

Zabbix 7

Receptury

Nowe funkcje projektowania, wdrażania
i optymalizacji monitoringu infrastruktury IT

Wydanie III



Nathan Liefing



Brian van Baekel

Tytuł oryginału: Zabbix 7 IT Infrastructure Monitoring Cookbook: Explore the new features of Zabbix 7 for designing, building, and maintaining your Zabbix setup, 3rd Edition

Tłumaczenie: Krzysztof Sawka

ISBN: 978-83-289-2476-5

Copyright © Packt Publishing 2024. First published in the English language under the title 'Zabbix 7 IT Infrastructure Monitoring Cookbook - Third Edition – (9781801078320)'

Polish edition copyright © 2025 by Helion S.A.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

helion.pl/user/opinie/zab7r3

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 230 98 63

e-mail: helion@helion.pl

WWW: helion.pl (księgarnia internetowa, katalog książek)

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści |

O autorach	17
O recenzentach	18
Przedmowa	19
Wstęp	20
ROZDZIAŁ 1	
Instalacja Zabbixa i przygotowanie frontendu	25
Wymogi techniczne	25
Instalacja serwera Zabbix	26
Przygotowanie	26
Wykonanie	26
Wyjaśnienie	29
Konfiguracja frontendu Zabbix	30
Przygotowanie	30
Wykonanie	30
Wyjaśnienie	35
To nie wszystko	35
Skonfigurowanie wysokiej dostępności (HA) serwerów Zabbix	36
Przygotowanie	36
Wykonanie	37
Wyjaśnienie	45
To nie wszystko	47
Korzystanie z frontendu Zabbix	47
Przygotowanie	47
Wykonanie	48
Poruszanie się po frontendzie Zabbix	56
Przygotowanie	56
Wykonanie	56

ROZDZIAŁ 2**Pierwsze kroki w zarządzaniu użytkownikami Zabbixa 64**

Wymogi techniczne	64
Tworzenie grup użytkowników	65
Przygotowanie	65
Wykonanie	65
To nie wszystko	69
Korzystanie z ról użytkownika Zabbixa	70
Przygotowanie	70
Wykonanie	70
Wyjaśnienie	72
To nie wszystko	76
Tworzenie pierwszych użytkowników	76
Przygotowanie	77
Wykonanie	77
Uwierzytelnianie użytkowników za pomocą protokołu Azure AD SAML i aprowizacji użytkowników JIT	81
Przygotowanie	83
Wykonanie	84
Wyjaśnienie	90
To nie wszystko	91
Uwierzytelnianie użytkowników za pomocą protokołu OpenLDAP i aprowizacji użytkowników JIT	92
Przygotowanie	92
Wykonanie	92
Wyjaśnienie	96

ROZDZIAŁ 3**Konfiguracja monitorowania z użyciem Zabbixa 98**

Wymogi techniczne	98
Konfigurowanie agenta monitorującego	99
Przygotowanie	99
Wykonanie	99
Wyjaśnienie	105
Zobacz także	106

Tradycyjne korzystanie z monitorowania SNMP	106
Przygotowanie	107
Wykonanie	107
Wyjaśnienie	111
Nowoczesne korzystanie z monitorowania SNMP	112
Przygotowanie	112
Wykonanie	112
Wyjaśnienie	117
Tworzenie prostych sprawdzeń i pozycji przechwytywania	120
Przygotowanie	120
Wykonanie	120
Wyjaśnienie	125
Praca z pozycjami obliczonymi i zależnymi	127
Przygotowanie	127
Wykonanie	127
Wyjaśnienie	132
Tworzenie sprawdzeń zewnętrznych	133
Przygotowanie	134
Wykonanie	134
Wyjaśnienie	134
Konfigurowanie monitorowania JMX	135
Przygotowanie	136
Wykonanie	136
Wyjaśnienie	138
Zobacz także	138
Konfigurowanie monitorowania bazodanowego	139
Przygotowanie	139
Wykonanie	139
Wyjaśnienie	142
To nie wszystko	142
Konfigurowanie agenta monitorowania HTTP	143
Przygotowanie	143
Wykonanie	143
Wyjaśnienie	143
Symulowanie użytkownika internetowego za pomocą pozycji przeglądarkowych Zabbix	146
Przygotowanie	146
Wykonanie	146
Wyjaśnienie	151

Modyfikowanie wartości pozycji za pomocą wstępnego przetwarzania	155
Przygotowanie	156
Wykonanie	156
Wyjaśnienie	160
Zobacz także	161

ROZDZIAŁ 4

Praca z wyzwalaczami i alertami	162
Wymogi techniczne	162
Konfigurowanie wyzwalaczy	163
Przygotowanie	163
Wykonanie	163
Wyjaśnienie	167
To nie wszystko	173
Zobacz także	173
Konfigurowanie zaawansowanych wyzwalaczy	173
Przygotowanie	174
Wykonanie	174
Wyjaśnienie	177
To nie wszystko	181
Konfigurowanie alertów	182
Przygotowanie	182
Wykonanie	182
Wyjaśnienie	186
To nie wszystko	189
Utrzymywanie alertów w dobrym stanie	189
Przygotowanie	189
Wykonanie	190
Wyjaśnienie	192
To nie wszystko	193
Dostosowywanie alertów	193
Przygotowanie	193
Wykonanie	193
Wyjaśnienie	195

ROZDZIAŁ 5**Tworzenie własnych ustrukturyzowanych szablonów 196**

Wymogi techniczne	196
Tworzenie szablonów	197
Przygotowanie	197
Wykonanie	197
Wyjaśnienie	198
To nie wszystko	200
Konfigurowanie znaczników na poziomie szablonu	200
Przygotowanie	200
Wykonanie	201
Wyjaśnienie	202
Zobacz także	203
Tworzenie pozycji szablonu	203
Przygotowanie	203
Wykonanie	203
Wyjaśnienie	206
Zobacz także	207
Tworzenie wyzwalaczy szablonu	207
Przygotowanie	207
Wykonanie	207
Wyjaśnienie	208
Konfigurowanie różnych typów makr	209
Przygotowanie	209
Wykonanie	209
Wyjaśnienie	211
To nie wszystko	213
Stosowanie reguł LLD w szablonach	213
Przygotowanie	214
Wykonanie	214
Wyjaśnienie	218
Zobacz także	223
Zagnieżdżanie szablonów	223
Przygotowanie	223
Wykonanie	223
Wyjaśnienie	225

ROZDZIAŁ 6**Wizualizowanie danych, inwentarz i tworzenie raportów 227**

Wymogi techniczne	228
Uzyskiwanie dostępu do danych wizualnych za pomocą wykresów	228
Przygotowanie	228
Wykonanie	228
Wyjaśnienie	233
Pilnowanie infrastruktury za pomocą map	234
Przygotowanie	235
Wykonanie	235
Wyjaśnienie	240
Uzyskiwanie odpowiedniego zarysu za pomocą pulpitów	242
Przygotowanie	242
Wykonanie	242
Wyjaśnienie	252
To nie wszystko	252
Praca z szablonami pulpitów na poziomie hosta	253
Przygotowanie	253
Wykonanie	253
Wyjaśnienie	256
Konfigurowanie inwentarza Zabbix	257
Przygotowanie	257
Wykonanie	257
Wyjaśnienie	259
Korzystanie z widżetu Geomap	259
Przygotowanie	259
Wykonanie	259
Wyjaśnienie	262
Praca z funkcją tworzenia raportów	263
Przygotowanie	264
Wykonanie	264
Tworzenie zaplanowanych raportów PDF	269
Przygotowanie	269
Wykonanie	269
Wyjaśnienie	272

Konfigurowanie usprawnionego monitorowania usługi biznesowej	273
Przygotowanie	273
Wykonanie	273
Wyjaśnienie	283
To nie wszystko	286

ROZDZIAŁ 7

Automatyczne tworzenie za pomocą narzędzia wykrywającego

Wymogi techniczne	290
Konfigurowanie wykrywania agenta sieci Zabbix	290
Przygotowanie	290
Wykonanie	290
Wyjaśnienie	294
To nie wszystko	296
Praca z wykrywaniem sieci SNMP	296
Przygotowanie	296
Wykonanie	296
Wyjaśnienie	299
Automatyzacja tworzenia hostów za pomocą automatycznej rejestracji aktywnego agenta	301
Przygotowanie	301
Wykonanie	301
Wyjaśnienie	304
To nie wszystko	305
Korzystanie z funkcji wykrywania liczników wydajności systemu Windows	306
Przygotowanie	306
Wykonanie	306
Wyjaśnienie	310
Wykrywanie obiektów JMX	312
Przygotowanie	312
Wykonanie	312
Wyjaśnienie	315
To nie wszystko	316
Konfigurowanie wykrywania sieci Zabbix SNMP w nowy sposób	316
Przygotowanie	317
Wykonanie	317
Wyjaśnienie	321

