Cloud Monitoring

CHECKM K CONFERENCE #5 – MUNICH, APRIL 29, 2019
CLOUD MONITORING

Agenda

1. PARADIGMhifts AND HOW WE ADDRESS THEM
2. OUR APPROACH TO CLOUD MONITORING
3. AWS
4. AZURE
Paradigm Shifts in the Cloud

Highly dynamic environments

“Labels“ as central concept to manage infrastructure

Rate and resource limits are externally imposed

Costs are pay-as-you-grow – but grow quickly
Dynamic Configuration Daemon (DCD) was built for these more dynamic cloud environments.

- For AWS and Azure, DCD performs two primary tasks:
  - Dynamic creation of Piggyback Hosts
  - Automatic Service Discovery
ADDRESSING THE CHALLENGES
How: Handling of ‘Service Labels’

- Automatic discovery of labels through HW/SW inventory ("Host Label Discovery")

- Two use cases:
  - Searching hosts and services
  - Special Agent configuration: limiting retrieved data to certain labels
ADDRESSING THE CHALLENGES

How: Handling of Rate & Resource Limits

- Resource limits
  - `checkmk` monitors account resource limits
  - Limits provided by API (usually), individual limits (i.e. from custom contracts) can be edited
  - Limits are monitored at the account level (e.g. max 20 EC2 instances per region)
  - Monitoring at a resource level where sensible
How: Handling of Rate & Resource Limits

- Rate limits
  - Azure Agent monitors rate limits for Azure API
  - To limit use, agent bundles requests and internally caches data
  - Can be further optimized through explicit config
### ADDRESSING THE CHALLENGES

**How: Handling of Cost Monitoring**

<table>
<thead>
<tr>
<th>STATE</th>
<th>SERVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>AWS/CE 710145618630 Amazon Elastic Compute Cloud - Compute</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/CE 710145618630 Amazon Elastic Load Balancing</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/CE 710145618630 Amazon Simple Storage Service</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/CE 710145618630 EC2 - Other</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/CE Summary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>STATUS DETAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2019-04-09) Unblended USD: 0.00</td>
</tr>
<tr>
<td>(2019-04-09) Unblended USD: 0.00</td>
</tr>
<tr>
<td>(2019-04-09) Unblended USD: 0.00</td>
</tr>
<tr>
<td>(2019-04-09) Unblended USD: 0.00</td>
</tr>
<tr>
<td>(2019-04-09) Total Unblended USD: 0.00</td>
</tr>
</tbody>
</table>

Cost Monitoring is the very next development for Azure
How much: Speaking of costs... monitoring costs?

- Unfortunately, monitoring cloud services is not free (at least with Amazon)
  - AWS charges 0.01 USD / 1000 API calls
  - Example: Cost for monitoring 300 AWS Cloudwatch metrics: \( \approx 2 \text{ USD/day} \)

- API Calls for Azure are not charged, but have a relatively strict rate limit

<table>
<thead>
<tr>
<th>STATE</th>
<th>SERVICE</th>
<th>STATUS DETAIL</th>
<th>CHECK PLUGIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Azure Agent Info</td>
<td>Remaining API reads: 11996, Monitored</td>
<td>azure_agent_info</td>
</tr>
<tr>
<td></td>
<td></td>
<td>groups: Glastonbury, Woodstock, 0 warnings, 0 exceptions</td>
<td></td>
</tr>
</tbody>
</table>
OUR APPROACH

Technical Concept

- Technical concept for monitoring cloud services is well-established
- **checkmk** dynamically creates *Piggyback Hosts*
- Data is piggybacked by AWS/Azure Host to these hosts
Multiple data sources used

**Internal Monitoring API**
AWS Cloudwatch

**Service APIs**
Directly from resource / service (e.g. EC2 instance)

**Global Services**
Log / Event services
Cost Explorer

* similar for Azure
Amazon Web Services
AWS MONITORING

Working on checks for the most important AWS services

<table>
<thead>
<tr>
<th>Compute</th>
<th>Storage</th>
<th>Database</th>
<th>Networking</th>
<th>Serverless</th>
<th>Mgmt Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2</td>
<td>S3</td>
<td>RDS</td>
<td>Elastic Load Balancing</td>
<td>Lambda</td>
<td>Cost Explorer</td>
</tr>
<tr>
<td>Elastic Beanstalk</td>
<td>EBS</td>
<td>DynamoDB</td>
<td>CloudFront</td>
<td>Elastic Container Service</td>
<td>CloudTrail</td>
</tr>
<tr>
<td></td>
<td>Glacier</td>
<td>Redshift</td>
<td></td>
<td>SNS</td>
<td>CloudWatch</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Fargate</td>
<td>API Gateway</td>
</tr>
</tbody>
</table>

Existing  Planned  Future
How it works: AWS Monitoring

AWS Resources...

