converter.schemas.enable=false
# Flush much faster (10s) than normal, which is useful for testing/debugging
offset.flush.interval.ms=10000
plugin.path=/etc/kafka-connect/jars
group.id=kafka-connect-splunk-hec-sink
config.storage.topic=__kafka-connect-splunk-task-configs
config.storage.replication.factor=3
offset.storage.topic=__kafka-connect-splunk-offsets
offset.storage.replication.factor=3
offset.storage.partitions=25
status.storage.topic=__kafka-connect-splunk-statuses
status.storage.replication.factor=3
status.storage.partitions=5
# These are provided to inform the user about the presence of the REST host and port␣
↪configs
# Hostname & Port for the REST API to listen on. If this is set, it will bind to the␣
↪interface used to listen to requests.
#rest.host.name=
rest.port=8082
# The Hostname & Port that will be given out to other workers to connect to i.e. URLs␣
↪that are routable from other servers.
# rest.advertised.host.name=kafka-connect-1
```

Apply the following command against Kafka connect running the Splunk Kafka sink connector: (https://splunkbase.splunk.com/app/3862)

- replace <ip address or host> by the IP address or the host name of the Kafla connect node, if you run the command locally, simply use localhost
- replace the port if required (default is 8082)
- replace the HEC token
- replace the HEC destination
- adapt any other configuration item up to your needs

Achieve the following command:

```
curl localhost:8082/connectors -X POST -H "Content-Type: application/json" -d '{
"name": "kafka-connect-telegraf",
"config": {
"connector.class": "com.splunk.kafka.connect.SplunkSinkConnector",
"tasks.max": "3",
"topics":"telegraf",
"splunk.indexes": "telegraf",
```

(continues on next page)

(continued from previous page)

```
"splunk.sourcetypes": "kafka:telegraf:graphite",
"splunk.hec.uri": "https://myhecinput.splunk.com:8088",