Tworzenie hostów za pomocą technologii LLD i niestandardowych plików JSON	323
Przygotowanie	324
Wykonanie	324
Wyjaśnienie	328
To nie wszystko	330

ROZDZIAŁ 8

Konfiguracja serwerów proxy Zabbix	331
Wymogi techniczne	332
Konfigurowanie serwera proxy Zabbix	332
Przygotowanie	332
Wykonanie	332
Wyjaśnienie	334
To nie wszystko	335
Praca z pasywnymi serwerami proxy	335
Przygotowanie	335
Wykonanie	335
Wyjaśnienie	338
Praca z aktywnymi serwerami proxy	338
Przygotowanie	339
Wykonanie	339
Wyjaśnienie	341
Monitorowanie hostów za pomocą serwerów proxy	342
Przygotowanie	342
Wykonanie	342
Wyjaśnienie	344
To nie wszystko	346
Dodatkowe informacje	346
Szyfrowanie połączenia z serwerem proxy za pomocą współdzielonych kluczy	346
Przygotowanie	347
Wykonanie	347
Wyjaśnienie	349
Konfigurowanie równoważenia obciążenia z udziałem serwerów proxy	349
Przygotowanie	350

Wykonanie	350
Wyjaśnienie	352
Mechanizm wykrywania a serwery proxy	354
Przygotowanie	354
Wykonanie	355
Wyjaśnienie	357
Monitorowanie serwerów proxy	357
Przygotowanie	357
Wykonanie	358
Wyjaśnienie	362

ROZDZIAŁ 9

Integrowanie Zabbixa z usługami zewnętrznymi	365
Wymogi techniczne	365
Konfigurowanie wysyłania alertów Slack z Zabbixem	366
Przygotowanie	366
Wykonanie	366
Wyjaśnienie	377
Zobacz także	378
Konfigurowanie wysyłania alertów Microsoft Teams z Zabbixem	378
Przygotowanie	379
Wykonanie	379
Wyjaśnienie	384
Zobacz także	386
Używanie botów Telegram z Zabbixem	386
Przygotowanie	386
Wykonanie	387
Wyjaśnienie	395
To nie wszystko	395
Zobacz także	396
Integrowanie platformy Atlassian Opsgenie z Zabbixem	396
Przygotowanie	396
Wykonanie	396
Wyjaśnienie	403
To nie wszystko	404

ROZDZIAŁ 10**Rozszerzanie funkcjonalności Zabbixa za pomocą****własnych skryptów oraz interfejsu API Zabbixa 405**

Wymogi techniczne	406
Konfigurowanie tokenów API i zarządzanie nimi	406
Przygotowanie	406
Wykonanie	406
Wyjaśnienie	410
Rozszerzanie funkcjonalności Zabbixa za pomocą interfejsu API	411
Przygotowanie	411
Wykonanie	411
Wyjaśnienie	414
Zobacz także	415
Tworzenie jumphosta za pomocą interfejsu API Zabbixa i Pythona	415
Przygotowanie	416
Wykonanie	416
Wyjaśnienie	419
Zobacz także	420
Włączanie i wyłączanie hosta na mapach Zabbixa	420
Przygotowanie	420
Wykonanie	421
Wyjaśnienie	424
To nie wszystko	424
Zobacz także	426

ROZDZIAŁ 11**Utrzymywanie konfiguracji Zabbixa 427**

Wymogi techniczne	427
Konfigurowanie okresów konserwacji	428
Przygotowanie	428
Wykonanie	428
Wyjaśnienie	430
Tworzenie kopii zapasowej konfiguracji Zabbixa	430
Przygotowanie	431
Wykonanie	431
Wyjaśnienie	433
To nie wszystko	434

Modernizowanie backendu ze starszych wersji PHP	
do wersji od 8.2 wzwyż	434
Przygotowanie	435
Wykonanie	435
Wyjaśnienie	437
Modernizowanie bazy danych Zabbix ze starszych wersji MariaDB	
do wersji 10.11	438
Przygotowanie	438
Wykonanie	438
Wyjaśnienie	441
To nie wszystko	442
Modernizowanie konfiguracji Zabbixa	442
Przygotowanie	442
Wykonanie	443
Wyjaśnienie	446
Zobacz także	446
Utrzymywanie wydajności Zabbixa przez dłuższy czas	446
Przygotowanie	446
Wykonanie	447
Wyjaśnienie	451
To nie wszystko	455

ROZDZIAŁ 12

Zaawansowane zarządzanie bazami danych Zabbix	456
Wymogi techniczne	456
Konfigurowanie partycjonowania bazy danych MySQL	457
Przygotowanie	457
Wykonanie	457
Wyjaśnienie	464
Zobacz także	465
Korzystanie z możliwości bazy danych PostgreSQL TimescaleDB	465
Przygotowanie	465
Wykonanie	466
Wyjaśnienie	469
Zobacz także	470
Zabezpieczanie bazy danych MySQL Zabbix	470
Przygotowanie	471
Wykonanie	471
Wyjaśnienie	478

ROZDZIAŁ 13

Łączenie Zabbixa z usługami chmurowymi	480
Wymogi techniczne	480
Konfigurowanie monitorowania usługi AWS	481
Przygotowanie	481
Wykonanie	481
Wyjaśnienie	487
To nie wszystko	489
Konfigurowanie monitorowania usługi Microsoft Azure	489
Przygotowanie	489
Wykonanie	489
Wyjaśnienie	496
To nie wszystko	497
Przygotowanie Zabbixa do monitorowania Dockera	498
Przygotowanie	498
Wykonanie	498
Wyjaśnienie	499
To nie wszystko	500
Skorowidz	501

Praca z wyzwalaczami i alertami

Na co zdałoby się gromadzenie tych wszystkich danych w Zabbixie bez jakiegś formy powiadamiania administratora w przypadku pojawienia się nieprawidłowości? Możemy oczywiście używać Zabbixa jedynie do zbierania danych i przeglądania je ręcznie, ale staje się on znacznie przydatniejszy, gdy zaczniemy wysyłać powiadomienia do użytkowników. W ten sposób nie musimy cały czas siedzieć we frontendzie Zabbix, lecz możemy pozwolić wyzwalaczom i alertom wykonywać za nas ciężką pracę, a sami wchodzić do frontendu tylko wtedy, gdy to konieczne.

W wersji Zabbix 7 występuje nowa składnia wyrażeń wyzwalaczy w porównaniu do wydania 5. Składnia ta występuje już od wersji 5.4, dlatego jeżeli miałeś dłuższą przerwę, to może być Twój pierwszy kontakt z nią. Jeżeli miałeś do czynienia jedynie ze starszymi wydaniem Zabbixa, pamiętaj, że do nowej składni trzeba się przyzwyczaić. Począwszy od wersji 5.4, składnia ta wygląda tak samo w nowszych wydaniach.

Nauczymy się konfigurować skuteczne wyzwalacze za pomocą nowego formatu wyrażeń, a także korzystać z alertów w następujących recepturach:

- konfigurowanie wyzwalaczy,
- konfigurowanie zaawansowanych wyzwalaczy,
- konfigurowanie alertów,
- utrzymywanie alertów w dobrym stanie,
- dostosowywanie alertów.

Wymogi techniczne

W tym rozdziale będzie nam potrzebny serwer Zabbix, na przykład używany w poprzednim rozdziale:

- Serwer Zabbix zainstalowany w dowolnej dystrybucji Linuksa. Będziemy tu korzystać z serwera skonfigurowanego w rozdziale 1.
- Baza danych MariaDB dostosowana do pracy z serwerem Zabbix.

- Serwer NGINX lub Apache obsługujący frontend Zabbix.
- Będziemy również potrzebować hosta do monitorowania, na którym będziemy tworzyć interesujące wyzwalacze.

Konfigurowanie wyzwalaczy

Wyzwalacze (ang. *triggers*) stanowią ważny element Zabbixa, ponieważ dzięki nim dowiadujemy się, co się dzieje z naszymi danymi. Chcemy, żeby wyzwalacz się uruchamiał, gdy wartości danych przekraczają określony próg lub gdy otrzymujemy wyznaczoną wartość.

Zatem zacznijmy konfigurować ciekawe wyzwalacze. Istnieje mnóstwo różnych ustawień definiujących wyzwalacze, ale po przeczytaniu tej receptury powinieneś być w stanie samodzielnie tworzyć najistotniejsze wyzwalacze. Wprowadźmy Twoje doświadczenia z wyzwalaczami na wyższy poziom.

Przygotowanie

W tej recepturze będzie nam potrzebny gotowy serwer Zabbix, a także host Linuksa. Będę używał hosta `lar-book-agent_simple` z poprzedniego rozdziału, ponieważ mamy już tam przygotowane pewne pozycje.

