BASELINE MONITORING AND ANOMALY DETECTION

- all our microphones are muted
- ask your questions in Q&A, not in the Chat
- use Chat for discussion, networking or applause
- use the hashtag if you post something in your social media channels: #ZabbixMeetingOnline
ZABBIX 6.0 LTS

One of the new features of Zabbix 6.0 LTS is focus on anomaly detection

Zabbix 6.0 offers:

- Possibility to use **trends** to analyze large periods of data
- Baseline calculation using **baselinewma** and **baselinedev** functions
- Anomalous metric detection using **trendstl** function
BASELINE MONITORING
OVERVIEW
BASELINE AND ANOMALIES

Anomaly detection is a type of data analytics whose goal is detecting unusual patterns in a dataset:

- Data must be **normally distributed**, following a set of rules
- Anomaly detection measures how far a data point is away from the mean
- When the value deviates too much from the mean, it is considered to be anomalous
BASELINE VS FIXED TRIGGERS

BASELINE MONITORING CAN MONITOR VALUES WHICH FOLLOW A PATTERN

- Fixed trigger thresholds will either give false alarms or ignore the problem
- Baseline monitoring can adapt to such situation

This situation was "normal" a few weeks ago
ANOMALY DETECTION OVERVIEW

Anomaly detection is based on a set of statistic functions using:

- Standard deviation ($\sigma$)
- Mean absolute deviation (MAD)
- Weighted moving average (WMA)
- Seasonal and Trend decomposition using Loess (STL)
The **standard deviation** $\sigma$ defines how far the normal distribution is spread around the mean

- When a metric is normally distributed it follows some **interesting laws**:
  - 68% of all values fall between $[\text{mean}-\sigma, \text{mean}+\sigma]$
  - 95% of all values fall between $[\text{mean}-2\sigma, \text{mean}+2\sigma]$
  - 99.7% of all values fall between $[\text{mean}-3\sigma, \text{mean}+3\sigma]$
MEAN ABSOLUTE DEVIATION (MAD)

The **mean absolute deviation** (MAD) of a dataset is the average distance between each data point and the mean.

- Calculate the mean
- Calculate how far away each data point is from the mean using positive distances (deviations)
- Sum those deviations together
- Divide the sum by the number of data points

\[
\text{MAD} = \frac{\sum \left| x_i - \bar{x} \right|}{n}
\]
WEIGHTED MOVING AVERAGE ALGORITHM

A weighted moving average (WMA) puts more weight on recent data and less on past data

- The most recent data is more heavily weighted, and contributes more to the final WMA value
- The weighting factor used to calculate the WMA is determined by the period

For example, a 5 period WMA would be calculated as follows:

\[
WMA = \frac{P_1 \times 5 + P_2 \times 4 + P_3 \times 3 + P_4 \times 2 + P_5 \times 1}{5 + 4 + 3 + 2 + 1}
\]
TIMESHIFTS
TIMESHIFT SYNTAX

Zabbix can use absolute or relative timeshift to compare current and past periods of data

- Relative timeshift specifies time period relatively to the current time:

  \[ \text{trendavg}(/\text{host/key}, 1\text{d}: \text{now}-3\text{d}) \]

- Absolute timeshift specifies the time period for analysis:

  \[ \text{trendavg}(/\text{host/key}, 1\text{d}: \text{now}/\text{d}-3\text{d}) \]
Relative timeshift specifies sliding time period relatively to the current time:

```
trendavg(//key,1d: now-3d)
```

```
trendavg(//key,1d)
```

<table>
<thead>
<tr>
<th>Time</th>
<th>Date</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Three days ago</td>
<td>2022-04-24 00:00</td>
<td>Two days ago</td>
</tr>
<tr>
<td></td>
<td>2022-04-24 16:00</td>
<td></td>
</tr>
<tr>
<td>Yesterday</td>
<td>2022-04-26 00:00</td>
<td>Today</td>
</tr>
<tr>
<td></td>
<td>00:00:00</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Now</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Absolute timeshift specifies fixed time period calculated from the end of the period:

```
trendavg(/key,1d:now/d-3d)
trendavg(/key,1d:now/d+1d)
```
TREND FUNCTIONS
TREND FUNCTIONS

Zabbix 6.0 offers eight different trend functions for long-term data analysis

- Trends analysis:
  - trendsum (/host/key,time period:time shift)
  - trendavg (/host/key,time period:time shift)
  - trendcount (/host/key,time period:time shift)
  - trendmax (/host/key,time period:time shift)
  - trendmin (/host/key,time period:time shift)
  - trendstl (/host/key,eval period:time shift,detection period,season,<dev>,<devalg>,<s_window>)

- Baseline calculation:
  - baseline_dev (/host/key,data period:time shift,season_unit,num_seasons)
  - baseline_wma (/host/key,data period:time shift,season_unit,num_seasons)
STORING TRENDS