"splunk.hec.token": "fd96ffb6-fb3e-43aa-9e8b-de911356443f",
"splunk.hec.raw": "true",
"splunk.hec.ack.enabled": "true",
"splunk.hec.ack.poll.interval": "10",
"splunk.hec.ack.poll.threads": "2",
"splunk.hec.ssl.validate.certs": "false",
"splunk.hec.http.keepalive": "true",
"splunk.hec.max.http.connection.per.channel": "4",
"splunk.hec.total.channels": "8",
"splunk.hec.max.batch.size": "1000000",
"splunk.hec.threads": "2",
"splunk.hec.event.timeout": "300",
"splunk.hec.socket.timeout": "120",
"splunk.hec.track.data": "true"
}
}'
```

telegraf output configuration:

Configure your Telegraf agents to send data directly to the Kafka broker in graphite format with tag support:

```
[[outputs.kafka]]
  ## URLs of kafka brokers
  brokers = ["kafka-1:19092","kafka-2:29092","kafka-3:39092"]
  ## Kafka topic for producer messages
  topic = "telegraf"
  data_format = "graphite"
  graphite_tag_support = true
```

Et voila. Congratulations, you have built a massively scalable, distributable, open and resilient metric collection infrastructure.

Check data availability in Splunk:

```
| mcatalog values(_dims) as dimensions values(metric_name) as metric_name where␣
↪index=telegraf metric_name=*
```

2.3 Entities discovery

The ITSI entities discovery is a fully automated process that will discover and properly configure your entities in ITSI depending on the data availability in Splunk.

2.3.1 Entities automatic import

In a nutshell, the following reports are automatically scheduled:

- DA-ITSI-TELEGRAF-OS-Inventory_Search_linux
- DA-ITSI-TELEGRAF-OS-Inventory_Search_windows

When entities are discovered, entities will be added automatically with the following informational field:

- itsi_role=telegraf_host

This main information, in addition with the family of the Operating System, are used to automatically provide the relevant entity health page view in ITSI.