Będziemy także potrzebować jeszcze jednego hosta monitorowanego przez agenta Zabbix z opracowanym szablonem. Wykorzystamy jedną z pozycji znajdujących się na tym hoście do stworzenia wyzwalacza. Będzie to mianowicie host `lar-book-agent_passive` z poprzedniego rozdziału.

Na tym hoście już znajdują się pewne gotowe wyzwalacze, ale rozwinimy je jeszcze bardziej, aby przekazywały nam dokładniejsze informacje.

Wykonanie

W tej sekcji utworzymy trzy wyzwalacze monitorujące zmiany stanu. Zacznijmy od pierwszego wyzwalacza.

Pierwszy wyzwalacz — monitorowanie usługi SSH

Stworzymy prosty wyzwalacz na hoście `lar-book-agent_simple`. Stworzyliśmy tu już proste sprawdzenie o nazwie `Check if port 22 is available`, ale nie mamy jeszcze żadnego mechanizmu powiadamiającego nas o dostępności portu.

1. Najpierw otwórz stronę *Data collection/Hosts*, następnie wybierz hosta i przejdź do zakładki *Triggers*. To tutaj znajdziesz istniejące wyzwalacze i będziesz tworzyć nowe. Aby utworzyć nowy wyzwalacz, kliknij przycisk *Create trigger* w prawym górnym rogu ekranu.

2. Stwórz nowy wyzwalacz za pomocą informacji zawartych na rysunku 4.1.

* Name: Service unreachable: Port 22 (SSH)

Event name: Service unreachable: Port 22 (SSH)

Operational data:

Severity: Not classified Information **Warning** Average High Disaster

* Expression: last(/lar-book-agent_simple/net.tcp.service[ssh,,22])=0 Add

Expression constructor

OK event generation: **Expression** Recovery expression None

PROBLEM event generation mode: **Single** Multiple

OK event closes: **All problems** All problems if tag values match

Allow manual close: ☐

URL:

Description:

Enabled: ☒

Add Cancel

Rysunek 4.1. Strona tworzenia wyzwalacza — usługa nieosiągalna

3. Kliknij przycisk *Add*, aby zakończyć tworzenie wyzwalacza. Otrzymasz wyzwalacz, który będzie uruchamiany, gdy port *SSH* (*Secure Shell*) stanie się niedostępny.
4. Przetestujmy jego działanie, otwierając wiersz poleceń na hoście, gdzie zablokujemy dostęp serwera Zabbix do portu 22. Dodamy regułę *iptables*, blokującą ruch przychodzący do portu 22 (SSH):

```
iptables -A INPUT -p tcp -i ens192 -s 10.16.16.152 --destination-port 22 -j DROP
```

5. Zmień interfejs *ens192* na swój własny, jak również podaj swój adres IP w miejsce adresu 10.16.16.152. Odpowiednie informacje uzyskasz po wpisaniu komendy:

```
ip addr
```

6. Teraz po kliknięciu kategorii *Dashboards* w pasku bocznym i odczekaniu krótkiej chwili powinieneś zobaczyć taki komunikat, jak na rysunku 4.2.

Time ▼	Info	Host	Problem • Severity
14:14:51	•	lar-book-agent_simple	Service unreachable: Port 22 (SSH)

Rysunek 4.2. Problemy wyświetlane na pulpicie — niedziałający port 22

Drugi wyzwalacz — wyzwalanie w momencie wypuszczenia nowej wersji Zabbixa

Podnieśmy teraz nieco poprzeczkę. W rozdziale 3., w recepturze „Konfigurowanie agenta monitorowania HTTP”, utworzyliśmy pozycję sprawdzającą na stronie internetowej Zabbixa najnowszą wersję oprogramowania. Teraz nawet sami dla siebie chcielibyśmy mieć oko na publikację nowej wersji Zabbixa:

1. Otwórz stronę *Data collection/Hosts* i wybierz hosta `lar-book-agent_simple`.
2. Otwórz teraz *Triggers* i kliknij przycisk *Create trigger*. Stwórz wyzwalacz z ustawieniami zaprezentowanymi na rysunku 4.3.

Trigger

Trigger

Tags

Dependencies

* Name

New Zabbix 7.0 release: {ITEM.VALUE}

Event name

New Zabbix 7.0 release: {ITEM.VALUE}

Operational data

Severity

Not classified

Information

Warning

Average

High

Disaster

* Problem expression

change(/lar-book-agent_simple/get.zabbix.7.latest)=1

Add

Expression constructor

OK event generation

Expression

Recovery expression

None

* Recovery expression

last(/lar-book-agent_simple/get.zabbix.7.latest,#1)=last(/lar-book-agent_simple/get.zabbix.7.latest,#5)

Add

Expression constructor

PROBLEM event generation mode

Single

Multiple

OK event closes

All problems

All problems if tag values match

Allow manual close

☐

Rysunek 4.3. Strona tworzenia wyzwalacza — wyzwalacz reagujący na wydanie nowej wersji Zabbixa 7

3. Kliknij przycisk *Add*, aby zakończyć proces tworzenia wyzwalacza.

Możesz nie ujrzeć w najbliższym czasie komunikatu wysyłanego przez ten wyzwalacz we frontendzie, ale omówię mechanizm jego działania w punkcie „Wyjaśnienie”.

Trzeci wyzwalacz — korzystanie z wielu pozycji w ramach jednego wyzwalacza

Do tej pory widzieliśmy wyzwalacze korzystające z jednej pozycji, ale możemy również stosować wiele pozycji w ramach jednego wyzwalacza. Zbudujmy nowy wyzwalacz, wykorzystujący wiele pozycji w jednym wyrażeniu:

1. Otwórz stronę *Data collection/Hosts* i wybierz hosta *lar-book-agent_passive*. Następnie przejdź do sekcji *Triggers* i kliknij przycisk *Create trigger*.
2. Utwórz wyzwalacz z ustawieniami takimi, jak na rysunku 4.4.

Rysunek 4.4. Strona tworzenia wyzwalacza — wyzwalacz reagujący na pakiety przychodzące oraz wychodzące

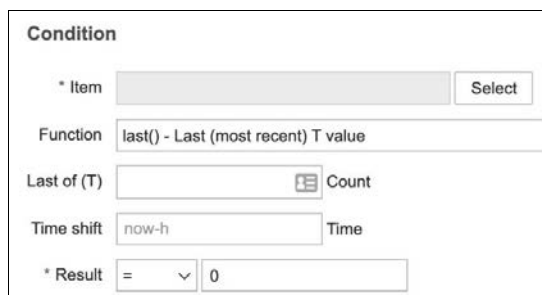
3. Pamiętaj, że u Ciebie może występować inna nazwa interfejsu. W moim przypadku nosi on nazwę *ens192*, dlatego musisz podmienić jego wystąpienia swoją własną nazwą. Za pomocą poniższego polecenia możesz sprawdzić nazwę interfejsu na Twoim gościu:

```
ip addr
```

4. Kliknij przycisk *Add*, aby zakończyć tworzenie wyzwalacza.

Wskazówka

Na stronie tworzenia wyzwalacza kliknij przycisk *Add* obok pola *Expression*, aby dodać warunek i ułatwić sobie tworzenie wyrażenia. Możemy na przykład użyć przycisku *Select*, aby wybrać pozycję z listy. Również bardzo przydatne jest krótkie objaśnienie każdej dostępnej funkcji wyzwalacza, wyświetlane podczas korzystania z listy rozwijanej *Function* (rysunek 4.5).



Rysunek 4.5. Strona tworzenia wyzwalacza

Tak wygląda proces tworzenia wyzwalacza korzystającego z dwóch pozycji.

Wyjaśnienie

Jeśli chcemy stworzyć sprawną platformę monitorowania, musimy dobrze zrozumieć mechanizm tworzenia oraz działania wyzwalaczy. Jest szczególnie ważne, aby prawidłowo konfigurować wyzwalacze, a także je testować po utworzeniu. Wyzwalacze są bardzo istotne, ponieważ odgrywają kluczową rolę w informowaniu nas o stanie monitorowanych elementów. Skonfiguruj wyzwalacze zbyt „luźno”, a pewne informacje będą Ci umykać. Skonfiguruj je zbyt „ściśle”, a będziesz zasypywany nadmiarem danych.

Istotność wyzwalacza jest definiowana przez poziom zagrożenia, co widać na rysunku 4.6.



Rysunek 4.6. Poziomy wyzwalacza w Zabbixie

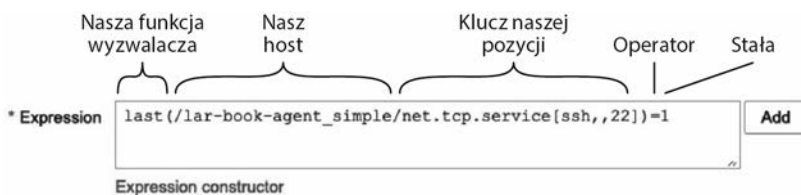
Poziomy te odgrywają ważną rolę w prawidłowym oznaczeniu istotności alertów. Możemy je także filtrować w różnych obszarach frontendu, a nawet w takich elementach, jak akcje.

