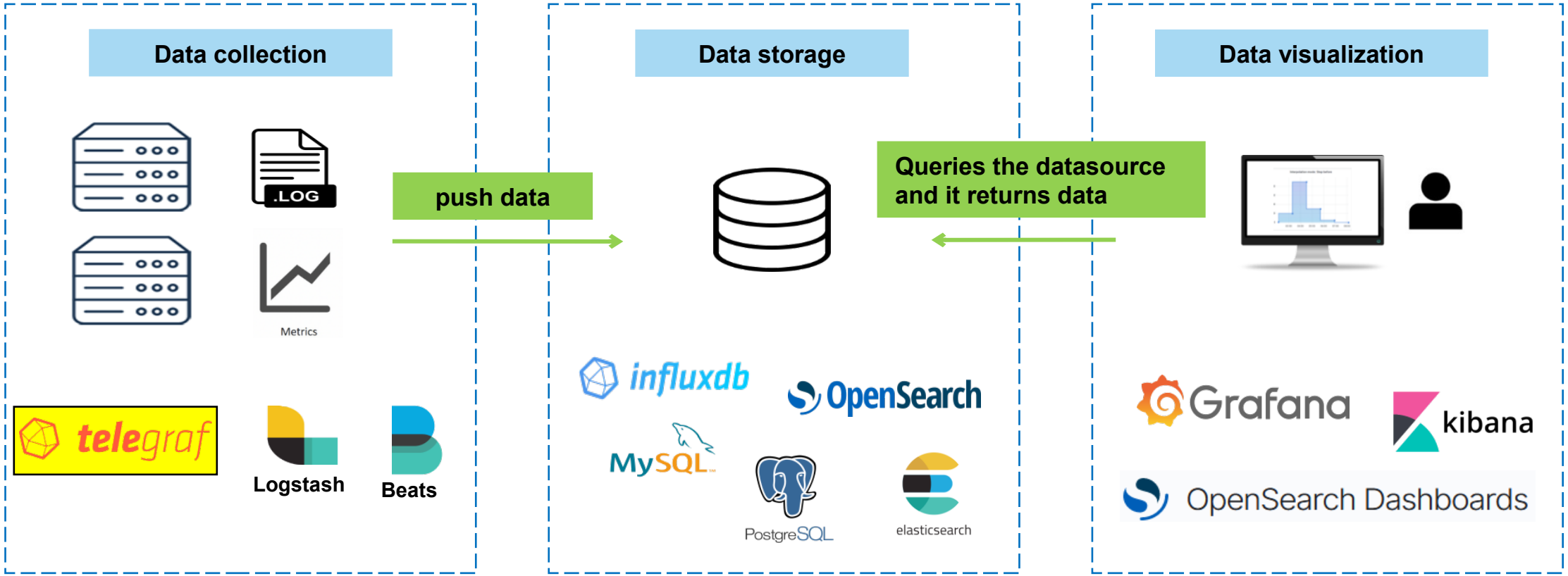
Karlsruhe Institute of Technology

Telegraf

Introduction and overview

Components of monitoring architecture



Agenda

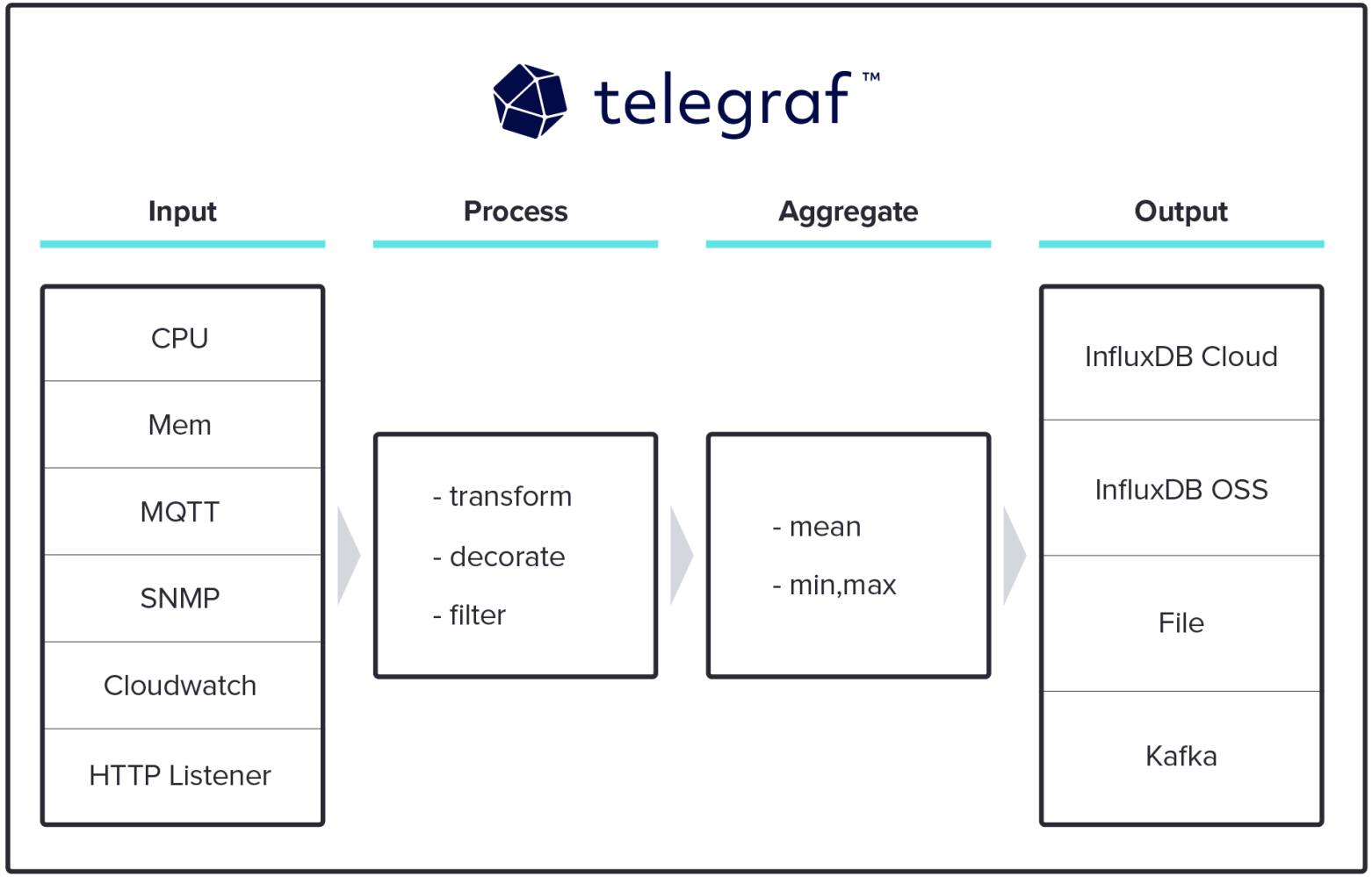
- What is Telegraf?
- What are plugins?
- How to use it at SDM?
- How to implement your own metrics?