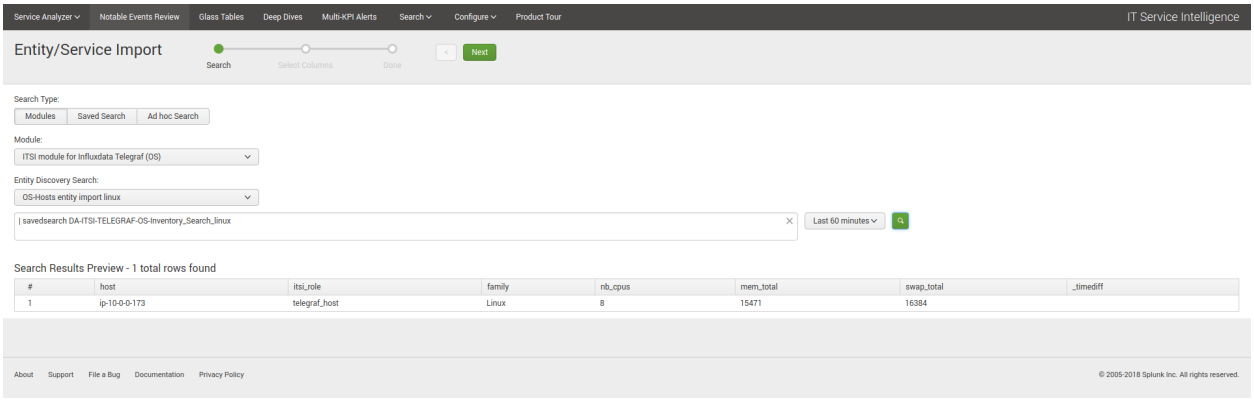
2.3.2 Manual entities import

It is possible to manually import the entities in ITSI, and use the searches above:

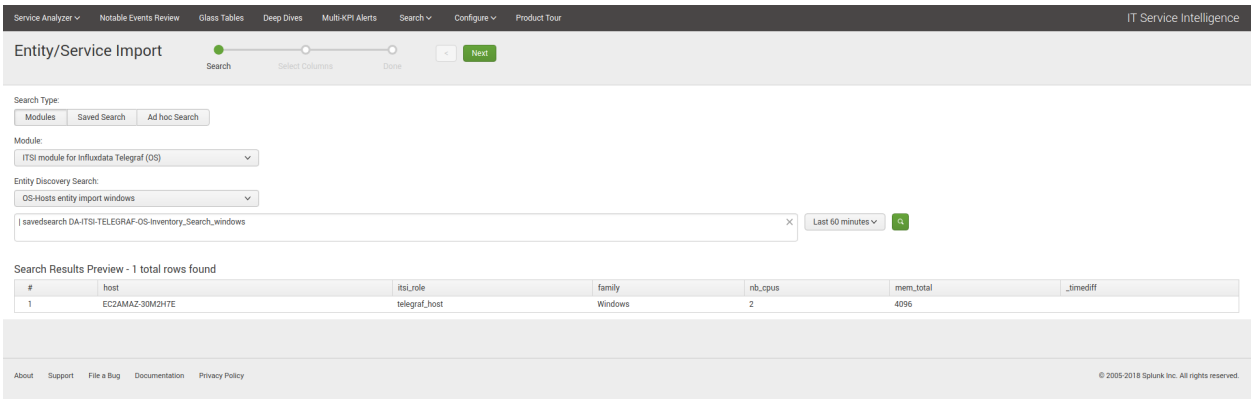
Configure / Entities / New Entity / Import from Search

Then select the module name, and depending on your needs select the relevant search.

Linux hosts discovery:



Windows hosts discovery:



2.4 Services creation

The ITSI module for Telegraf OS provides builtin services templates, relying on several base KPIs retrieving data from the metric store.

At the moment, the following templates are provides:

- Linux OS
- Windows OS

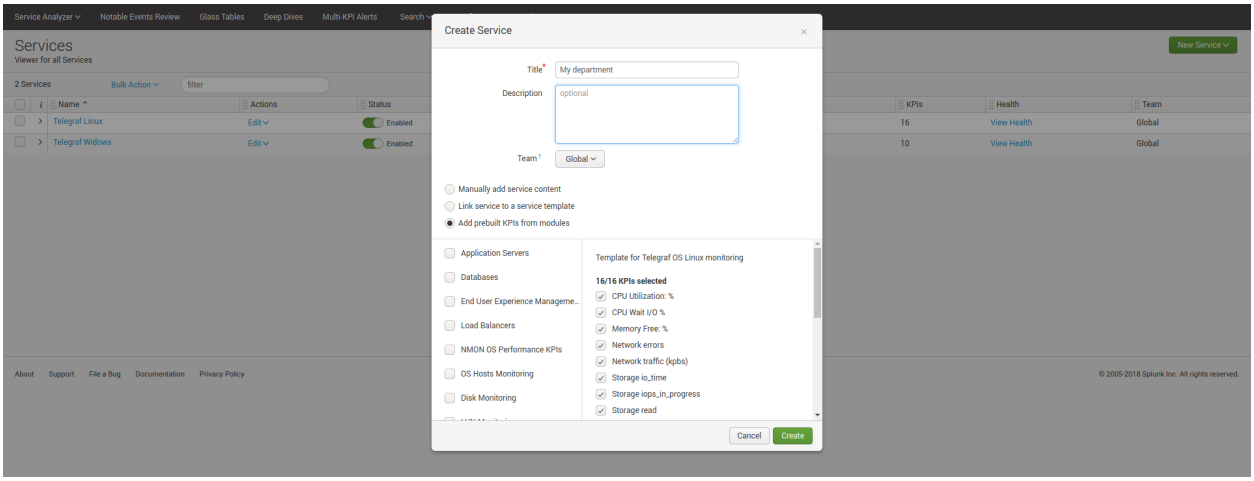
2.4.1 Creating a new service

Creating a new service and importing the KPIs definition is very easy:

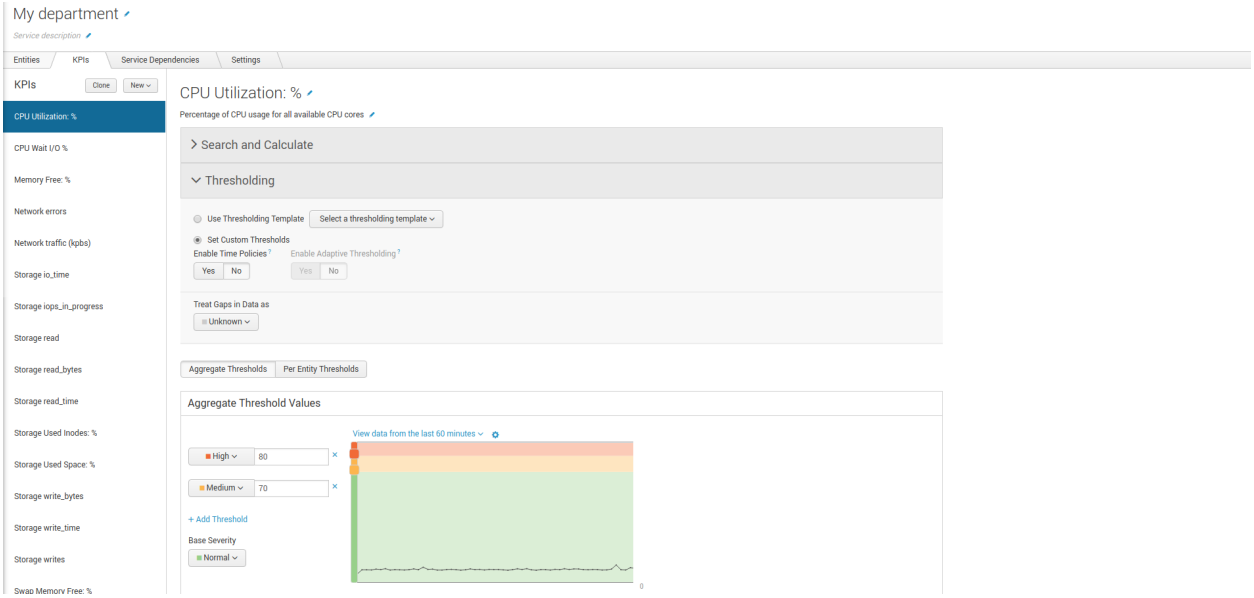
- Configure / Services / New Service / New Service

Then fill the required information, select the option "Add prebuilt KPIs from modules", and depending on your needs:

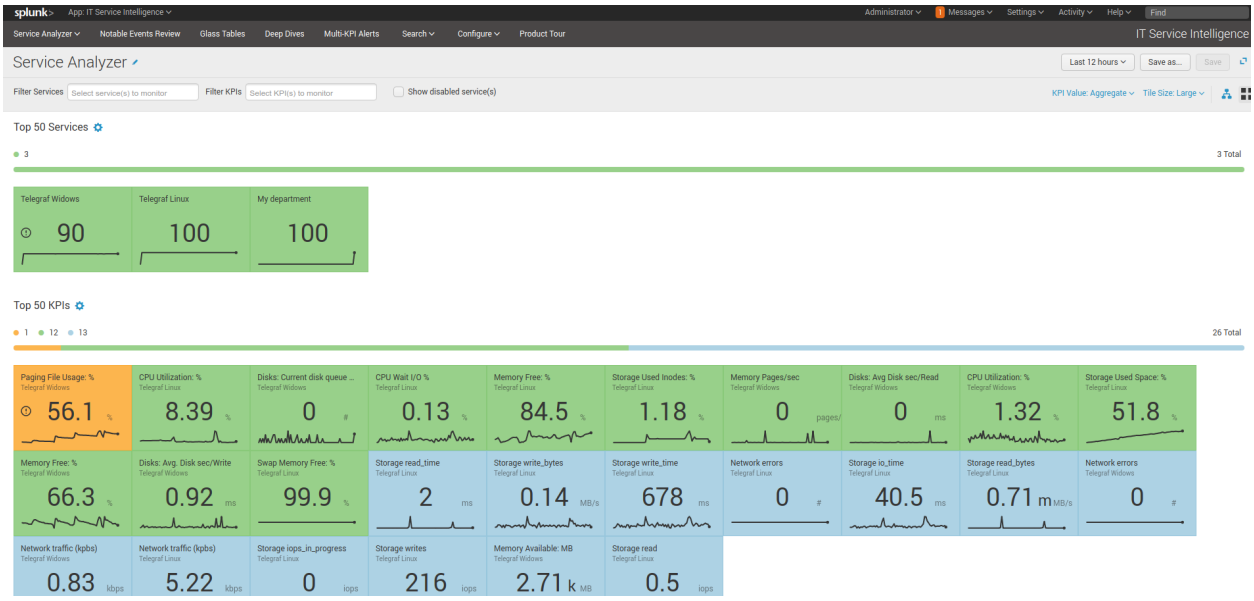
- Telegraf OS Monitoring (Linux)
- Telegraf OS Monitoring (Windows)



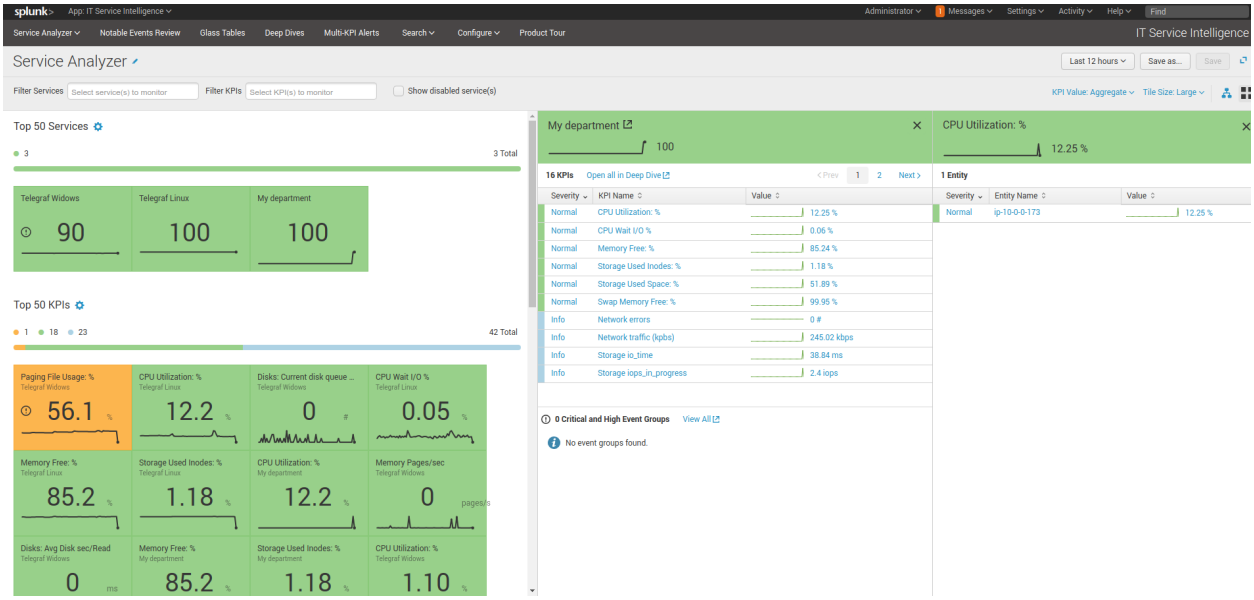
Once the service is created, achieve any modification you need such as the hosts to be matched, customize threshold values if required, and finally activate your service.



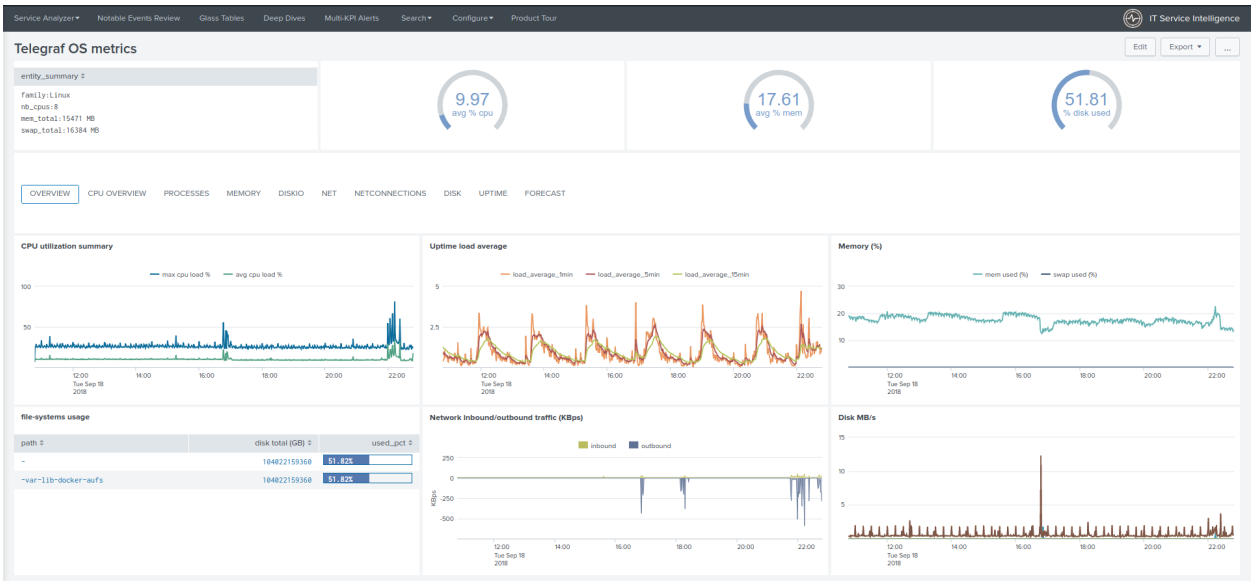
After the service was created, ITSI will start to generate metrics and you service will be visible in the Service Analyser:



Developing the entities part of the service provides access to the entity health view:



Where you can access to the Telegraf OS health view:

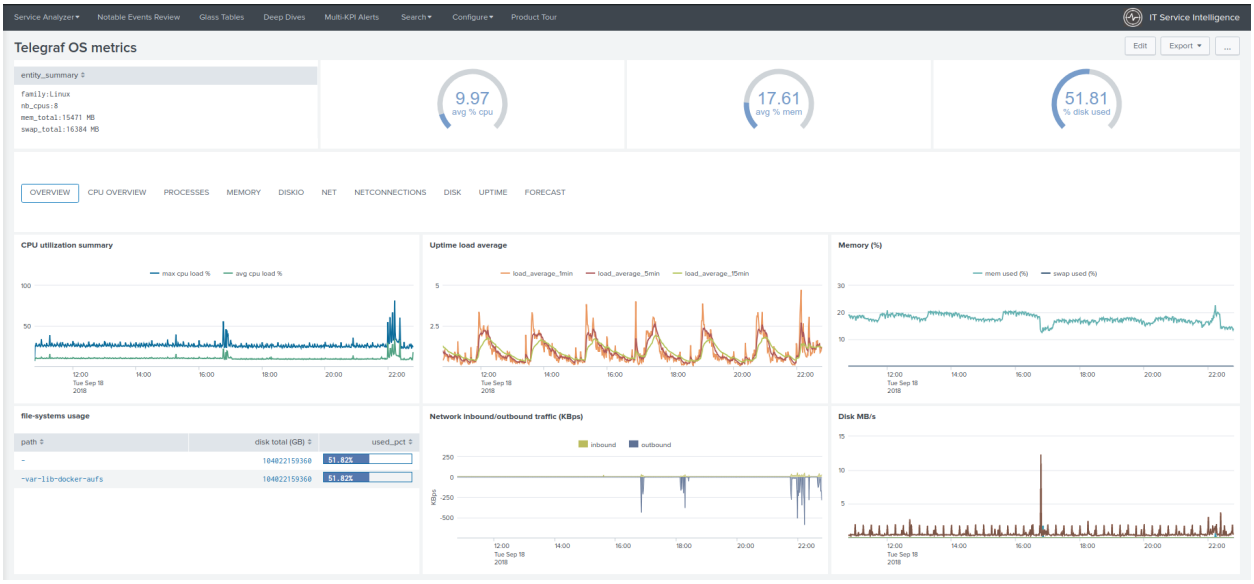


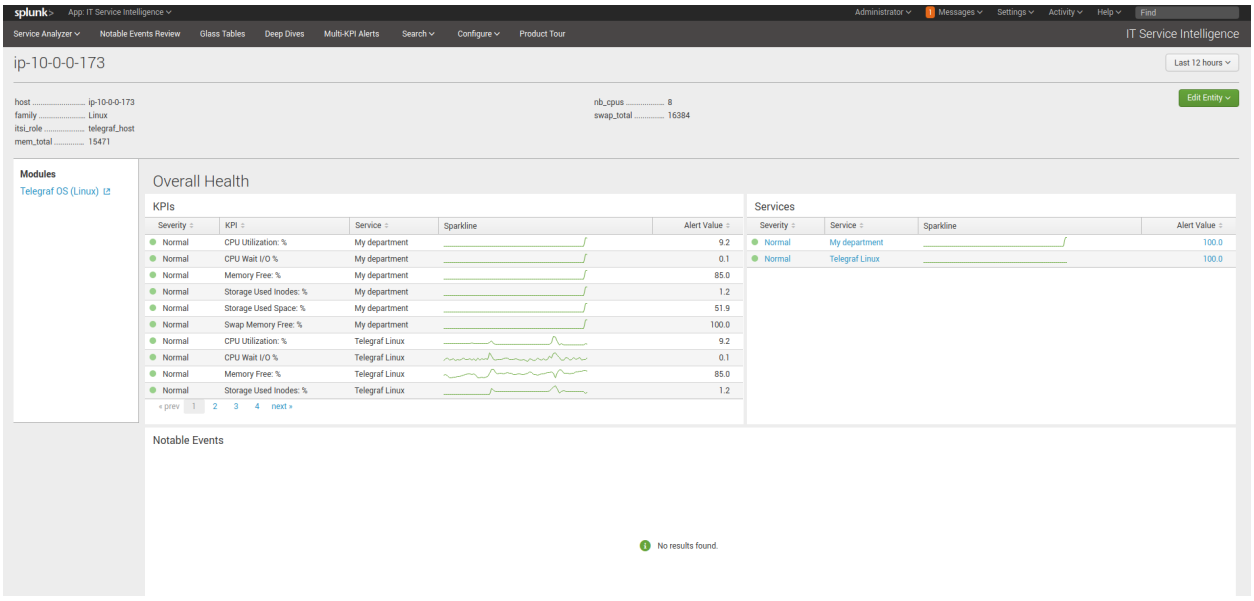
2.5 Telegraf Health views

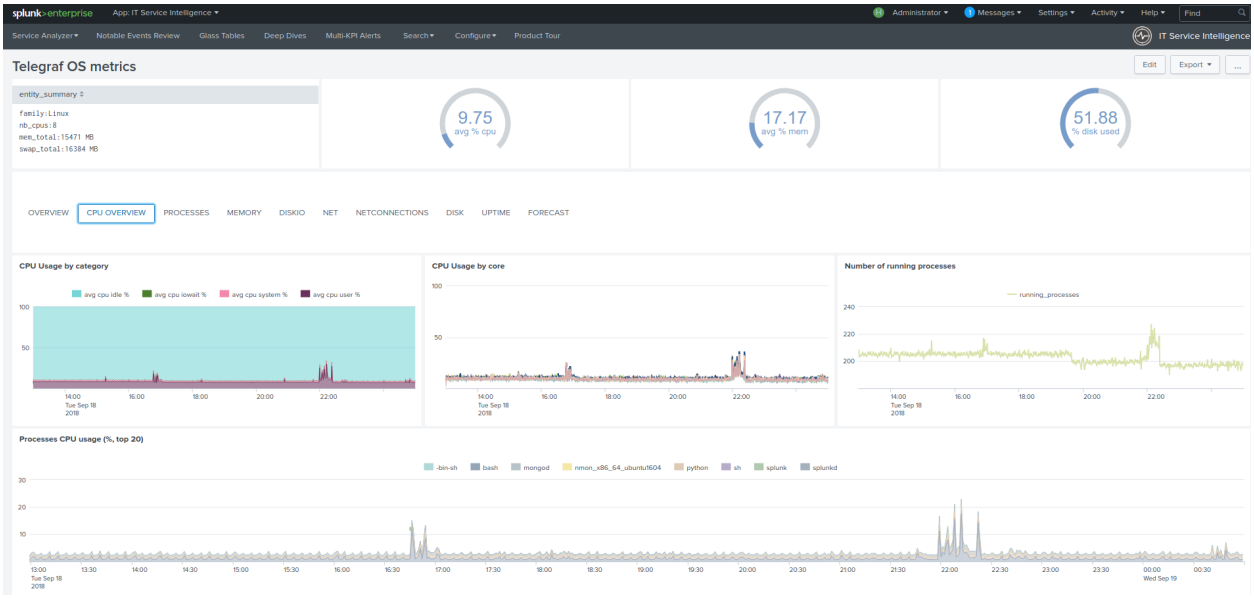
Through builtin ITSI deepdive links, you can automatically and easily access to the rich Telegraf OS views.