Teraz wyjaśnimy sobie, dlaczego zbudowaliśmy wyzwalacze w ten, a nie inny sposób.

Pierwszy wyzwalacz — monitorowanie usługi SSH

Jest to bardzo prosty, ale skuteczny wyzwalacz. Gdy zostaje zwrócona wartość 1, oznaczająca UP (działająca usługa), lub 0 dla DOWN (nie działająca usługa), możemy z łatwością tworzyć tego typu wyzwalacze — nie tylko do monitorowania portów logicznych, lecz także każdego elementu powodującego prostą zmianę wartości, na przykład z 0 na 1 lub odwrotnie.

Przyjrzyjmy się teraz naszemu wyrażeniu (rysunek 4.7).



Rysunek 4.7. Wyrażenie wyzwalacza — port 22 (SSH)

Jak widać, wyrażenie składa się z pięciu członów:

- **Funkcja wyzwalacza.** Funkcja wyzwalacza stanowi tę składową wyrażenia, w której definiujemy nasze oczekiwania dotyczące wartości, na przykład tego, czy interesuje nas wyłącznie ostatnia wartość, czy średnia z określonego czasu.
- **Host.** Tutaj definiujemy hosta, wobec którego ma być używany wyzwalacz. W większości przypadków jest to host (lub szablon), na którym pracujemy.
- **Klucz pozycji.** Tutaj określamy klucz pozycji, za pomocą którego będziemy odczytywać wartość (wartości) z hosta i przekazywać je do funkcji wyzwalacza.
- **Operator.** Operator określa mechanizm działania funkcji w porównaniu na przykład do stałej albo innego wyrażenia. Dostępne są przeróżne operatory, co zostało zaprezentowane w tabeli 4.1.
- **Stała.** Jest to rzeczywiście stała (często wartość), za pomocą której wyzwalacz określa, czy powinien znajdować się w stanie OK lub PROBLEM. Możemy tu wstawiać również makra.

W naszym pierwszym wyzwalaczu zdefiniowaliśmy naszego hosta oraz pozycję określającą status usługi SSH. Przekazujemy przez funkcję wyzwalacza komunikat, że chcemy, aby wyzwalala go ostatnia wartość równa 0.

W przypadku tej pozycji oznaczałoby to uruchomienie wyzwalacza przed upływem minuty, ponieważ zdefiniowaliśmy ją tak jak na rysunku 4.8.

Jeśli spojrzymy na pole *Update interval* na stronie *Item*, to okaże się, że podczas tworzenia tego wyzwalacza oczekujemy wartości równej 0 oraz że wykrycie niedziałającego portu 22 zajmie maksymalnie minutę z powodu interwału równego 1m.

Tabela 4.1. Przykładowe operatory funkcji wyzwalacza

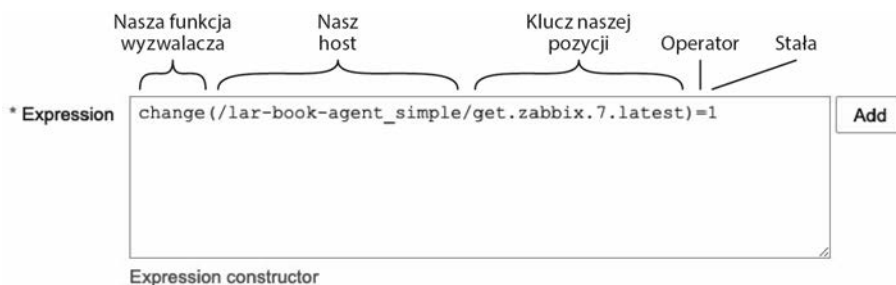
=	Równe
<>	Nie jest równe
>	Większe od
<	Mniejsze od
>=	Większe lub równe
<=	Mniejsze lub równe
+	Suma
-	Różnica
/	Iloraz
*	Iloczyn
and	Logiczna operacja AND. Używana na przykład do wyboru jednego i drugiego wyrażenia.
or	Logiczna operacja OR. Używana na przykład do wyboru jednego lub drugiego wyrażenia.
not	Logiczna operacja NOT. Używana na przykład do negowania wyrażenia.

Item	Tags	Preprocessing
* Name Check if port 22 is available		
Type Simple check		
* Key net.tcp.service[ssh,,22] Select		
Type of information Numeric (unsigned)		
Host interface 10.16.16.153:10050		
User name		
Password		
Units		
* Update interval 1m		

Rysunek 4.8. Strona konfigurowania pozycji — dostępność portu 22

Drugi wyzwalacz — wyzwalanie w momencie wypuszczenia nowej wersji Zabbixa

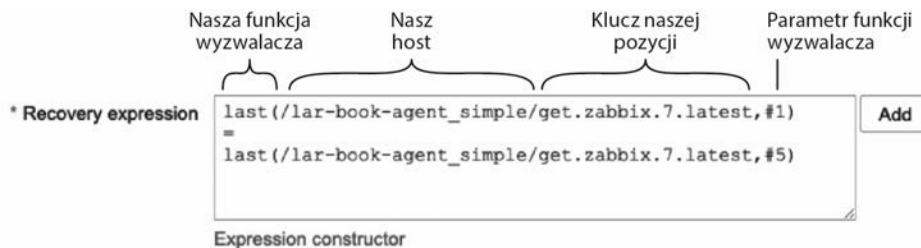
W drugim wyzwalaczu skorzystaliśmy z innego mechanizmu. Przygotowaliśmy wyrażenie nie tylko służące do uruchomienia wyzwalacza w przypadku spełnienia właściwych warunków, lecz także określające warunki potrzebne do przywrócenia stanu domyślnego wyzwalacza. W polu *Problem expression* definiujemy funkcję wyzwalacza, w której ostatnia wartość jest porównywana z wcześniejszą. Wykorzystaliśmy w tym celu funkcję `change` (rysunek 4.9).



Rysunek 4.9. Wyrażenie wyzwalacza — sprawdzanie HTTPS

Zatem nasz wyzwalacz będzie aktywowany w przypadku zmiany najnowszej wersji Zabbixa. Moglibyśmy sprawić, żeby wyzwalacz spełnił swoje zadanie, gdy tylko wartość bieżąca i poprzednia będą takie same, ale chcę, żeby jeszcze przez chwilę znajdował się w stanie `PROBLEM`.

Dlatego zdefiniowałem także **wyrażenie przywracania** (ang. *recovery expression*). Za jego pomocą sprawiam, że problem zostanie rozwiązany dopiero wtedy, gdy najnowsza oraz piąta w kolejności wartość będą takie same. Widać to na rysunku 4.10.



Rysunek 4.10. Inne wyrażenie wyzwalacza — sprawdzanie HTTPS za pomocą innej wartości

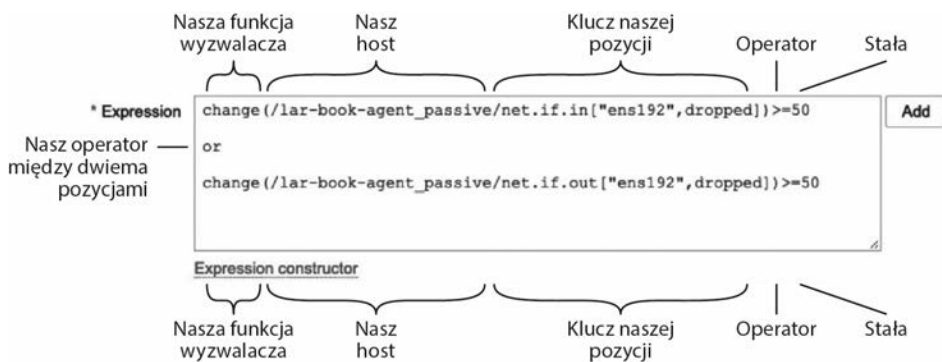
Wyrażenia przywracania bardzo się przydają, gdy chcesz mieć większą kontrolę nad warunkami powrotu wyzwalacza do stanu `OK`.

Wskazówka

Możemy używać wyrażeń przywracania do przedłużania stanu PROBLEM wyzwalacza ponad warunki zdefiniowane w polu *Problem expression*. W ten sposób będziemy wiedzieć, że wciąż znajdujemy się blisko stanu PROBLEM. Za pomocą wyrażenia przywracania określamy, że chcemy, aby wyzwalacz wrócił do stanu OK dopiero po osiągnięciu kolejnej wartości progowej wyznaczonej w tym wyrażeniu. Mechanizm ten działa poprzez oszacowanie obydwu wyrażeń, przy czym wyrażenie problemu musi uzyskać wartość FALSE, a wyrażenie przywracania — wartość TRUE.