Telegraf

- server-based agent for collecting and sending metrics
- written in Go
- compiles into a single binary with no external dependencies
- requires a very minimal memory footprint

- Learn more:
  - https://university.influxdata.com/courses/data-collection-with-telegraf-tutorial/
  - https://www.influxdata.com/time-series-platform/telegraf/
  - https://github.com/influxdata/telegraf

Plugins I


telegraf™
Input
Process
Aggregate
Output
CPU
Mem
MQTT
SNMP
Cloudwatch
HTTP Listener
- transform
- decorate
- filter
- mean
- min,max
InfluxDB Cloud
InfluxDB OSS
File
Kafka

Plugins II

- more than 300 plugins availabe: https://www.influxdata.com/products/integrations/

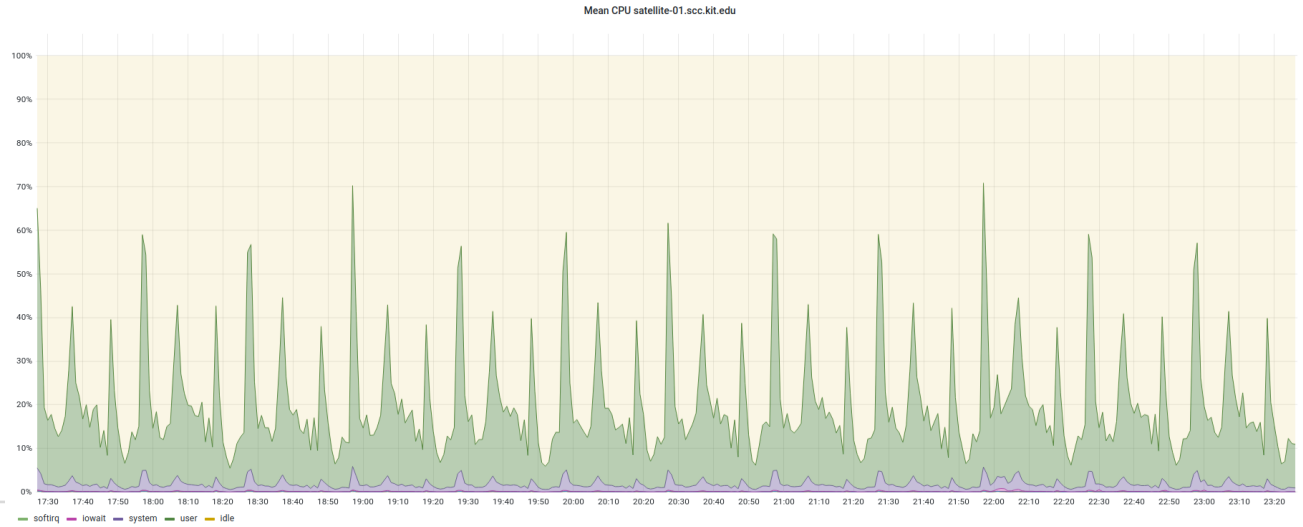
activemq	chrony	dmcache	filestat
aerospike	cisco_telemetry_mdt	dns_query	fireboard
amqp_consumer	clickhouse	docker	fluentd
apache	cloud_pubsub	docker_log	github
apcupsd	cloud_pubsub_push	dovecot	gnmi
aurora	cloudwatch	ecs	graylog
azure_storage_que ue	conntrack	elasticsearch	haproxy
bcache	consul	ethtool	hddtemp
beanstalkd	couchbase	eventhub_consumer	http
bind	couchdb	exec	http_listener_v2
bond	cpu	execd	http_response
burrow	dcos	fail2ban	httpjson
cassandra	disk	fibaro	icinga2
ceph	diskio	file	infiniband
cgroup	disque	filecount	influxdb