- EC2 (Elastic Compute Cloud)
- EBS (Elastic Block Store) *usually together*
- ELB (Elastic Load Balancer)
- S3 (Simple Storage Service)
- RDS (Relational Database Service)
- CE (Cost Explorer)
AWS MONITORING
How it works: AWS Monitoring

AWS Resources... ... are available by region or globally...

- EC2 (Elastic Compute Cloud)
- EBS (Elastic Block Store) *usually together*
- ELB (Elastic Load Balancer)
- S3 (Simple Storage Service)
- RDS (Relational Database Service)
- CE (Cost Explorer)

per region (e.g. eu-west-1)
globally
AWS MONITORING

How it works: AWS Monitoring

AWS Resources... ... are available by region or globally... ... and can exist in multiple instances

- EC2 (Elastic Compute Cloud)
- EBS (Elastic Block Store) *usually together*
- ELB (Elastic Load Balancer)
- S3 (Simple Storage Service)
- RDS (Relational Database Service)
- CE (Cost Explorer)

per region (e.g. eu-west-1)

globally
AWS MONITORING

How it works: AWS Monitoring

AWS Resources and instances

- EC2
- EBS
- ELB
- S3
- RDS
- CE

... are treated in two different ways in checkmk

Piggyback Host
(1 per instance)

Piggyback Host
How it works: AWS Monitoring

AWS Resources and instances

- EC2
- EBS
- ELB
- S3
- RDS
- CE

... are treated in two different ways in checkmk

- Piggyback Host (1 per instance)
- Services

Summary, Limits, Cloudwatch metrics
AWS MONITORING

How it works: AWS Monitoring

AWS Resources and instances

EC2
EBS
ELB
S3
RDS
CE

... are treated in two different ways in checkmk

Piggyback Host (1 per instance)

Services

Services

Summary, Limits, CloudWatch metrics

Exception for EBS w/o EC2

Piggyback Host

Services

Services

Services

Costs & Usage
How it works: AWS Monitoring

AWS Resources and instances

- EC2
- EBS
- ELB
- S3
- RDS
- CE

... are treated in two different ways in checkmk

- Piggyback Host (1 per instance)
- Services

Exception for EBS w/o EC2

- Piggyback Host
- Services

Summary, Limits, Cloudwatch metrics

- But don’t services require a host in checkmk?

- Services

Costs & Usage

- Services
How it works: AWS Monitoring

AWS Resources and instances

- EC2
- EBS
- ELB
- S3
- RDS
- CE

... are treated in two different ways in checkmk

- Piggyback Host (1 per instance)
- Exception for EBS w/o EC2
- Piggyback Host

Summary, Limits, Cloudwatch metrics

Well, yes...

Costs & Usage
How it works: AWS Monitoring

AWS Resources and instances

- EC2
- EBS
- ELB
- S3
- RDS
- CE

... are treated in two different ways in checkmk

- Piggyback Host (1 per instance)
  - Services

- Exception for EBS w/o EC2
  - Services

- Piggyback Host
  - Services

- AWS Host
  - S3
  - RDS
  - CE
  - Services
  - Costs & Usage

Host role is fulfilled by AWS host itself

Summary, Limits, Cloudwatch metrics
AWS MONITORING

How it works: AWS Monitoring

AWS Account

- EC2
- EBS
- ELB
- S3
- RDS
- CE

checkmk

AWS Host

Data Source:
AWS Special Agent
AWS MONITORING

How it works: AWS Monitoring

AWS Account

- EC2
- EBS
- ELB
- S3
- RDS
- CE

checkmk

AWS Host

Data Source:
AWS Special Agent

Access Key ID
+ Secret Access Key
How it works: AWS Monitoring

AWS Account
- EC2
- EBS
- ELB
- S3
- RDS
- CE

checkmk

AWS Host
Data Source: AWS Special Agent

Access Key ID + Secret Access Key

EC2 Piggyback H. (1 per instance)

ELB Piggyback H.

data piggybacked to
<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>AWS/EBS Health vol-0566dcfcf23d9ab37c</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/EBS Summary</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/EC2 CPU Credits</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/EC2 CPU utilization</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/EC2 Disk IO Summary</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/EC2 Limits</td>
</tr>
<tr>
<td>OK</td>
<td>AWS/EC2 Network IO Summary</td>
</tr>
</tbody>
</table>
How it works: AWS Monitoring

AWS Account

- EC2
- EBS
- ELB
- S3
- RDS
- CE

AWS API

Access Key ID + Secret Access Key

Data Source: AWS Special Agent

checkmk

- EC2 Piggyback Host
  (1 per instance)
- AWS Host
  Exception if EBS w/o EC2
- ELB Piggyback Host
- AWS Host
  S3
  RDS
  CE

Actually the same host
data piggybacked to
AWS MONITORING

What it looks like: AWS Host
AWS MONITORING

How it works: AWS Monitoring

AWS Account

- EC2
- EBS
- ELB
- S3
- RDS
- CE

AWS API

Access Key ID + Secret Access Key

checkmk

AWS Host

Data Source: AWS Special Agent

EC2 Piggyback H.
(1 per instance)

AWS Host

Exception EBS w/o EC2

ELB Piggyback H.