Trzeci wyzwalacz — korzystanie z wielu pozycji w ramach jednego wyzwalacza

Trzeci wyzwalacz może wydawać się bardziej skomplikowany z powodu użycia dwóch pozycji, ale w zasadzie mamy do czynienia z taką samą strukturą (rysunek 4.11).



Rysunek 4.11. Wyrażenie wyzwalacza wykorzystujące wiele pozycji

Struktura naszego wyrażenia pozostaje identyczna, z takimi samymi pozycjami funkcji, hosta, klucza pozycji i wartości. Jednak podczas korzystania z wielu pozycji możemy wstawiać między nimi operator `or`. W ten sposób możemy tak zdefiniować wyzwalacz, żeby w celu przejścia w stan PROBLEM został spełniony warunek którejś z pozycji. W tym przypadku wyzwalacz zostanie aktywowany, gdy wartość którejś z pozycji przekroczy wyznaczony próg.

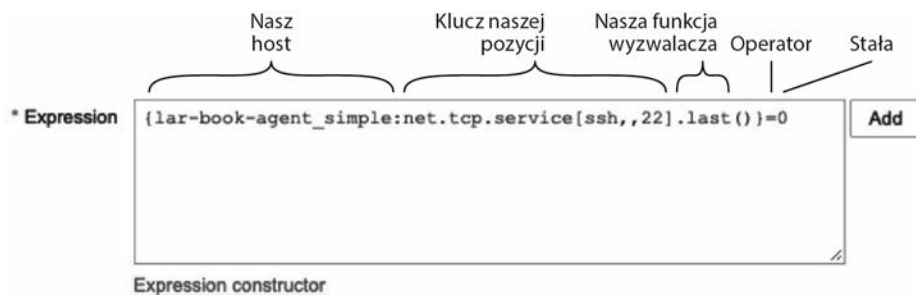
Ważna informacja

W wyrażeniu na rysunku 4.11 widzimy puste wiersze między poszczególnymi wyrażeniami pozycji. Zupełnie to nie przeszkadza w działaniu kodu, a wręcz poprawia czytelność. Korzystaj z tego mądrze podczas tworzenia wyzwalaczy.

Porównanie starej i nowej składni wyrażeń wyzwalaczy

Ten podpunkt może być szczególnie interesujący dla osób, które ostatni raz miały kontakt z Zabbixem sprzed wersji 5.4. Jak już wspomnieliśmy na początku rozdziału, wprowadzono wielką aktualizację wyrażeń w Zabbixie. Wyrażenia wyzwalacza działają teraz w odmienny sposób, taki sam jak w pozycjach obliczanych i innych miejscach, dzięki czemu ujednolicono składnię w całej platformie Zabbix.

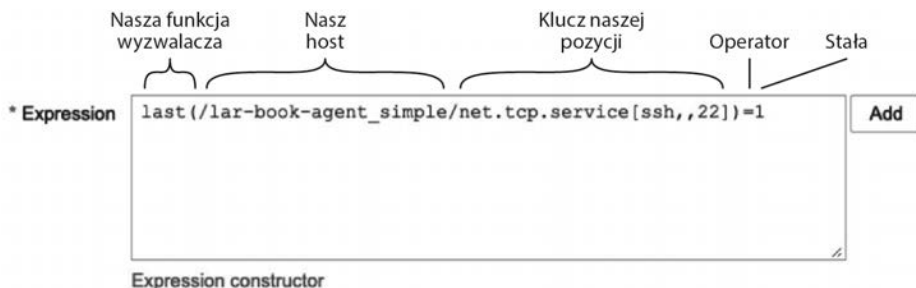
Spójrzmy na starą składnię, używaną do wersji Zabbix 5.2 (rysunek 4.12).



Rysunek 4.12. Stara składnia wyrażeń wyzwalacza

W starej składni wyrażenie zawsze rozpoczynało się od *nawiasu klamrowego*, po którym podawaliśmy nazwę hosta lub szablonu. Między nazwą hosta/szablonu a kluczem pozycji umieszczaliśmy *średnik*. Koniec klucza pozycji oznaczaliśmy *kropką*, ale w samym kluczu również mogą występować kropki. Po niej wstawialiśmy funkcję wyzwalacza i zamykaliśmy całość drugim *nawiasem klamrowym*. Na koniec umieszczaliśmy jeszcze operator i stałą porównywaną do wyrażenia.

Łatwo sobie wyobrazić, że takie wyrażenie szybko mogło stać się zagmatwane, zwłaszcza jeśli używaliśmy kropek w kluczach pozycji. Z kolei nowa składnia została zaprezentowana na rysunku 4.13.



Rysunek 4.13. Nowa składnia wyrażeń wyzwalacza

Nowa składnia wyzwalacza rozpoczyna się od razu funkcją wyzwalacza; od razu wiadomo, jakie jest zadanie tego wyrażenia. Po niej następują *nawias okrągły* i *ukośnik prawy*, po czym podajemy nazwę hosta lub szablonu. Kolejny *ukośnik prawy* służy do rozdzielenia nazwy hosta/szablonu od klucza pozycji. Zamykamy całość *nawiasem okrągłym*, po którym wstawiamy jeszcze operator i porównywaną wartość.

Dzięki umieszczeniu funkcji wyzwalacza na początku łatwo można stwierdzić, jakie zadanie jest wykonywane przez wyrażenie. Umieszczenie nazwy hosta/szablonu między nawiasami, a następnie oddzielenie ich ukośnikami prawymi od klucza pozycji poprawiają czytelność i spójność w trakcie tworzenia wyrażeń. Nie wprowadzamy już również mylących kropek. Zmiana składni jest bardzo dobrze przemyślana, chociaż przyzwyczajenie się do niej wymaga trochę czasu.

To dzięki takim szczegółom całe oprogramowanie daje wrażenie większego profesjonalizmu i okazuje się dobrze przemyślane. Zabbix naprawdę zyskał na tych zmianach.

To nie wszystko

Możemy nie tylko wybierać jedną z pozycji w wyrażeniu wyzwalacza, lecz także zastosować operację logiczną `and`. Dzięki temu możemy sprawić, że wyzwalacz przejdzie do stanu `PROBLEM` jedynie wtedy, gdy wiele pozycji uzyska określoną wartość progową. Wyzwalacze zapewniają pod tym względem olbrzymią swobodę i pozwalają nam szczegółowo określać własne kryteria. Nic tu nie jest predefiniowane — możemy umieszczać w naszych wyrażeniach dowolną liczbę instrukcji `and`, `not` czy `or`, a także funkcji wyzwalacza. Dostosowuj wyzwalacze do swoich potrzeb, a nagle okaże się, że będziesz mógł spać o wiele spokojniej, ponieważ będą Cię one informowały o wszelkich problemach.

Zobacz także

Więcej informacji o wyrażeniach wyzwalacza znajdziesz w dokumentacji Zabbixa. Zostały tam opisane funkcje pozwalające nam tworzyć idealnie dopasowane wyzwalacze. Szczegóły znajdziesz pod adresem <https://www.zabbix.com/documentation/current/en/manual/config/triggers/expression>.

Konfigurowanie zaawansowanych wyzwalaczy

Wyzwalacze w Zabbixie stają się coraz bardziej zaawansowane i czasami może być ciężko nadążyć za zmianami. Osoby przesiadające się z wersji Zabbix 5.2 lub starszej do wersji 7 natrafią nie tylko na nową składnię, lecz również na zupełnie nowy zestaw funkcji.

Sprawdźmy, jak się konfiguruje nieco bardziej zaawansowane wyzwalacze w Zabbixie 7.

Przygotowanie

W tej recepturze będzie nam potrzebny przygotowany serwer Zabbix, a także jeden host monitorowany przez agenta Zabbix z dołączonym szablonem. Wykorzystamy pozycje dostępne na tym hoście do tworzenia wyzwalaczy. Użyjemy tu hosta `lar-book-agent_passive` z poprzedniego rozdziału.

Jeżeli nie masz wspomnianego hosta, wystarczy przygotować nowego hosta z domyślnym szablonem pasywnego monitorowania Linuksa o nazwie *Linux by Zabbix agent*.

Poruszymy tu także bardziej zaawansowane zagadnienia, którymi zajmiemy się w dalszej części książki. Jeśli nie wiesz na przykład, jak korzystać z **wykrywania niskopoziomowego** (ang. *Low-Level Discovery* — LLD), warto najpierw zajrzeć do rozdziału 7., „Automatyczne tworzenie za pomocą narzędzia wykrywania”.