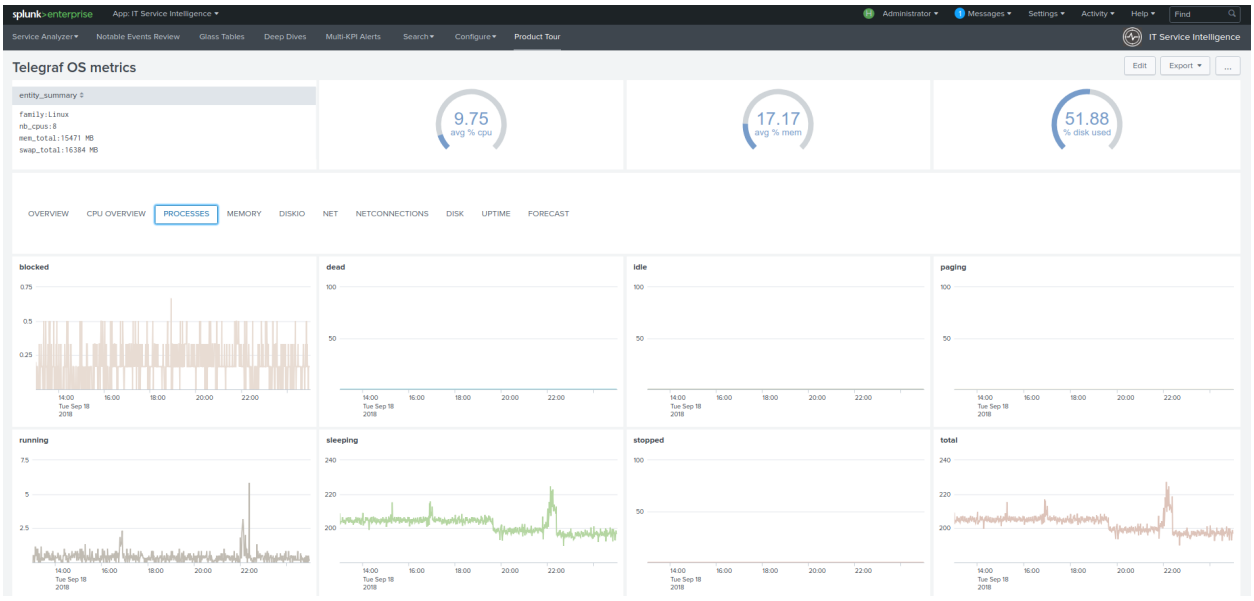
2.5.1 View for Linux OS

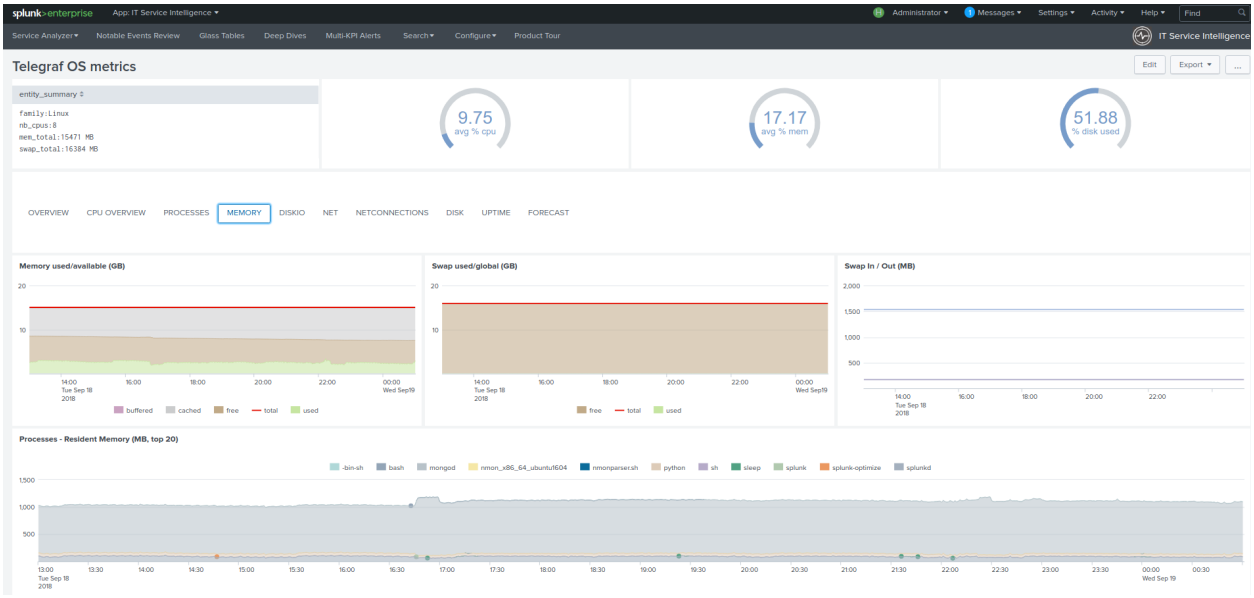
The Health view for Linux OS automatically appears as “Telegraf OS (Linux)” deepdive drilldown link when entities are discovered:

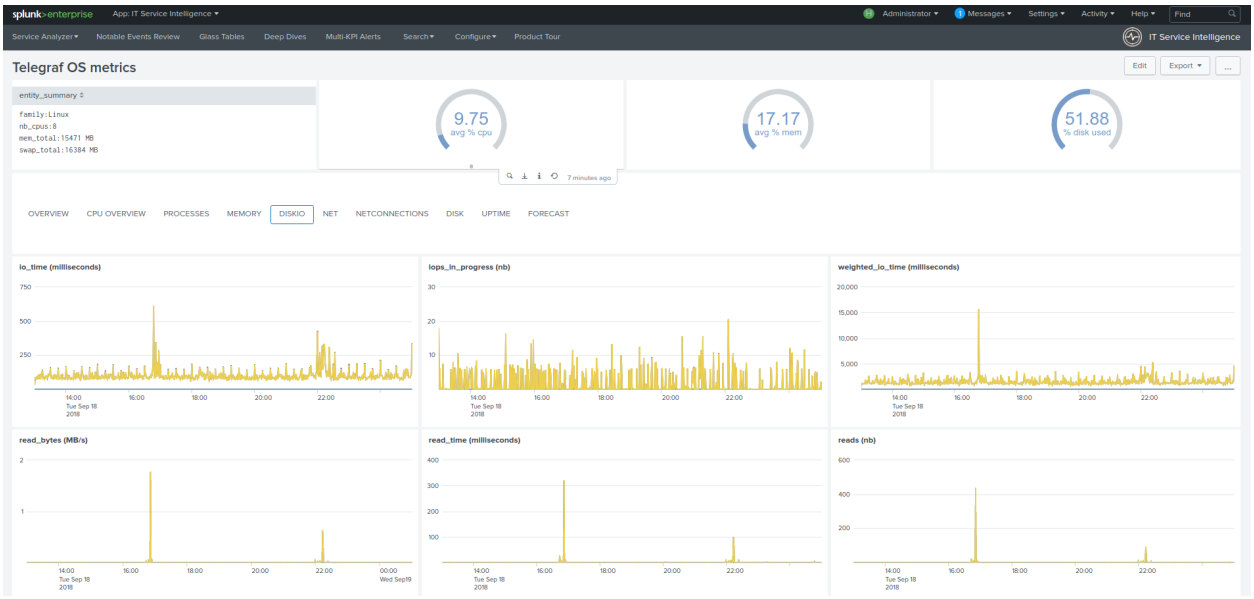


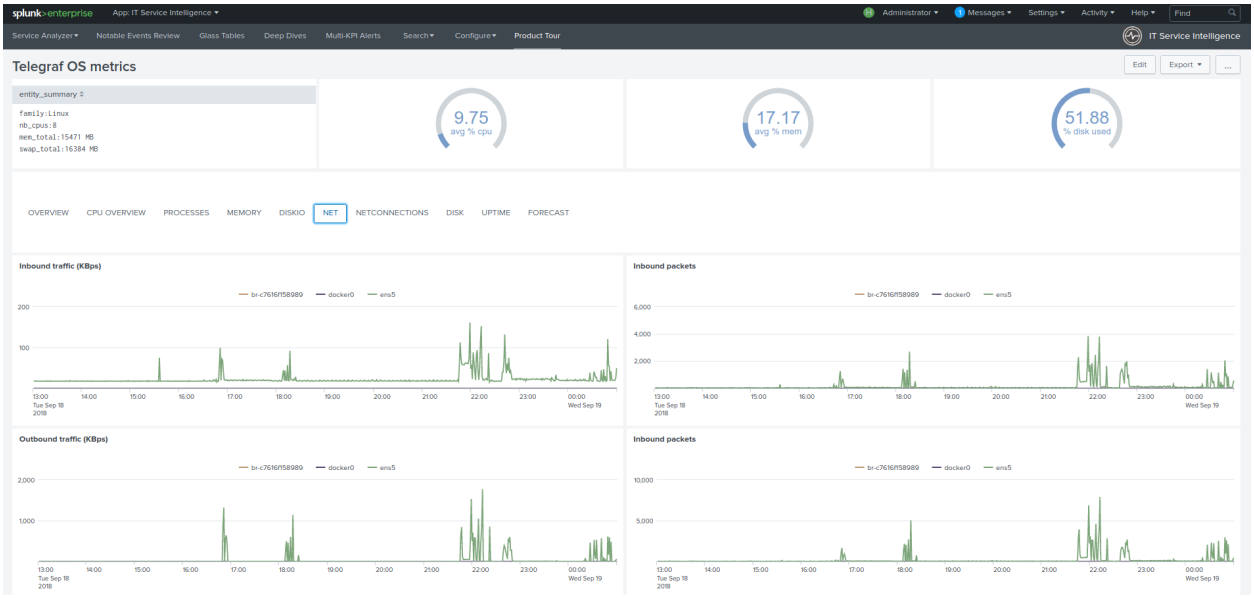

splunk> App: IT Service Intelligence
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
ip-10-0-0-173
Last 12 hours
host ip-10-0-0-173
family Linux
itsi_role telegraf_host
mem_total 15471
nb_cpus 8
swap_total 16384
Edit Entity
Modules
Telegraf OS (Linux)
Overall Health
KPIs
Severity
KPI
Service
Sparkline
Alert Value
Normal CPU Utilization: % My department 9.2
Normal CPU Wait I/O % My department 0.1
Normal Memory Free: % My department 85.0
Normal Storage Used Inodes: % My department 1.2
Normal Storage Used Space: % My department 51.9
Normal Swap Memory Free: % My department 100.0
Normal CPU Utilization: % Telegraf Linux 9.2
Normal CPU Wait I/O % Telegraf Linux 0.1
Normal Memory Free: % Telegraf Linux 85.0
Normal Storage Used Inodes: % Telegraf Linux 1.2
« prev 1 2 3 4 next »
Services
Severity
Service
Sparkline
Alert Value
Normal My department 100.0
Normal Telegraf Linux 100.0
Notable Events
No results found.

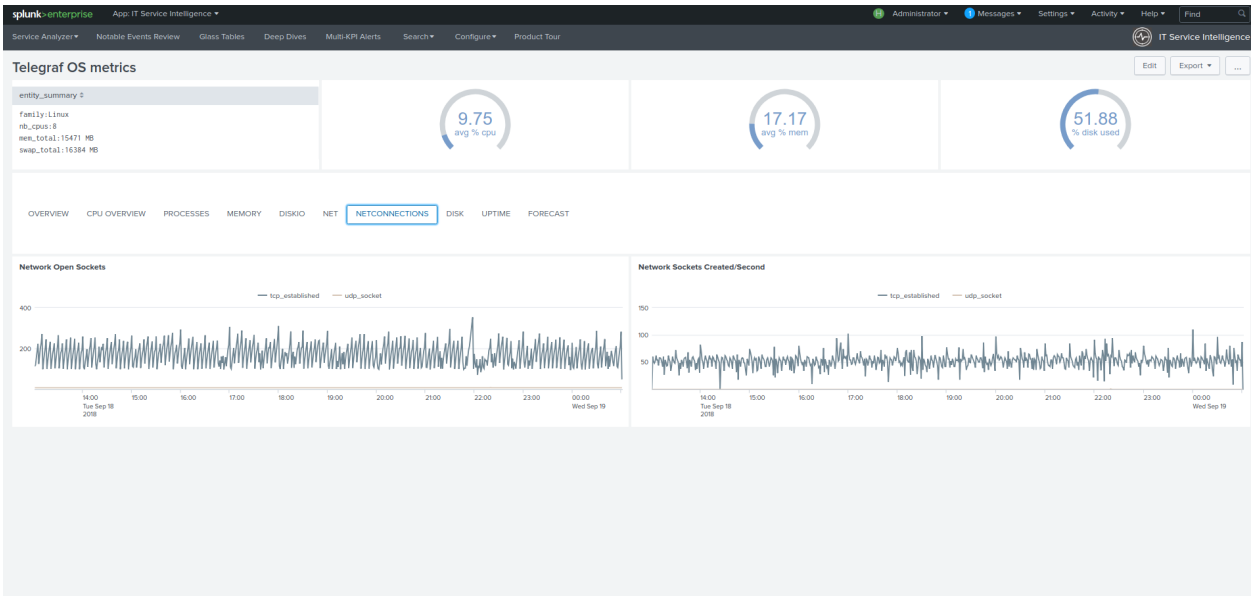

splunk>enterprise App: IT Service Intelligence
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
CPU Usage by category
avg cpu idle %
avg cpu iowait %
avg cpu system %
avg cpu user %
CPU Usage by core
Number of running processes
running_processes
Processes CPU usage (%, top 20)
-bin-sh
bash
mongod
nmon_x86_64_ubuntu1604
python
sh
splunk
splunkd

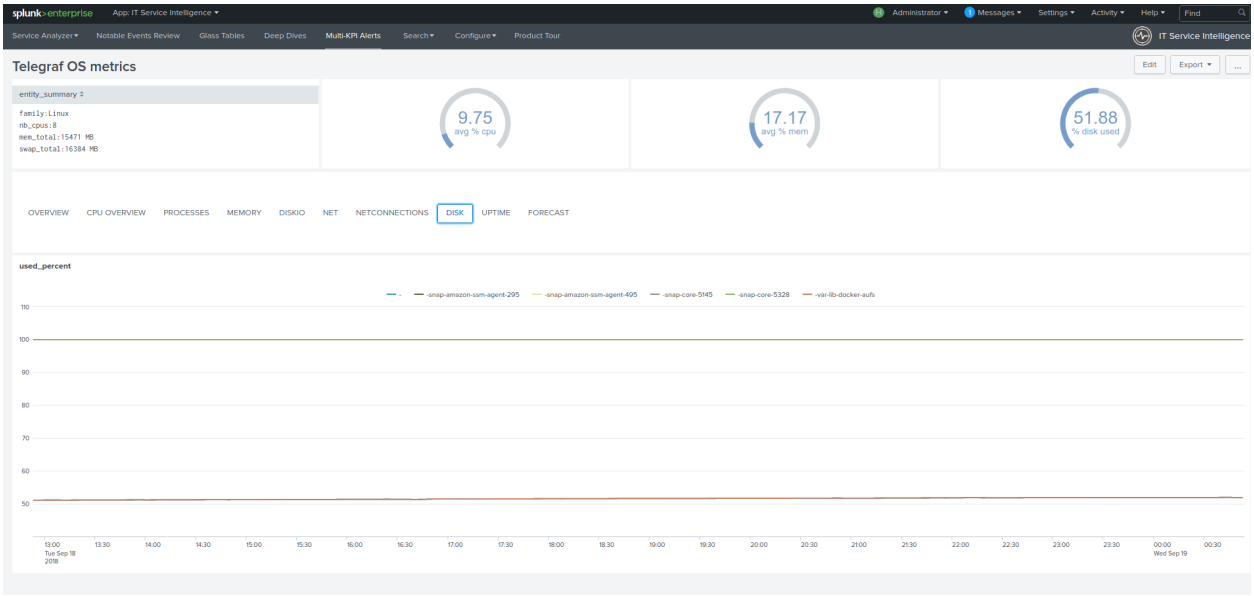

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
blocked
dead
idle
paging
running
sleeping
stopped
total

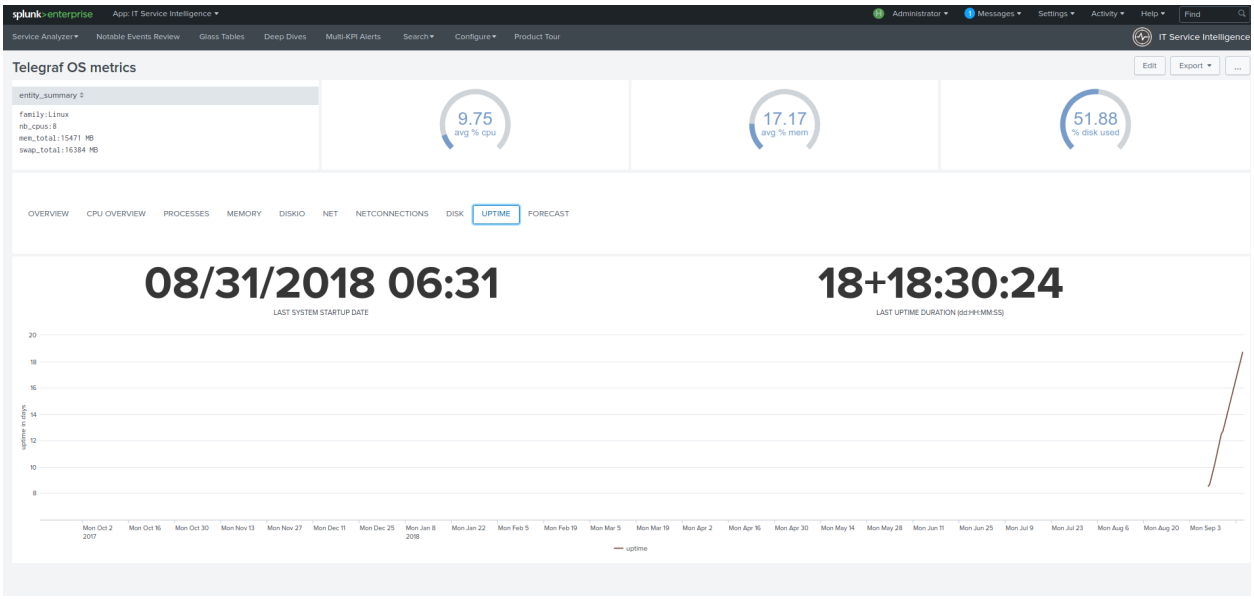

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
