ntop and Checkmk.
A dream team for network monitoring.

ntopConf ’23

checkmk.com
Who I am

Love to build products ...

... and break things

Since 2018

**VP Product**, Checkmk

2017 - 2018

**Chief of Staff**, Teamviewer

2014 - 2016

**Consultant**, The Boston Consulting Group
Agenda

01 Introduction Checkmk

02 Checkmk + ntop

03 Learnings from building a ntop integration
Fast track your time-to-resolution

- 2,000 + plug-ins
- Auto-discovery
- Smart and granular alerts
- Ticket auto-creation in ITSM tools
- Out-of-the-box health assessment
- Built-in time series graphs
- Forecasting and capacity management
- Built-in and custom dashboards
- Root-cause analysis
- Custom self-healing
Our mission: bringing visibility into your hybrid IT

- Cloud-native & traditional workloads
- One integrated monitoring

Your Data Centers

Your Clouds

Azure

AWS

Google Cloud
Monitor everything: 2,000+ well-maintained plug-ins

Data center

Applications
- Databases
- Web Apps
- Application servers
- Web servers
- Message queues / caches
- Business software

Networking
- Router
- Firewalls
- Hyper-visors
- Containers
- Kubernetes

Virtualization
- Linux
- Windows
- MacOS
- AIX
- OpenVMS
- Solaris
- FreeBSD
- OpenBSD
- NetBSD
- HP-UX
- IBM zOS

Hardware & Sensors
- Server
- Storage
- Sensors
- HVAC
- Power supply
- Security

Cloud

Cloud services
- Apps
- Databases
- Costs
- Cloud Health
- DNS
- Firewall
- VPN
- CDN
- Load Balancers
- Caches
- Notifications
- Kubernetes
- Containers
- Functions
- Virtual Machines
- Block storage
- Object storage
- Archives
Highly automated to monitor at scale

Auto-register workloads

Asset starts (VM, pod, …) → Asset autoregisters → Monitoring services & metrics automatically added → Asset is terminated

Monitoring objects and metrics CAN automatically be removed from monitoring
Highly automated to monitor at scale

**Auto-register workloads**

- Auto-register workloads
- Auto-discover services
Highly automated to monitor at scale

- Auto-register workloads
- Auto-discover services
- Metrics dashboards
Highly automated to monitor at scale

Auto-register workloads
Auto-discover services
Metrics dashboards
Application dashboards

Off-the-shelf
Highly automated to monitor at scale

- Auto-register workloads
- Auto-discover services
- Metrics dashboards
- Application dashboards
- Automated alerting

ITSM: ServiceNow, PagerDuty, Splunk On-Call

Messaging: Slack, SMS, Email

Applications: Jira, Mattermost, Ops Genie, Webex teams, MS Teams, Email
Hyper-scalable distributed set-ups

Scale vertically
100k+ services per instance

Scale horizontally
with massively distributed set-ups
Extensible open-source monitoring

Build your own integrations
with simple scripts extending agents or by writing entire plug-ins yourself

```python
from .agent_based_api.v1 import register, Result, Service, State

def discover_myhostgroups(section):
    yield Service()

def check_myhostgroups(section):
    attr = section.get("check_mk")
    hosts = attr("members") if attr else ""
    if hosts:
        yield Result(state=State.CRIT, summary="Default group is not empty: {hosts}")
    else:
        yield Result(state=State.OK, summary="Everything is fine")

register.check_plugin(
    name="myhostgroups",
    service_name="Hostgroup check_mk",
    discovery_function=discover_myhostgroups,
    check_function=check_myhostgroups,
)
```

Extend existing integrations
to accommodate own requirements

- Majority of code base open source
- Easily readable and modifiable Python code
- Developer APIs for writing monitoring integrations
- Built-in logic to handle customized code
- Large partner ecosystem for customizations
The Checkmk Community
Where IT Monitoring experts meet

User forum
>6,000 users
>10,000 daily visits

GitHub
>180 contributors

Integration exchange
540+ packages

Translations
6 languages
Checkmk — The Company

- 150+ employees, privately held, debt free
- Based in Munich, Germany, and Atlanta, USA
- Focusing on IT monitoring for 15+ years
- Open-source enthusiasts
Checkmk + ntop
Integrate network flow monitoring in Checkmk

What you already get from Checkmk

- Network performance monitoring & metrics
  - Bandwidth (e.g. bits in/out)
  - Packet rate
  - Error rate
- Network interface status and speed
- Alerting

What you might want to do in addition

- Deeper root cause analyses (e.g. quick identification of network bottlenecks)
- Network flow analyses (e.g. top talkers, …)
- In-depth performance monitoring (e.g. delay, round-trip-times, …)
- Support threat detection (e.g. quick identification of threats like DDoS attacks)
Real-life scenario

Low bandwidth capacity at remote location - only 2Mbit :(.
Machines there mostly use Citrix Virtual Desktop. Normally bandwidth sufficient, but every once in a while, completely unusable as someone occupying all bandwidth.

Bandwidth monitoring via Checkmk to receive alert when ‘culprit’ is at work again.

Alert!!!
Look into ntop for analysis. A lot of ssh traffic!