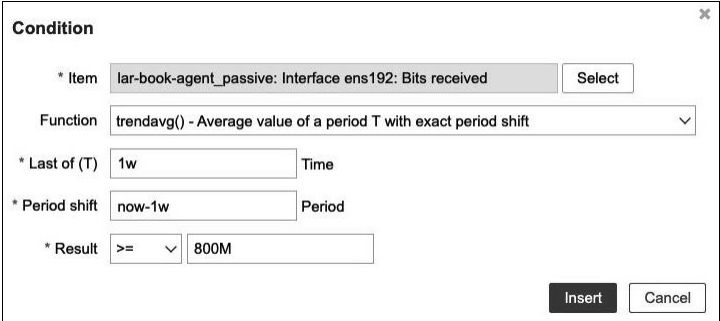
Wykonanie

Przyjrzyjmy się trzem *bardziej zaawansowanym* wyzwalaczom w porównaniu do omówionych w poprzedniej recepturze. Będą to: `trendavg`, analizujący tendencje w danych, `timeleft`, przewidujący przyszłe wartości, a także wyzwalacz realizujący **przesunięcie czasowe** (ang. *time shifting*) w celu porównywania danych z danymi przeszłymi.

Pierwszy zaawansowany wyzwalacz — funkcja `trendavg`

Najpierw zajmiemy się jednym z nowych elementów wyzwalaczy, mianowicie funkcją obliczania średniej tendencji:

1. Zaczniemy od utworzenia nowego wyzwalacza we frontendzie. Otwórz stronę *Data collection/Hosts* i wybierz hosta `lar-book-agent_passive`.
2. Przejdź do zakładki *Triggers* i kliknij przycisk *Create trigger* w prawym górnym rogu ekranu.
3. Kliknij biały przycisk *Add* obok pola *Expression*. Wypełnij dane zgodnie z rysunkiem 4.14.



The screenshot shows a 'Condition' dialog box in Zabbix. It contains the following fields and values:

- * Item:** `lar-book-agent_passive: Interface ens192: Bits received` (with a 'Select' button).
- Function:** `trendavg() - Average value of a period T with exact period shift` (dropdown menu).
- * Last of (T):** `1w` (Time).
- * Period shift:** `now-1w` (Period).
- * Result:** `>=` (dropdown) and `800M` (text input).

At the bottom right, there are 'Insert' and 'Cancel' buttons.

Rysunek 4.14. Konstruktor wyrażenia wyzwalacza `trendavg`

4. Kliknij przycisk *Insert* i podaj nazwę. Powinno to wyglądać tak, jak na rysunku 4.15.

The screenshot shows the Zabbix trigger configuration interface. At the top, there are three tabs: 'Trigger', 'Tags', and 'Dependencies'. The 'Trigger' tab is selected. Below the tabs, there are several input fields and a severity selector. The 'Name' field contains 'Average incoming interface usage last week >800Mbps'. The 'Event name' field also contains the same text. The 'Operational data' field is empty. The 'Severity' field has a dropdown menu with options: 'Not classified', 'Information', 'Warning', 'Average' (selected), 'High', and 'Disaster'. The 'Expression' field contains the text: `trendavg (/lar-book-agent_passive/net.if.in["ens192"],1w:now-1w)>=800M`. To the right of the 'Expression' field is an 'Add' button. Below the 'Expression' field, there is a link labeled 'Expression constructor'.

Rysunek 4.15. Wypełniony formularz wyzwalacza *trendavg*

5. Kliknij teraz niebieski przycisk *Add* u dołu ekranu, aby dokończyć tworzenie tego wyzwalacza.

To tyle w kwestii tworzenia tego wyzwalacza. W punkcie „Wyjaśnienie” znajdziesz więcej informacji na jego temat.

Drugi zaawansowany wyzwalacz — funkcja *timeleft*

Kolejną interesującą nas funkcją jest *timeleft*, która bardzo się przydaje w takich zadaniach, jak wykorzystanie powierzchni handlowej. Popatrzmy:

1. Utworzymy nowy wyzwalacz we frontendzie Zabbix. Otwórz stronę *Data collection/Hosts* i wybierz hosta *lar-book-agent_passive*.
2. Otwórz okno *Discovery rules* i kliknij przycisk *Trigger prototype* obok *Mounted filesystem discovery*.

Ważna informacja

W tym przypadku tworzymy prototyp wyzwalacza bezpośrednio na hoście za pomocą dostępnej reguły wykrywania szablonu. Jeżeli chcesz wprowadzić tego typu wyzwalacz do każdego hosta za pomocą szablonu, musisz utworzyć go na poziomie szablonu. Reguły wykrywania zostały wyjaśnione dokładniej w rozdziale 7., „Automatyczne tworzenie za pomocą narzędzia wykrywającego”.

3. Teraz kliknij *Create trigger prototype*.
4. Kliknij biały przycisk *Add* obok pola *Expression*. Podaj dane zgodne z rysunkiem 4.16.

Condition

* Item:

Function:

* Last of (T):

Time shift:

* Threshold:

Fit:

* Result:

Rysunek 4.16. Konstruktor wyrażenia wyzwalacza timeleft

Ważna informacja

Nie jest zalecane stosowanie krótkich przedziałów w wyzwalaczach predykcyjnych do przewidywania długich przedziałów czasu. Wybierz odpowiedni zestaw danych czasowych w odniesieniu do przewidywanego okresu.

5. Kliknij niebieski przycisk *Insert*. Ukończony wyzwalacz będzie wyglądał tak, jak na rysunku 4.17.

Trigger prototype Tags Dependencies

* Name: ⓘ

Event name:

Operational data:

Severity:

* Expression: ⓘ

Expression constructor

Rysunek 4.17. Wypełniony formularz wyzwalacza timeleft

6. Kliknij niebieski przycisk *Add* u dołu ekranu, aby zakończyć konfigurowanie wyzwalacza.

Właśnie zdefiniowaliśmy nowy wyzwalacz zawierający funkcję `timeleft`. Będzie nas powiadamiał, kiedy pozostanie nam tydzień do momentu, gdy skończy się miejsce na dyskach twardych. Więcej informacji na jego temat znajdziesz w punkcie „Wyjaśnienie”.

Trzeci zaawansowany wyzwalacz — przesunięcie czasowe za pomocą funkcji matematycznych

Na koniec zajmiemy się przesunięciem czasowym, a w tym przypadku zrobimy to w połączeniu z funkcją matematyczną. Będzie to nieco bardziej skomplikowany przykład, dlatego zachowaj cierpliwość:

1. Otwórz stronę *Data collection/Hosts* i wybierz naszego hosta `lar-book-agent_passive`.
2. Przejdź do okna *Triggers* i kliknij niebieski przycisk *Create trigger*.
3. Dodaj wyzwalacz widoczny na rysunku 4.18.

Trigger Tags Dependencies

* Name 20% more memory load than last week - average 1 hour

Event name 20% more memory load than last week - average 1 hour

Operational data

Severity Not classified Information Warning Average High Disaster

* Expression { avg (/lar-book-agent_passive/vm.memory.size[pavailable],1h:now-1w) - avg (/lar-book-agent_passive/vm.memory.size[pavailable],1h) } >20 Add

Expression constructor

Rysunek 4.18. Wypełniony formularz wyzwalacza średniego przesunięcia czasowego

Jest to wyzwalacz bardzo złożony w konfiguracji, dlatego omówimy go dokładnie w punkcie „Wyjaśnienie”.

Wyjaśnienie

Zaawansowane wyzwalacze bywają bardzo skomplikowane. Zaprezentowane tu przez nas to zaledwie czubek góry lodowej. Nie załamuj się ich złożonością, ponieważ znajdziesz w internecie mnóstwo dokumentacji pomagającej w ich skonfigurowaniu: <https://www.zabbix.com/documentation/current/en/manual/config/triggers>.

Opisanie wszystkich przypadków użycia w tej książce graniczyłoby z cudem, dlatego tak dobraliśmy omawiane tu wyzwalacze, żeby pokazać Ci dostępne możliwości. Korzystaj ze zdobytej tu wiedzy we własnych zastosowaniach, pamiętaj jednak, żeby każdy problem rozpatrywać indywidualnie.

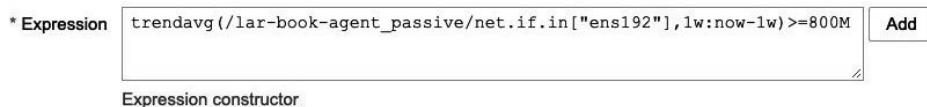
Pierwszy zaawansowany wyzwalacz — funkcja trendavg

Zacznijmy od wyrażenia zajmującego się średnią tendencji. Jest to jedna z niewielu funkcji wyzwalacza, w której wykorzystywane są dane tendencji, a nie historyczne. Przypomnijmy sobie pokrótce, czym są dane tendencji i dane historyczne w Zabbixie. Dane historyczne to dokładne wartości, które pozycja otrzymuje z monitorowanego hosta. Z kolei dane tendencji to wartości średnia, minimalna i maksymalna z ponad jednej godziny danych obliczonych na podstawie danych historycznych oraz liczby użytych wartości.