Memory used/available (GB)
buffered
cached
free
total
used
Swap used/global (GB)
free
total
used
Swap In / Out (MB)
Processes - Resident Memory (MB, top 20)

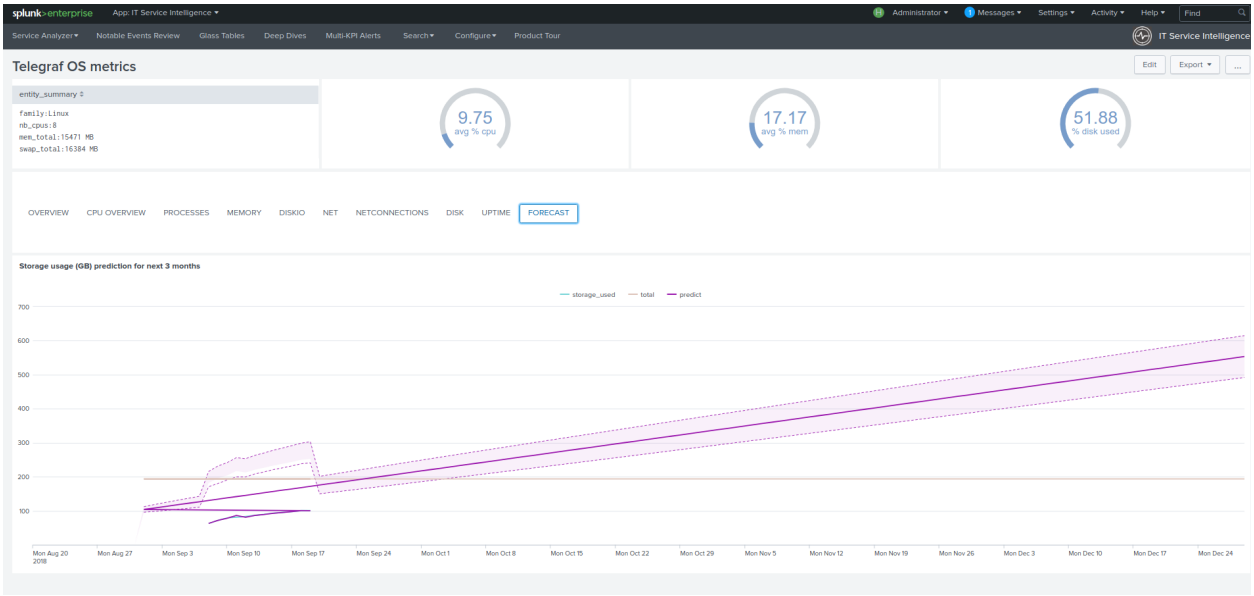

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
7 minutes ago
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
io_time (milliseconds)
iops_in_progress (nb)
weighted_io_time (milliseconds)
read_bytes (MB/s)
read_time (milliseconds)
reads (nb)


splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
Inbound traffic (KBps)
Inbound packets
Outbound traffic (KBps)
Inbound packets


splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
Network Open Sockets
tcp_established
udp_socket
Network Sockets Created/Second
tcp_established
udp_socket

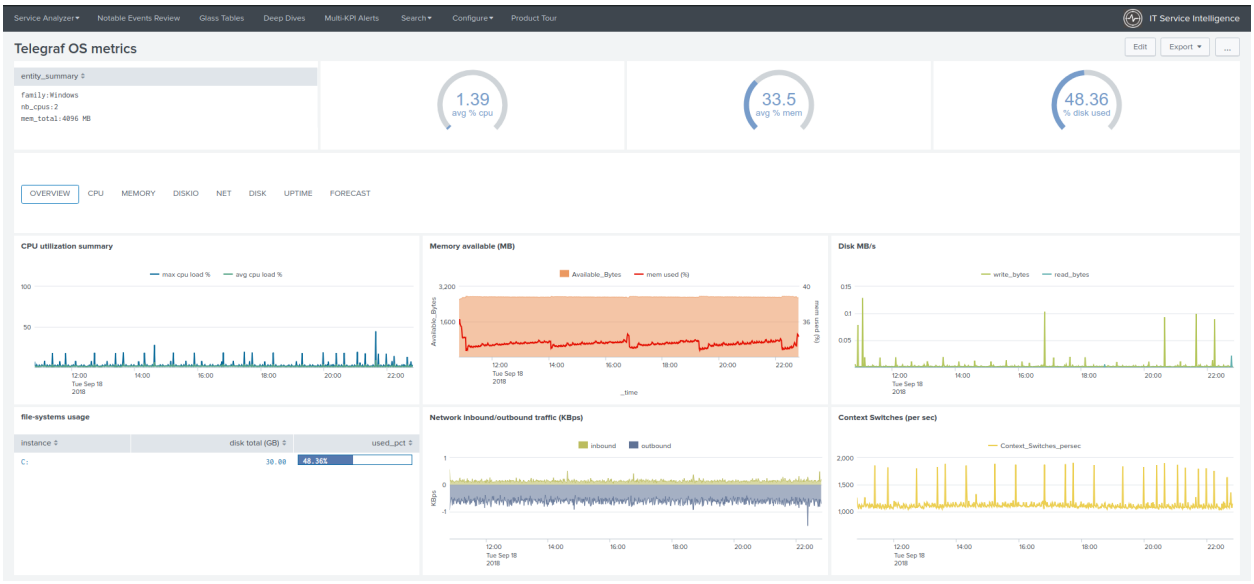

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Linux
nb_cpus:8
mem_total:15471 MB
swap_total:16384 MB
9.75
avg % cpu
17.17
avg % mem
51.88
% disk used
OVERVIEW
CPU OVERVIEW
PROCESSES
MEMORY
DISKIO
NET
NETCONNECTIONS
DISK
UPTIME
FORECAST
used_percent

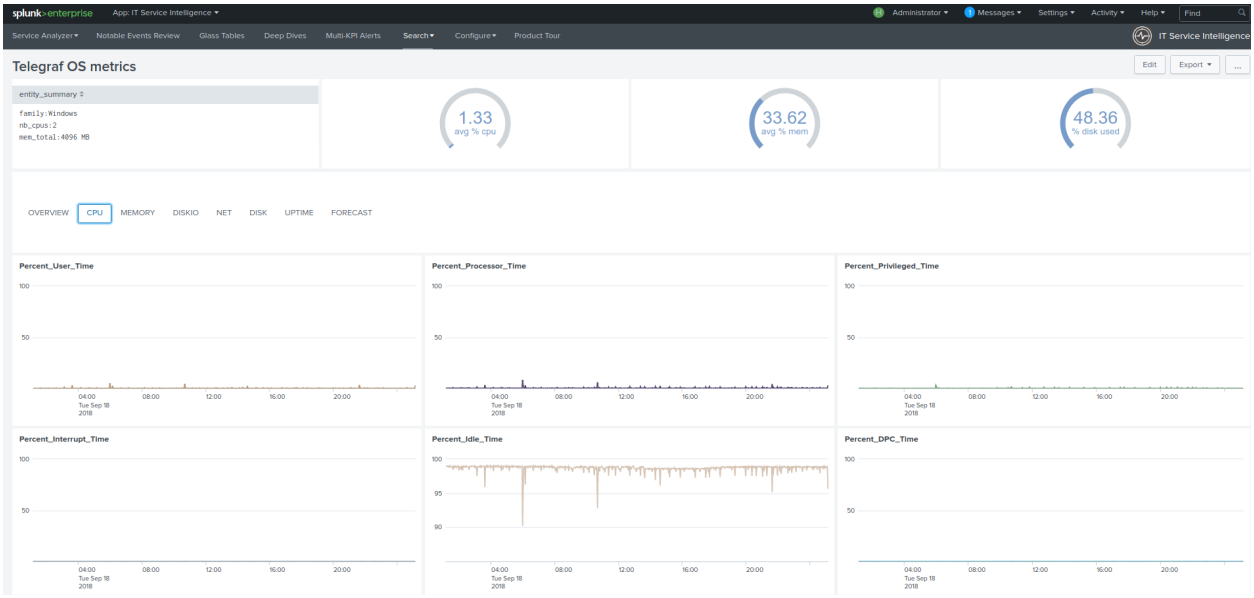


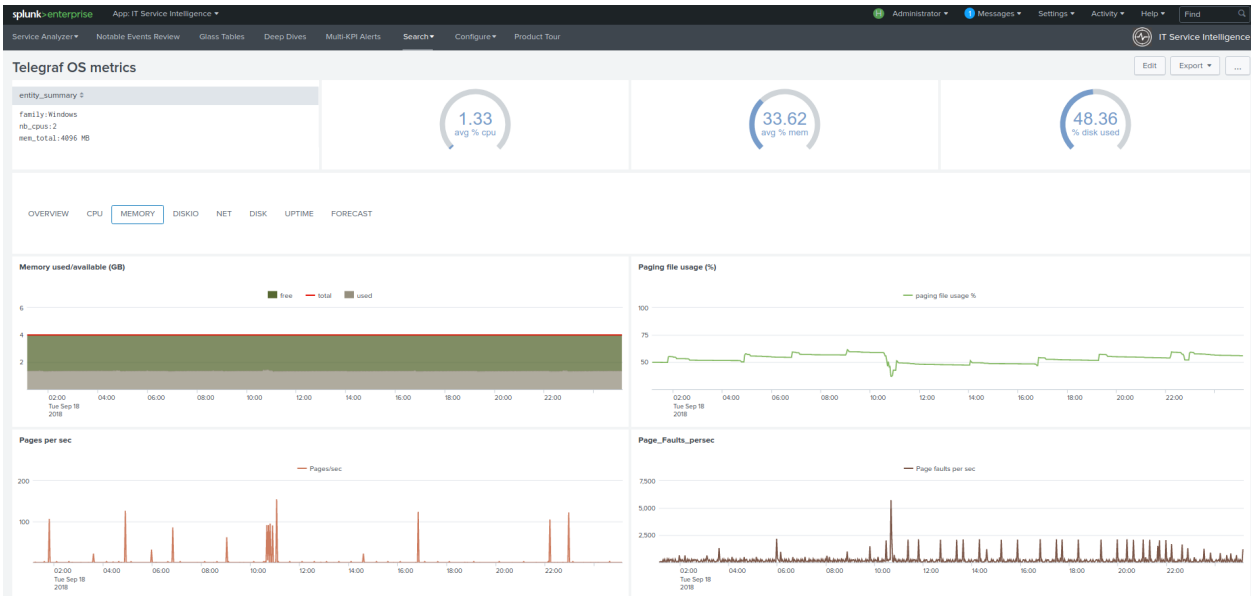


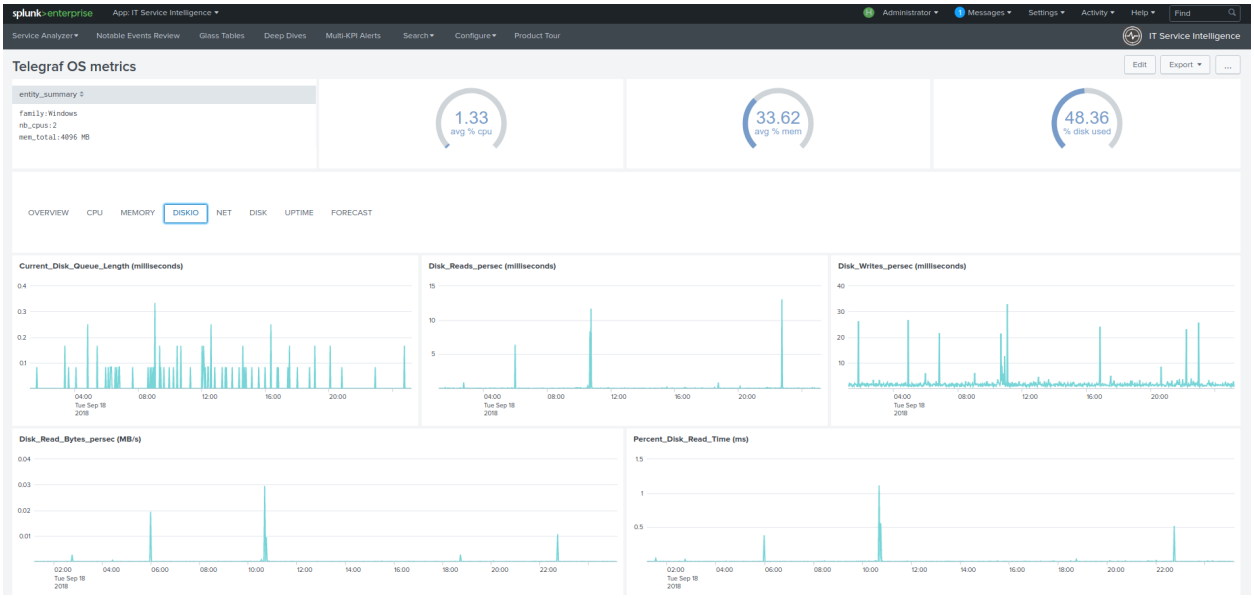
2.5.2 View for Windows OS

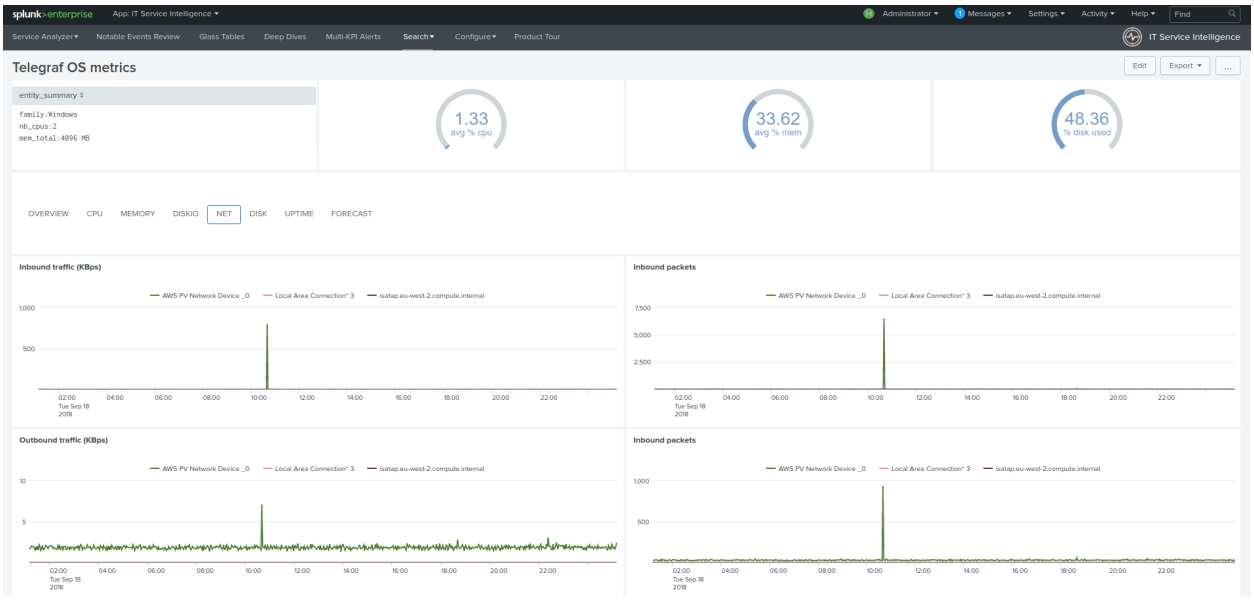
The Health view for Linux OS automatically appears as “Telegraf OS (Windows)” deepdive drilldown link when entities are discovered:

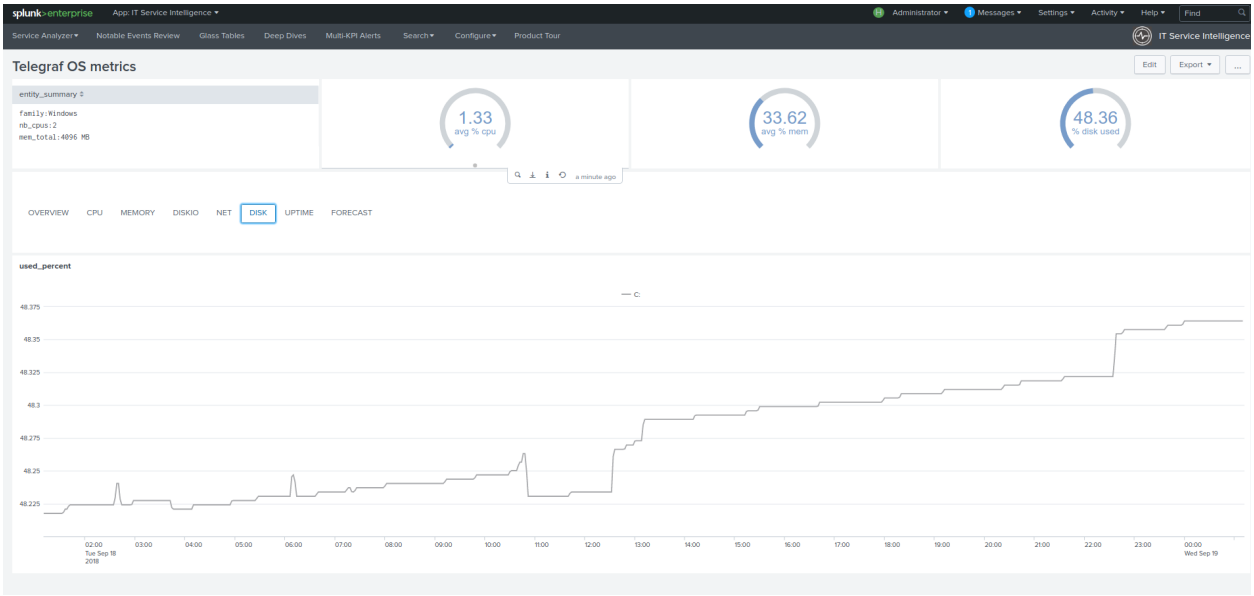

Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.39
avg % cpu
33.5
avg % mem
48.36
% disk used
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
CPU utilization summary
max cpu load %
avg cpu load %
Memory available (MB)
Available_Bytes
mem used (%)
Disk MB/s
write_bytes
read_bytes
file-systems usage
instance
disk total (GB)
used_pct
C:
30.00
48.36%
Network inbound/outbound traffic (KBps)
inbound
outbound
Context Switches (per sec)