influxdb_listener	kapacitor	mcrouter	net_response
influxdb_v2_listener	kernel	mem	nginx
intel_rdt	kernel_vmstat	memcached	nginx_plus
internal	kibana	mesos	nginx_plus_api
interrupts	kinesis_consumer	minecraft	nginx_sts
ipmi_sensor	kube_inventory	modbus	nginx_upstream_check
ipset	kubernetes	mongodb	nginx_vts
iptables	lanz	monit	nsd
ipvs	leofs	mqtt_consumer	nsq
jenkins	linux_sysctl_fs	multifile	nsq_consumer
jolokia	logparser	mysql	nstat
jolokia2	logstash	nats	ntpq
jti_openconfig_telemetr y	lustre2	nats_consumer	nvidia_smi
kafka_consumer	mailchimp	neptune_apex	opcua
kafka_consumer_legacy	marklogic	net	openldap

openntpd	prometheus	snmp_legacy	tcp_listener	win_services
opensmtpd	proxmox	snmp_trap	teamspeak	wireguard
openweathermap	puppetagent	socket_listener	temp	wireless
passenger	rabbitmq	solr	tengine	x509_cert
pf	raindrops	sqlserver	tomcat	zfs
pgbouncer	ras	stackdriver	trig	zipkin
phpfpm	redfish	statsd	twemproxy	zookeeper
ping	redis	suricata	udp_listener	
postfix	rethinkdb	swap	unbound	
postgresql	riak	synproxy	uwsgi	
postgresql_extensib le	salesforce	syslog	varnish	
powerdns	sensors	sysstat	vsphere	
powerdns_recursor	sflow	system	webhooks	
processes	smart	systemd_units	win_eventlog	
procstat	snmp	tail	win_perf_counte rs	

How to use it?

- telegraf is installed automatically via puppet
- many plugins are alread activated
- metrics are stored in influx databases
  - lsdf-28-124.scc.kit.edu
  - influx.gridka.de
- metrics can be visuallized in grafana




```
telegraf::inputs:
  bond: [{}]
  cpu:
    percpu: true
    totalcpu: true
  disk:
    ignore_fs:
      - tmpfs
      - devtmpfs
      - devfs
      - gpfs
  diskio: [{}]
  kernel: [{}]
  kernel_vmstat: [{}]
  linux_sysctl_fs: [{}]
  mem: [{}]
  net: [{}]
  netstat: [{}]
  processes: [{}]
  swap: [{}]
  system: [{}]
telegraf::outputs:
  influxdb:
```

Define your own metric

- EXEC-Plugin allows for easy implementation of your own metrics
- Implement a script that calculates your metric
  - Bash, Python, GO ...
  - Script outputs one line: MetricName value1=$value1, value2=$value2
- telegraf uses the EXEC-Plugin to run your script

- How?
  - Expand the existing puppet module https://git-cm.scc.kit.edu/Puppet-Modules/monitoring_telegraf_input
  - Write a new puppet class for your metric
  - Add your script that calculates the new metric to the puppet module
  - Apply the new puppet class on your host

Define your own metric – new puppet class

```
class monitoring_telegraf_input::puppet_compilations{
  telegraf::input { 'puppet_compilations':
    plugin_type => 'exec',
    options     => [{
      'commands'    => [
        'sudo /usr/local/bin/puppet_compilations.sh',
      ],
      'data_format' => 'influx',
      'timeout'     => '20s',
    }],
    require     => File['/usr/local/bin/puppet_compilations.sh'],
  }

  file{'/usr/local/bin/puppet_compilations.sh':
    mode    => '0755',
    owner   => root,
    group   => root,
    source => "puppet:///modules/${module_name}/puppet_compilations.sh",
  }

  sudoers::rules{'monitoring_telegraf_input_puppet_compilations':
    content => "telegraf  ALL = NOPASSWD: /usr/local/bin/puppet_compilations.sh\n"
  }
}
```

Define your own metric – new script

```
#!/bin/bash
PERIODE=60
NOW=$(date -d now +%s)
CNT=0

IFS=$'\n'
while read -r line;
do
   DATE=$(date +%s -d $(echo $line | cut -d" " -f1))
   DIFF=$((${NOW}-${DATE}))
   if [ ${DIFF} -lt ${PERIODE} ]; then
       CNT=$((CNT+1))
   fi
done <<< "$(tail -n100 /var/log/puppetlabs/puppetserver/puppetserver.log | grep Compile)"

echo PuppetCompilations Count=${CNT}
```

Define your own metric – visualization