Spójrzmy, jakie mamy do dyspozycji funkcje wyzwalacza korzystające z danych tendencji:

- **trendavg.** Uzyskuje średnią wartość z danych tendencji dla określonego przedziału czasu.
- **trendmax.** Uzyskuje wartość maksymalną z danych tendencji dla określonego przedziału czasu.
- **trendmin.** Uzyskuje wartość minimalną z danych tendencji dla określonego przedziału czasu.
- **trendcount.** Uzyskuje liczbę wartości z danych tendencji dla określonego przedziału czasu.
- **trendsum.** Uzyskuje sumę danych tendencji dla określonego przedziału czasu.

Jak już wspomniałem, funkcje te wykorzystują dane tendencji. Wartości te są przechowywane w specjalnym obszarze pamięci Zabbixa, skąd mogą być wykorzystywane przez wyzwalacz. My użyliśmy funkcji trendavg. Przypomnijmy sobie, jak to wyglądało (rysunek 4.19).



Rysunek 4.19. Wyrażenie wyzwalacza trendavg

Najpierw podajemy funkcję trendavg, a następnie *nazwę hosta/szablону* i *klucz pozycji*, podobnie jak w poprzedniej recepturze. Nowością jest człon 1w:now-1w. Określamy tu przedział czasu; w tym przypadku chcemy skorzystać z wartości sprzed tygodnia.

Zgodnie z tym wyrażeniem, jeżeli średnia wartość tendencji sprzed tygodnia przekroczy 800 Mbps, to wyzwalacz przejdzie w stan PROBLEM.

Drugi zaawansowany wyzwalacz — funkcja `timeleft`

Jest to kolejna wielce interesująca funkcja wyzwalacza. Za pomocą funkcji `timeleft` możemy tworzyć wyzwalacze, które są aktywowane tylko wtedy, gdy przewidywane będzie przekroczenie jakiejś wartości progowej w przyszłości. Jest to tak zwany wyzwalacz predykcyjny, ponieważ stara się przewidywać przyszłość na podstawie wcześniejszych danych.

Przyjrzyjmy się ponownie temu wyrażeniu wyzwalacza (rysunek 4.20).

* Expression `timeleft(/lar-book-agent_passive/vfs.fs.size[{#FSNAME},pused],7h,100)<1w` Add

Rysunek 4.20. Wyrażenie wyzwalacza `timeleft`

Jak widać, początek wyrażenia jest standardowy: *funkcja wyzwalacza, nazwa hosta/szablonu oraz klucz pozycji*. W tym przypadku powiązaliśmy te elementy z przedziałem czasu, na podstawie którego wyzwalacz predykcyjny ma uzyskać przewidywania. Wartość `7h` oznacza dane historyczne z ostatnich siedmiu godzin. Dodaliśmy do tego wartość progową `100`, dzięki czemu wyzwalacz zostanie aktywowany, jeśli pojawi się oczekiwane maksymalne wykorzystanie pamięci dyskowej. Do pełni szczęścia potrzebujemy jeszcze jednego elementu, czyli spodziewanego rezultatu, w tym przypadku `<1w`.

Podsumowując: to wyrażenie wyzwalacza analizuje *siedem godzin* danych historycznych i jeżeli będzie przewidywać, że wykorzystanie pamięci osiągnie *100%* w ciągu *najbliższego tygodnia*, to wprowadzi wyzwalacz w stan `PROBLEM` i wyśle do Ciebie ostrzeżenie.

Wskazówka

Możesz połączyć funkcję wyzwalacza `timeleft` z innymi funkcjami, aby ograniczyć liczbę wysyłanych powiadomień. Na przykład w przypadku miejsca na dysku możemy przewidywać, że zacznie go brakować w ciągu tygodnia, ale być możesz chcesz się o tym dowiedzieć, gdy zostanie mniej niż 50 GB wolnej przestrzeni. Dodaj wyrażenie zaprezentowane na rysunku 4.21 i gotowe.

```
last(/Linux filesystems by Zabbix agent/vfs.fs.size[{#FSNAME},pused])>90%
and
timeleft(/Linux filesystems by Zabbix agent/vfs.fs.size[{#FSNAME},pused],1h,100)<1d
```

Rysunek 4.21. Wyrażenie funkcji wyzwalacza `timeleft`

Trzeci zaawansowany wyzwalacz — przesunięcie czasowe za pomocą funkcji matematycznych

Podczas zajęć dotyczących użytkownika Zabbixa zawsze muszę poświęcić więcej czasu z uczniami na omówienie wyrażeń wyzwalacza przesunięcia czasowego. Jest to logiczne, ponieważ mamy tu do czynienia z jednym z bardziej skomplikowanych wyrażeń, dodatkowo połączonym z funkcjami matematycznymi.

Przypomnijmy sobie to wyrażenie i rozbijmy je na czynniki pierwsze (rysunek 4.22).

```
1 {  
2   avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h:now-1w)  
3   -  
4   avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h)  
5 }  
6 >20
```

Rysunek 4.22. Wyrażenie wyzwalacza przesunięcia czasowego

Dla naszej wygody ponumerowałem poszczególne wiersze. Teraz możemy zająć się omówieniem każdego z nich po kolei:

1. Jest to nawias otwierający operację matematyczną, przy czym obydwie pozycje rozdziela operator.
2. Nasza pierwsza pozycja wykorzystująca funkcję przesunięcia czasowego. Pozycja ta będzie pozyskiwać wartość procentową dostępnej pamięci sprzed tygodnia, počawszy dokładnie od tego momentu. Jeżeli bieżący dzień to poniedziałek, 24 listopada, i jest godzina 14:00, to pozycja ta pozyska średnią z jednej godziny z poniedziałku, 17 listopada, między godziną 13:00 a 14:00.
3. Nasz operator matematyczny oznaczający różnicę. Będziemy odejmować wynik pierwszego wyrażenia od wyniku drugiego wyrażenia.
4. To nasza druga pozycja. Nie korzystamy w niej z przesunięcia czasowego. Będzie ona uzyskiwała średnią wartości z zeszłej godziny.
5. Nawias zamykający naszą operację matematyczną.
6. Na końcu pozostają operator i stała. W tym przypadku wyzwalacz będzie aktywowany tylko wtedy, gdy wynik operacji matematycznej będzie większy niż 20.

Skoro już wiemy, co się dzieje w poszczególnych wierszach, zastanówmy się nad mechanizmem działania całego wyrażenia. Wstawimy wartości własnoręcznie i sprawdzimy, czy dane wyrażenie uzyskuje wartość TRUE czy FALSE. Wartość TRUE oznacza, że pojawił się problem, a FALSE mówi nam, że wszystko jest w porządku. Zatem wygląda to tak:

(Zeszły tydzień – Bieżący tydzień) = Wynik
Jeśli Wynik jest większy niż 20, to wtedy wyrażenie ma wartość TRUE.
Wartość tego wyrażenia: TRUE/FALSE

Jeżeli w zeszłym tygodniu było 80% dostępnej pamięci, a w tym jest jej tylko 50%, to dzieje się rzecz następująca:

$(80 - 50) = 30$
Jeśli 30 jest większe niż 20, to wtedy wyrażenie ma wartość TRUE.
Wartość tego wyrażenia: TRUE

Spróbujmy jeszcze raz, ale tym razem dla 80% pamięci dostępnej w zeszłym tygodniu i 70% w tym tygodniu:

$(80 - 70) = 10$

Jeśli 10 jest większe niż 20, to wtedy wyrażenie ma wartość TRUE.

Wartość tego wyrażenia: FALSE

W taki właśnie sposób należy konfigurować wyrażenia przesunięcia czasowego. Możesz skorzystać z notatnika lub dowolnego innego narzędzia, rozpisać wyrażenie w prosty sposób, tak jak powyżej, i przeprowadzić obliczenia.

To nie wszystko

Wyrażenia wyzwalaczy można także testować w samym Zabbixie. Jeżeli otworzysz stronę *Data Collection/Hosts/Triggers* i wybierzesz któryś z zaawansowanych wyzwalaczy, to będziemy mogli przeprowadzić mały test. Na przykład dla wyzwalacza przesunięcia czasowego możemy otworzyć *Expression constructor* (rysunek 4.23).

Target	Expression	Action	Info
☒	A (avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h) - avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h:now-1w)) >20	Remove	

Rysunek 4.23. Konstruktor wyrażeń — wyzwalacz przesunięcia czasowego

Możemy tutaj kliknąć przycisk *Test* i wypełnić nasze wartości. Skorzystajmy z naszych przykładowych wartości 80% i 50% (rysunek 4.24).