Infrastructure was ‘unknowingly’ used by students who copied a lot of research data around with scp...
Configured alert based on speed of interface
Network statistics and flows of build-fre-001.lan.tribe2k.com

<table>
<thead>
<tr>
<th>Host</th>
<th>Display</th>
<th>Help</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>IP address</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Engaged Alerts</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Score</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Active alerted flows</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**First Seen**
2023-09-09 02:02:33 [11d ago]

**Last Seen**
2023-09-20 13:56:28 [1.00 s ago]

**Sent vs Received Traffic Breakdown**

<table>
<thead>
<tr>
<th>Sent</th>
<th>Received</th>
</tr>
</thead>
<tbody>
<tr>
<td>392,363,949 Pkts / 753.30 GiB</td>
<td>691,852,833 Pkts / 2.24 TIB</td>
</tr>
</tbody>
</table>

**Additional Host Names**

- build-fre-001.lan/tribe2k.com (DNS Resolution)

**As client**

<table>
<thead>
<tr>
<th></th>
<th>As server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>68</td>
</tr>
<tr>
<td>Total</td>
<td>1128056</td>
</tr>
<tr>
<td>Unique</td>
<td>1159</td>
</tr>
<tr>
<td>Peer</td>
<td>18</td>
</tr>
</tbody>
</table>

Investigate via ntop integration: Host overview
Investigate via ntop integration: Host traffic
Investigate via ntop integration: Host peers
Investigate via ntop integration: Host apps
Investigate via ntop integration: Host flows
Investigate via ntop integration: Host alerts
Investigate via ntop integration: Top talkers
<table>
<thead>
<tr>
<th>Application</th>
<th>Protocol</th>
<th>Client</th>
<th>Server</th>
<th>Duration</th>
<th>Score</th>
<th>Breakdown</th>
<th>Actual Thpt</th>
<th>Total Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH</td>
<td>TCP</td>
<td>192.0.0.1</td>
<td>10.0.0.1</td>
<td>&lt;1s</td>
<td>10</td>
<td>Client</td>
<td>0 B</td>
<td>$0.00 E</td>
</tr>
<tr>
<td>SSH</td>
<td>TCP</td>
<td>192.0.0.1</td>
<td>10.0.0.1</td>
<td>&lt;1s</td>
<td>10</td>
<td>Client</td>
<td>0 B</td>
<td>$0.00 E</td>
</tr>
<tr>
<td>SSH</td>
<td>TCP</td>
<td>192.0.0.1</td>
<td>10.0.0.1</td>
<td>&lt;1s</td>
<td>10</td>
<td>Client</td>
<td>0 B</td>
<td>$0.00 E</td>
</tr>
</tbody>
</table>

**Investigate via ntop integration: Flows**
Main purpose of the ntop & Checkmk integration

**Single pane of glass**
Relevant ntop data directly integrated into Checkmk.
Filters to work with huge amounts of ntop data.

**Quick access to further information**
Deep links, both to Checkmk and ntop.
Learnings from building an ntop integration
Some learnings for developers

- There are API differences between commercial and free ntop
  - Make this clear to your users and testers, what is required

- Build it early on for multiple API calls to get data - Retrieving, e.g. all engaged alerts not possible via one API call – we used multiple API calls with categories to get all engaged alerts

- Performance can become a huge topic.
  - Our own environment is ‘very small’ - no performance problems (after we upgraded to ClickHouse DB)
  - Amount of data even in ‘smaller environments’ quickly explodes: 7 TB per day
  - Consider the amount of data of larger environments before you build such an integration – we had to invest a lot of time post-release to solve performance issues

- Definitely work with pagination due to amount of data available in ntop
  - A definite number of items returned by the API (e.g. if you want alerts from past time to present time)

- Work with caching!
ntop REST API can do more than meets the eye

### Get /lua/rest/v2/get/flow/alert/list.lua

Get flow alerts list

- **Description**: Get flow alerts list
- **Produces**: ['application/json']

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Position</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifid</td>
<td>query</td>
<td>Interface identifier</td>
</tr>
<tr>
<td>epoch_begin</td>
<td>query</td>
<td>Start time (epoch)</td>
</tr>
<tr>
<td>epoch_end</td>
<td>query</td>
<td>End time (epoch)</td>
</tr>
<tr>
<td>alert_id</td>
<td>query</td>
<td>Alert identifier (format: 'id: eq', where 'id' is the alert identifier)</td>
</tr>
<tr>
<td>severity</td>
<td>query</td>
<td>Severity identifier (format: 'id: eq', where 'id' is the severity identifier)</td>
</tr>
<tr>
<td>score</td>
<td>query</td>
<td>Score (format: 'id: eq', where 'id' is the score identifier)</td>
</tr>
<tr>
<td>ip_version</td>
<td>query</td>
<td>IP version (format: 'id: eq', where 'id' is the IP version)</td>
</tr>
<tr>
<td>ip</td>
<td>query</td>
<td>IP (format: 'id: eq', where 'id' is the IP)</td>
</tr>
<tr>
<td>cli_ip</td>
<td>query</td>
<td>Client IP (format: 'id: eq', where 'id' is the client IP)</td>
</tr>
<tr>
<td>srv_ip</td>
<td>query</td>
<td>Server IP (format: 'id: eq', where 'id' is the server IP)</td>
</tr>
</tbody>
</table>

This is also possible. 3 parameters are NOT documented :-)

.../list.lua?start=0&length=10&ifid=2&status=historical

Thanks for the ntop team! Very helpful. Very resourceful. Very responsive!