Trends in the trends cache are calculated in real-time independently from historical data

- TrendCache always has the actual trends value for every item
  - to calculate the new average after then nth number, you multiply the old average by n−1, add the new number, and divide the total by n.

$$\mu_n = \frac{(n - 1)\mu_{n-1} + x_n}{n}$$

- It is possible to store only trends data

* History storage period | Do not keep history | Storage period
* Trend storage period | Do not keep trends | Storage period | 720d
Zabbix trends are written to database at the beginning of each hour

- Trends for the current hour are unavailable for trend functions
- Two different Zabbix server internal caches are used by trends

### Option: TrendCacheSize

- Size of trend write cache, in bytes. Shared memory size for storing trends data.
- Range: 128K-2G
- Default: TrendCacheSize=512M

### Option: TrendFunctionCacheSize

- Size of trend function cache, in bytes.
- Shared memory size for caching calculated trend function data.
- Range: 128K-2G
- Default: TrendFunctionCacheSize=256M
TRENDS VS HISTORY

TRENDS UTILIZES LESS SPACE THAN HISTORY IN BOTH DATABASE AND MEMORY CACHES

- When functions with time-shift are used for analysis, all data between now and function period are stored in the memory cache
- History data could utilize gigabytes of data in such scenarios, trends are much more efficient

```sql
trendsum(/key,1d:now/d-3d)
```
BASELINE MONITORING
BASELINE MONITORING OVERVIEW

Baseline monitoring can be used to analyze recent data by:

- Comparing it to baseline from previous periods using the (baselinewma)
- Calculating the number of deviations from previous periods (baselinedev)

Previous data periods must be defined as seasons using:

- Season units (h, d, w, M, y) - cannot be smaller than data period
- Number of seasons

baseline*/(/host/key,1h:now/h,"d",3)

baseline function based on the last full hour within the last 3-day period
MORE BASELINE FUNCTION EXAMPLES

baseline*(/host/key,1d:now/d,"M",6)
baseline function based on the previous day and the same day of month in the previous 6 months. If the date does not exist in a previous month, last day of month will be used.

baseline*(/host/key,2h:now/h,"d",7)
baseline function based on the last two hours and the same hours within a 7-day period.

baseline*(/host/key,1w:now/w,"m",3)
baseline function based on the previous week and other weeks within a 3-month period.
BASELINEWMA() FUNCTION

Calculates baseline data by averaging data from the same timeframe in multiple equal time periods

- Weighted moving average algorithm (WMA) is used
- Baseline can be compared to recent trends data to detect anomalies

`baselinewma (host/key, data period:time shift, season_unit, num_seasons)`

`trendavg (Web/nginx.req, 1d:now/d) > baselinewma (Web/nginx.req, 1d, 4w) * 2`

Web requests yesterday (1d:now/d) is more than twice as high than baseline on the same weekdays (1d) over last 4 weeks
BASELINEDEV() FUNCTION

Calculates number of deviations ($\sigma$) between the last data period and periods in preceding seasons

- stddevpop algorithm is used (calculates standard deviation based on the entire population)
- High number of deviations indicates anomalies

`baselinedev (/host/key, data period:time shift, season_unit, num_seasons)`

`baselinedev (/Production server/system.cpu.load, 1h, 10d, 10) > 3`

Check if load for last hour is more than 3 deviations away from mean using 10 one-hour periods over last 10 days
ANOMALY DETECTION USING PATTERNS
TRENDSTL FUNCTION

STL will decompose data in predefined intervals and will find anomalies based on a repeating pattern.

- `trendstl()` function uses standard deviation to detect anomalies and returns **anomaly rate** (0 - 1):
  - Function compares smaller detection period to larger evaluation period
  - A standard deviation means how far values are from the average
  - By default, the MAD algorithm is used, can be also `stddevpop` or `stddevsamp`
  - The number of deviations can be specified (default is 3)
  - `s_window` is the span (in lags) of the loess window for seasonal extraction

`trendstl(/host/key, eval period, detection period, season, <dev>, <devalg>, <s_window>)`
Seasonal-Trend decomposition using LOESS (STL) is a robust method of time series decomposition

- The STL method uses locally fitted regression models to decompose a time series into
  - trend components
  - seasonal components
  - residual components
TRENDSTL FUNCTION EXAMPLE

trendstl(/host/net.if.out[eth0],30d:now/d,7d,12h) > 0.1

- Analyzing the last 30 days of trend data
- Find the anomalies rate for the previous 7 days of that period
- Expecting the periodicity to be 12h (traffic varies between day and night time)
- The number of deviations to count as anomaly equals 3 (default)
- MAD algorithm is used (default)
- Anomaly rate is larger than 0.1 (10% of all values)
Thank you