Context_Switches_persec

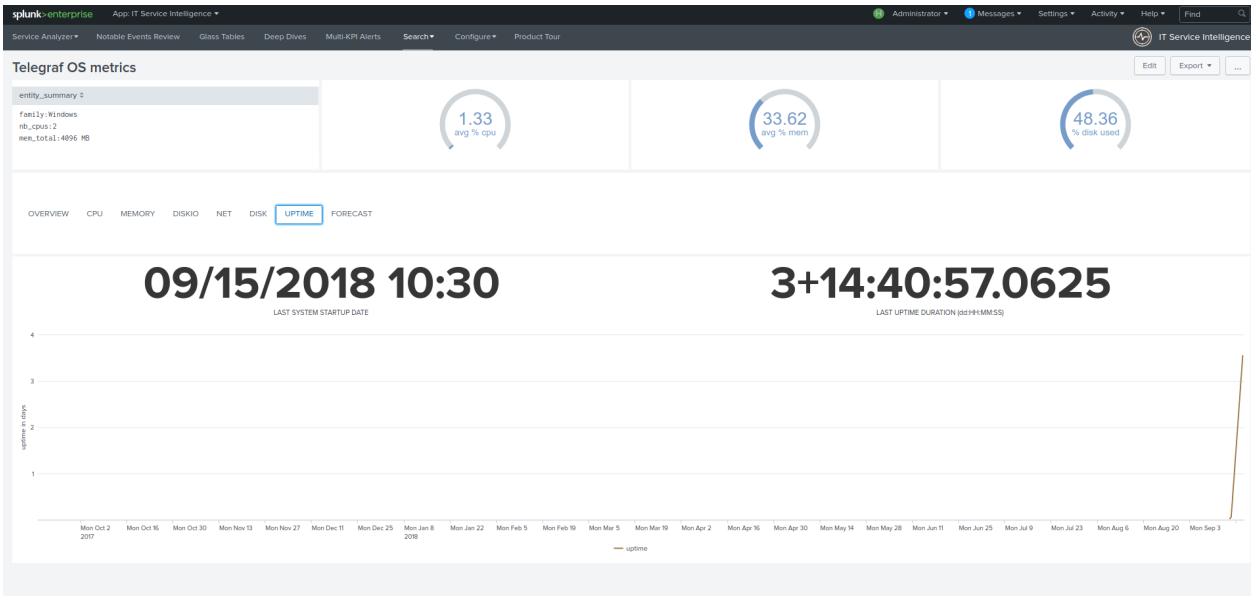

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.33
avg % cpu
33.62
avg % mem
48.36
% disk used
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
Percent_User_Time
Percent_Processor_Time
Percent_Privileged_Time
Percent_Interrupt_Time
Percent_Idle_Time
Percent_DPC_Time

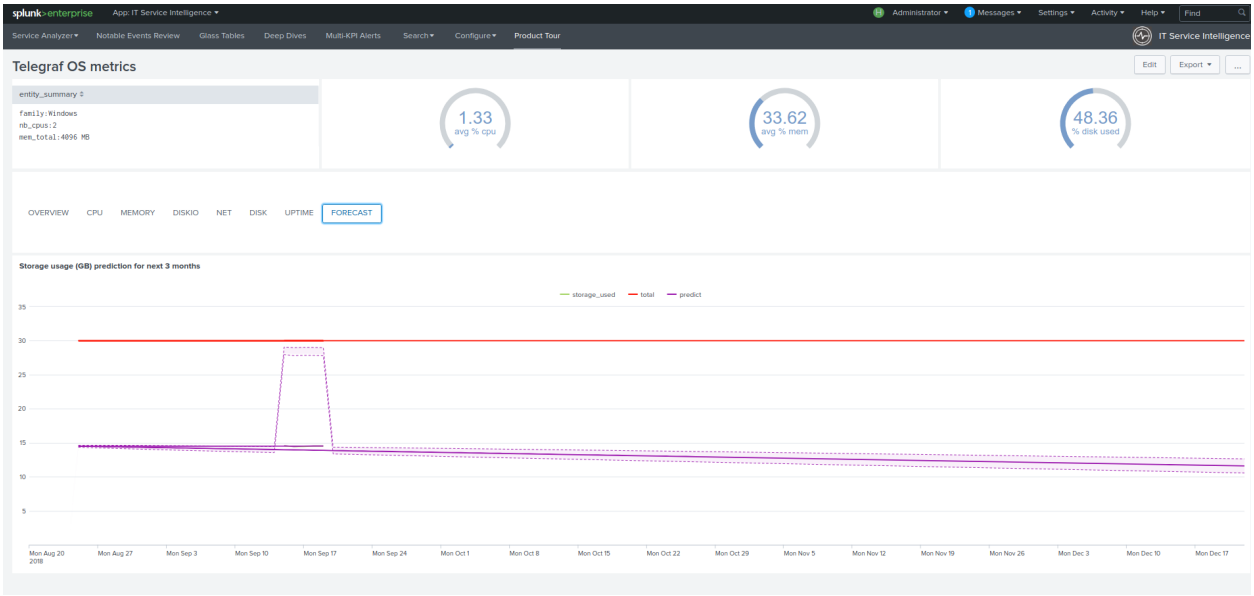

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.33
avg % cpu
33.62
avg % mem
48.36
% disk used
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
Memory used/available (GB)
free
total
used
Paging file usage (%)
paging file usage %
Pages per sec
Pages/sec
Page_Faults_persec
Page faults per sec


splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.33
avg % cpu
33.62
avg % mem
48.36
% disk used
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
Current_Disk_Queue_Length (milliseconds)
Disk_Reads_persec (milliseconds)
Disk_Writes_persec (milliseconds)
Disk_Read_Bytes_persec (MB/s)
Percent_Disk_Read_Time (ms)


splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.33
avg % cpu
33.62
avg % mem
48.36
% disk used
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
Inbound traffic (KBps)
Inbound packets
Outbound traffic (KBps)
Inbound packets
AWS PV Network Device _0
Local Area Connection* 3
Isatap.eu-west-2.compute.internal

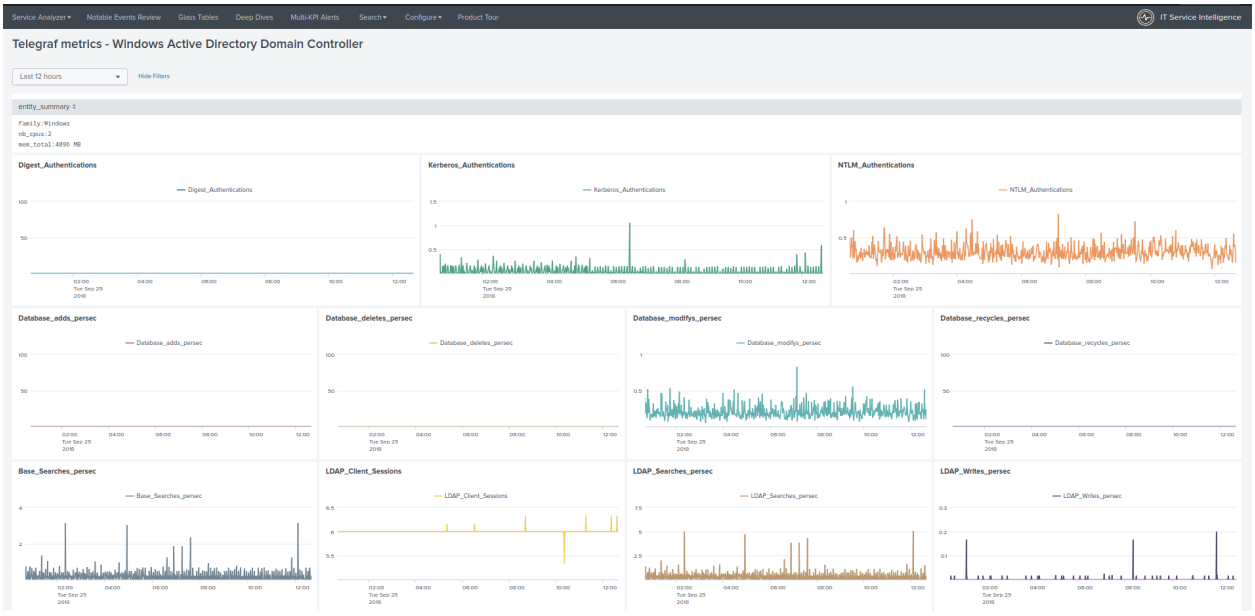

splunk>enterprise
App: IT Service Intelligence
Administrator
Messages
Settings
Activity
Help
Find
Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf OS metrics
Edit
Export
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
1.33
avg % cpu
33.62
avg % mem
48.36
% disk used
a minute ago
OVERVIEW
CPU
MEMORY
DISKIO
NET
DISK
UPTIME
FORECAST
used_percent
C:





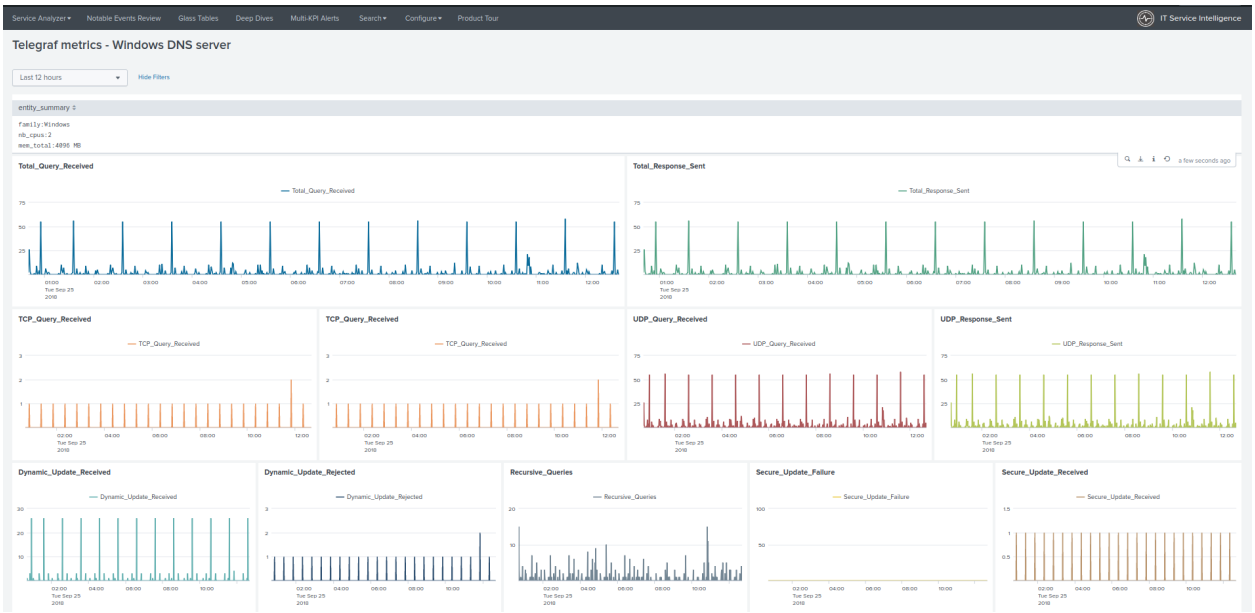
2.5.3 View for Windows Active Directory Domain Controller