AWS Host

S3

RDS

CE

Summary, Limits, Cloudwatch metrics

Costs & Usage

data piggybacked to
Microsoft Azure
AZURE MONITORING

We already developed several checks for Azure

### Selection of Azure services

<table>
<thead>
<tr>
<th>Compute &amp; Storage</th>
<th>Databases</th>
<th>Analytics &amp; IoT</th>
<th>Networking</th>
<th>Identity &amp; Security</th>
<th>Generic / Mgmt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Machines</td>
<td>Azure SQL DB</td>
<td>Event Hub</td>
<td>Load Balancer</td>
<td>Active Directory</td>
<td>Web Apps (Sites)</td>
</tr>
<tr>
<td>Blob Storage</td>
<td>SQL Data Warehouse</td>
<td>Data Factory</td>
<td>Virtual network</td>
<td>...</td>
<td>Backup</td>
</tr>
<tr>
<td>Functions</td>
<td>Azure Cosmos DB</td>
<td>IoT Hub</td>
<td>Azure DNS</td>
<td>...</td>
<td>Cost Management</td>
</tr>
<tr>
<td>Container Instances</td>
<td>Azure Cache</td>
<td>Stream Analytics</td>
<td>Network Watcher</td>
<td>...</td>
<td>Service Bus</td>
</tr>
</tbody>
</table>

Existing | Planned | Future
# AZURE MONITORING

**How it works: MS Azure Monitoring**

<table>
<thead>
<tr>
<th><strong>Microsoft Azure</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tenant / Directory</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>checkmk</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>tribe29</strong></th>
<th><strong>data piggybacked to</strong></th>
<th><strong>assigned to</strong></th>
<th><strong>Host</strong></th>
<th><strong>Service</strong></th>
</tr>
</thead>
</table>

32
AZURE MONITORING

How it works: MS Azure Monitoring

<table>
<thead>
<tr>
<th>Microsoft Azure</th>
<th>DB 1</th>
<th>DB 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tenant / Directory</td>
<td>ISS 1</td>
<td>ISS 2</td>
</tr>
<tr>
<td></td>
<td>VM 1</td>
<td>VM 2</td>
</tr>
</tbody>
</table>

checkmk

data piggybacked to assigned to

tribe29

Host

Service
How it works: MS Azure Monitoring

Resources must always be assigned

Microsoft Azure

Tenant / Directory

Resource Group 1

Resource Group 2

Resource Group 3

DB 1
DB 2
ISS 1
ISS 2
VM 1
VM 2

checkmk

tribe29

data piggybacked to

assigned to

Host

Service
AZURE MONITORING

How it works: MS Azure Monitoring

Microsoft Azure

Tenant / Directory

Resource Group 1
Resource Group 2
Resource Group 3

DB 1
DB 2
ISS 1
ISS 2
VM 1
VM 2

checkmk

agent_azure

Azure Host

Azure Agent Info

Host agent is assigned to

data piggybacked to

assigned to

Host

Service
How it works: MS Azure Monitoring

Microsoft Azure

Tenant / Directory

Resource Group 1

Resource Group 2

Resource Group 3

DB 1

DB 2

ISS 1

ISS 2

VM 1

VM 2

checkmk

agent_azure

Azure Host

Host “Group 1”

Host “Group 2”

Host “Group 3”

VM2

Data is “piggybacked”

data piggybacked to

assigned to

Host

Service
AZURE MONITORING

How it works: MS Azure Monitoring

Microsoft Azure

Tenant / Directory

Resource Group 1
Resource Group 2
Resource Group 3

DB 1
DB 2
ISS 1
ISS 2
VM 1
VM 2

checkmk

agent_azure

Azure Host

Azure Agent Info

Host “Group 1”
Host “Group 2”
Host “Group 3”
VM 2

Services visible on hosts

data piggybacked to
assigned to
Host
Service
ROADMAP
What we’re thinking about for the future

MORE
• More checks for more services
• Adding resource types based on customer demand & popularity

BETTER
• Improve simplicity & convenience, e.g.
  • Single Sign On
  • Pre-packaged monitoring config through Amazon Machine Image
  • One-click deployment of Azure monitoring with Microsoft Extension Manager
Thank you!