The Health view automatically appears as “Telegraf Win AD-DC” deepdive drilldown link when entities are discovered:



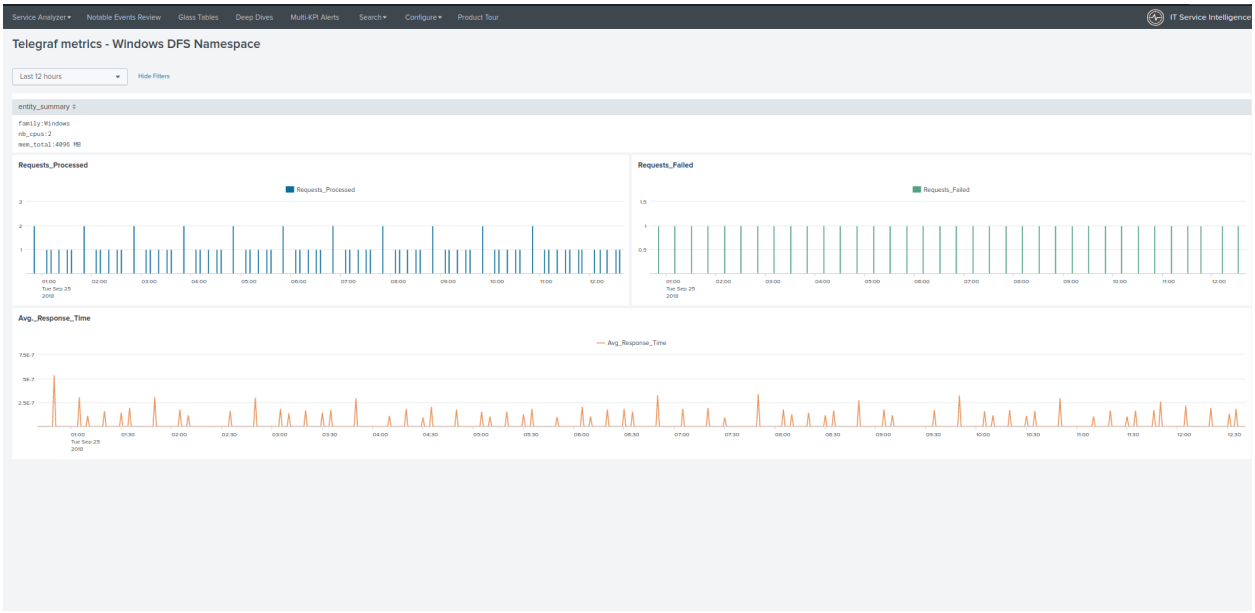
2.5.4 View for Windows DNS

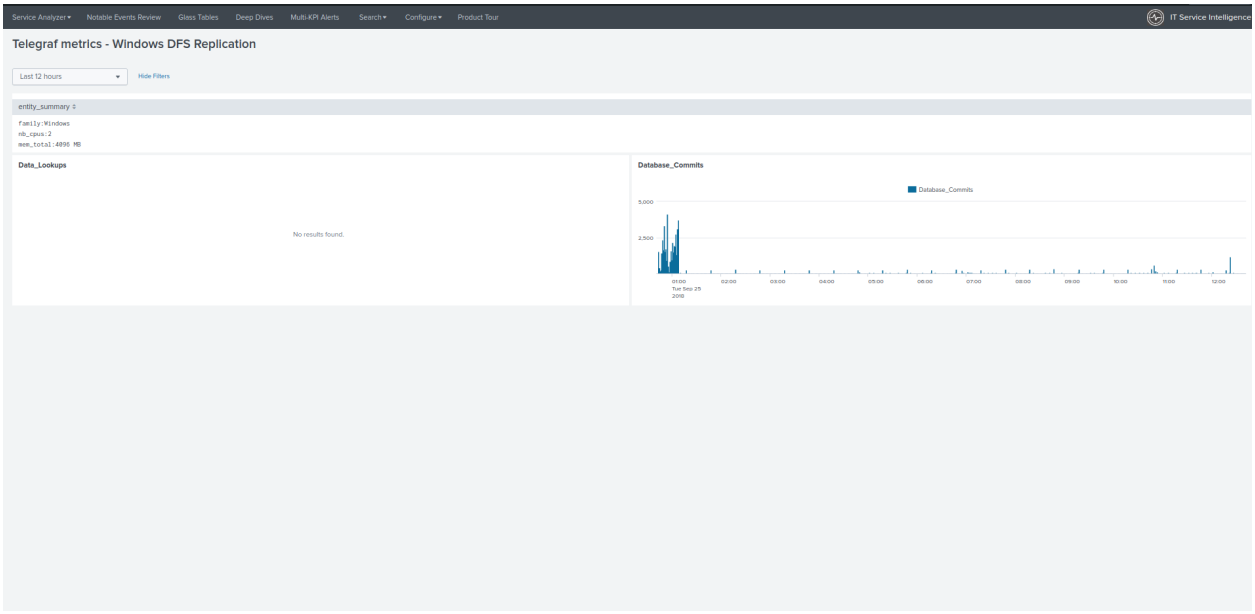
The Health view automatically appears as “Telegraf Win AD-DC” deepdive drilldown link when entities are discovered:



2.5.5 View for Windows DFS

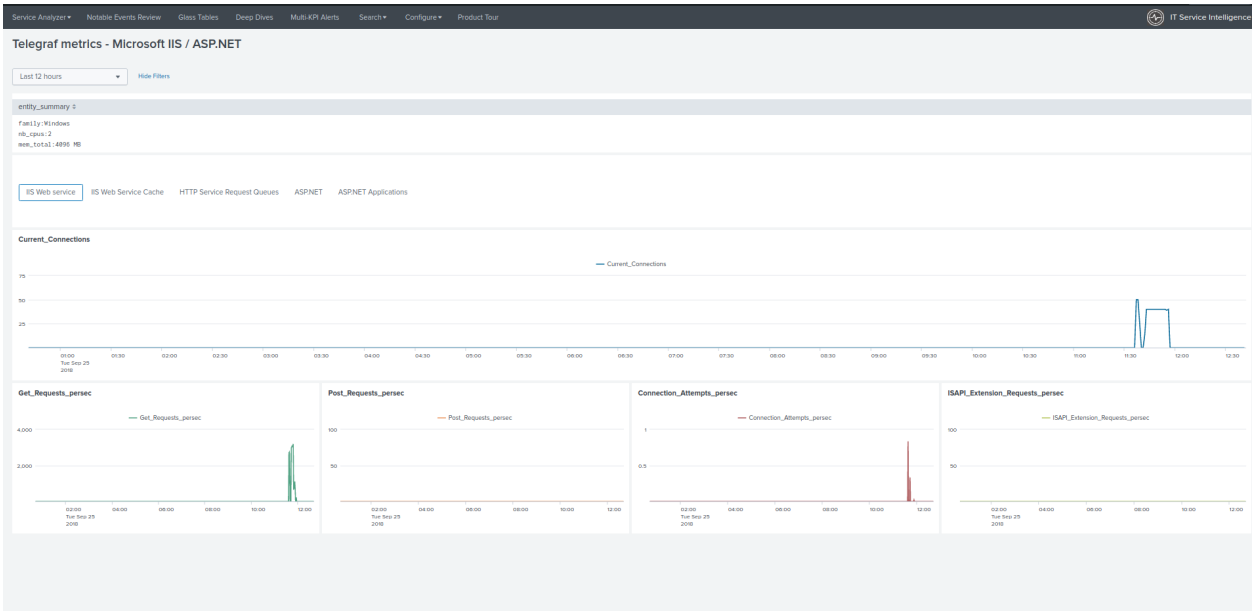
The Health view automatically appear as “Telegraf Win DFS-NS” and “Telegraf Win DFS-REP” deepdive drilldown links when entities are discovered:





2.5.6 View for Microsoft IIS/ASP.NET

The Health view automatically appears as “Telegraf IIS / ASP.NET” deepdive drilldown links when entities are discovered:


Service Analyzer
Notable Events Review
Glass Tables
Deep Dives
Multi-KPI Alerts
Search
Configure
Product Tour
IT Service Intelligence
Telegraf metrics - Microsoft IIS / ASP.NET
Last 12 hours
Hide Filters
entity_summary
family:Windows
nb_cpus:2
mem_total:4096 MB
IIS Web service
IIS Web Service Cache
HTTP Service Request Queues
ASP.NET
ASP.NET Applications
Current_Connections
Current_Connections
Get_Requests_persec
Get_Requests_persec
Post_Requests_persec
Post_Requests_persec
Connection_Attempts_persec
Connection_Attempts_persec
ISAPI_Extension_Requests_persec
ISAPI_Extension_Requests_persec

CHAPTER 3

Troubleshoot:

3.1 Troubleshoot & FAQ

3.1.1 Missing metrics for Windows memory

For Windows memory management, the default win_mem inputs does not retrieve some of the metrics we need.

You need to activate the memory inputs. (which on Windows uses WMI collection):

```
[[inputs.mem]]
# no configuration
```

3.1.2 Empy processes metrics (procstat)

In the linux views, the processes usage (both CPU and Memory) rely on the procstat inputs, which requires additional configuration depending on your context.

As for an example, the following configuration monitors all the processes owned by the “splunk” unix user:

```
[[inputs.procstat]]
#   ## PID file to monitor process
#   pid_file = "/var/run/nginx.pid"
#   ## executable name (ie, pgrep <exe>)
#   # exe = "nginx"
#   ## pattern as argument for pgrep (ie, pgrep -f <pattern>)
#   # pattern = "nginx"
#   ## user as argument for pgrep (ie, pgrep -u <user>)
    user = "splunk"
```

CHAPTER 4

Versioniong and build history:

4.1 Release notes

4.1.1 Version 1.0.6

- fix: Hard coded index name references (should be referring to the global macro)
- fix: Improved nix family detection, avoid relying on kernel metrics which is Linux specific and might mot be available if not configured
- fix: Missing host filter for network panel in Windows OS entity view
- fix: Prevents entities dup creation at installation time if data is available, switching to cron based schedule of modular inputs instead of seconds based

4.1.2 Version 1.0.5

- fix: bad selected metrics in Windows DNS view

4.1.3 Version 1.0.4

- fix: add machine name in entity summary display
- fix: reduce input token panel size in views

4.1.4 Version 1.0.3

- fix: missing by clause statement in diskio KPI basesearch for Linux generates wrong results
- fix: fixed remaining explicit calls to telegraf index instead of index macro

4.1.5 Version 1.0.2

- fix: Missing disk usage KPI base search / KPI template for Windows OS
- fix: Missing time range picker in Health views
- feature: entity discovery enhancement for Windows advanced metrics (AD, DFS, IIS/ASP, DNS)
- feature: Health view support for Windows Active directory
- feature: Health view support for Windows DNS servers
- feature: Health view support for Windows DFS namespace and replication
- feature: Health view support for Microsoft IIS/ASP.net

4.1.6 Version 1.0.1

- fix: total vol size conversion missing for Linux OS view

4.1.7 Version 1.0.0

- initial and first public release