Expression variable elements	Result type	Value
avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h)	Numeric (float)	80
avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h:now-1w)	Numeric (float)	50

Expression	Result	Error
A (avg(/lar-book-agent_passive/vm.memory.size[pavailable],1h) - avg(/lar-book-agent_passive/vm...	TRUE	
A	TRUE	

Rysunek 4.24. Konstruktor wyrażeń — wyzwalacz przesunięcia czasowego, test

Jak widać, dowiemy się z tego testu, czy nasze wyrażenie będzie miało wartość TRUE czy FALSE dla dowolnych wybranych przez nas wartości. Krótko mówiąc, jeśli chcesz się upewnić, że Twoje obliczenia na papierze będą zgodne z tymi w Zabbixie, przetestuj je za pomocą narzędzia *Expression constructor*.

Konfigurowanie alertów

Wysyłanie alertów stanowi bardzo ważny element architektury Zabbixa. Gdy konfigurowujemy alerty, chcemy, aby odbiorca dowiedział się, co się dzieje. Ważne jest także, żeby nie zalewać ludzi alertami; chcemy korzystać z nich skutecznie.

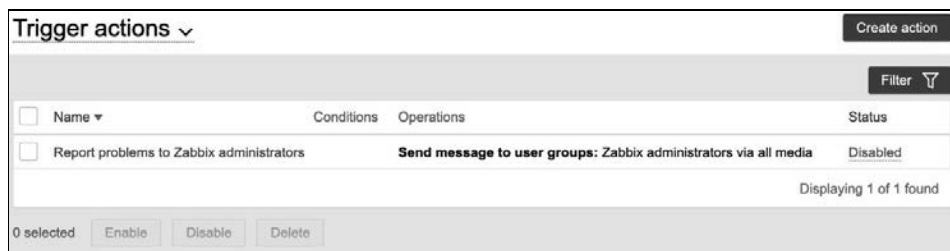
Dlatego w tej recepturze poznamy podstawy konfigurowania alertów, żeby od początku wiedzieć, jak należy robić to we właściwy sposób.

Przygotowanie

Przy tej recepturze będą nam potrzebne tylko dwie rzeczy. Alerty będziemy tworzyć z poziomu serwera Zabbix, a ponadto będziemy korzystać z wyzwalaczy, na przykład utworzonych w poprzedniej recepturze. Posłużą nam one do inicjowania procesu wysyłania alertów, żebyśmy mogli zobaczyć, w jaki sposób serwer Zabbix przekazuje te informacje.

Wykonanie

1. Zaczniemy od skonfigurowania akcji we frontendzie Zabbix. W tym celu otwieramy stronę *Alerts/Actions/Trigger actions*, gdzie powita nas ekran widoczny na rysunku 4.25.



Rysunek 4.25. Strona Trigger actions z jedną akcją wyzwalacza

Widzimy tu już jedną akcję skonfigurowaną do powiadamiania administratorów Zabbixa o problemach. W wersji Zabbix 7 wiele elementów, takich jak akcje czy media, jest ogólnie zdefiniowanych. W większości przypadków wystarczy je uaktywnić i uzupełnić informacjami.

2. Zdefiniujemy własną akcję, która będzie powiadamiać członków grupy *Zabbix administrators* o nowych wyzwalaczach. Kliknij niebieski przycisk *Create action*, aby zostało wyświetlone okno ukazane na rysunku 4.26.

Actions

Action Operations

* Name

Conditions

Label	Name	Action
Add		

Enabled ☒

* At least one operation must exist.

[Add](#) [Cancel](#)

Rysunek 4.26. Strona tworzenia akcji — powiadamianie czytelnika tej książki

3. Zaznacz tu opcję *Enabled*, aby uaktywnić tę akcję. Nadaj jej czytelną nazwę, dzięki czemu będziesz mógł szybko wybierać swoje akcje.
4. Przejdź teraz do zakładki *Operations* (rysunek 4.27).

New action

Action Operations

* Default operation step duration

Operations

Steps	Details	Start in	Duration	Action
Add				

Recovery operations

Details	Action
Add	

Update operations

Details	Action
Add	

Pause operations for symptom problems ☒

Pause operations for suppressed problems ☒

Notify about canceled escalations ☒

* At least one operation must exist.

[Add](#) [Cancel](#)

Rysunek 4.27. Strona tworzenia akcji — powiadamianie czytelnika tej książki, zakładka *Operations*

5. Domyślnie zakładka *Operations* jest pusta, dlatego musimy tu przygotować pewne operacje. Stworzymy tu trzy typy operacji — zaczniemy od operacji *Operations* (rysunek 4.28). Kliknij przycisk *Add*.

Operation details [X]

Operation: Send message

Steps: 1 - 1 (0 - infinitely)

Step duration: 0 (0 - use action default)

* At least one user or user group must be selected.

Send to user groups: Zabbix administrators Networking Select
type here to search

Send to users: type here to search Select

Send only to: - All -

Custom message: ☐

Label	Name	Action
Add		

Add Cancel

Rysunek 4.28. Strona ze szczegółami operacji *Operations* — powiadamianie czytelnika tej książki

6. Mamy tu możliwość dodawania użytkowników i/lub grup użytkowników, którzy mają dostawać alerty. Jeżeli postępowałeś zgodnie z instrukcjami zawartymi w rozdziale 1., „Instalacja Zabbixa i przygotowanie frontendu”, to możesz wybrać grupę *Networking*. W przeciwnym wypadku wystarczy również grupa *Zabbix administrators*.
7. Po kliknięciu niebieskiego przycisku *Add* u dołu formularza wrócimy na stronę *Actions*.
8. Teraz utworzymy następną operację, o nazwie *Recovery operations*. Tworzymy operację widoczną na rysunku 4.29.

Operation details [X]

Operation type: Notify all involved

Custom message: ☐

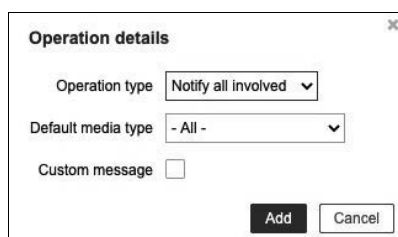
Add Cancel

Rysunek 4.29. Szczegóły operacji *Recovery operations* — powiadamianie czytelnika tej książki

9. Operacja ta będzie powiadamiała wszystkich użytkowników, którzy otrzymali alert pochodzący z pierwszej operacji. Wszyscy, którzy otrzymali powiadomienie o stanie PROBLEM, zostaną również powiadomieni o przywróceniu normalnego działania. Kliknij przycisk *Add* i kontynuujmy.

Jeżeli jesteś taki jak ja i zawsze chcesz być na bieżąco, możesz utworzyć powiadomienie z aktualizacją. W ten sposób dowiemy się na przykład, że ktoś już pracuje nad rozwiązaniem problemu. Zazwyczaj wybrałbym do takich alertów inny kanał informacyjny, na przykład wiadomości SMS do alertów o wysokim priorytecie, a kanał Slack czy Teams do pozostałych powiadomień.

10. Kliknij przycisk *Add* na stronie operacji *Update operations* i dodaj konfigurację widoczną na rysunku 4.30.



Rysunek 4.30. Szczegóły operacji *Update operations*
— powiadamianie czytelnika tej książki

11. Zrobimy to samo co w operacji *Recovery operations* i będziemy powiadamiać wszystkich wyznaczonych użytkowników o wszelkich postępach. Po kliknięciu przycisku *Add* kliknij kolejny niebieski przycisk *Add* na ekranie *Actions*, aby zakończyć tworzenie działania.
12. Teraz chcemy utworzyć typ mediów, za pomocą którego użytkownicy będą otrzymywać powiadomienia. Otwórz stronę *Alerts/Media types*, gdzie zobaczysz ekran widoczny na rysunku 4.31.

Jak widać, w wersji Zabbix 7 istnieje całkiem sporo gotowych typów mediów. Widzimy tu Slack, Opsgenie, a nawet Telegram. Zacznijmy jednak od kanału posiadanego niemal przez wszystkich: e-maila.
13. Kliknij typ mediów *Email* i skonfiguruj go do swoich potrzeb (rysunek 4.32).
14. Użyłem tu swoich ustawień z pakietu Office 365, ale powinien zadziałać każdy **protokół SMTP** (ang. *Simple Mail Transfer Protocol*). Wpisz swoje ustawienia SMTP, co powinno sprawić, że będziesz otrzymywać powiadomienia.
15. Sprawdź także następną zakładkę, *Message templates*. Na przykład szablon wiadomości dla generowanego zdarzenia *Problem* został zaprezentowany na rysunku 4.33.

Konfigurujemy go w taki sposób, aby otrzymywać wiadomości powiadamiające o sytuacji. Można go całkowicie modyfikować, aby przekazywał taki zakres informacji, jaki uznamy za słuszny.

☐ Name ▲	Type	Status
☐ Discord	Webhook	Enabled
☐ Email	Email	Enabled
☐ Email (HTML)	Email	Enabled
☐ Jira	Webhook	Enabled
☐ Jira ServiceDesk	Webhook	Enabled
☐ Jira with CustomFields	Webhook	Enabled
☐ Mattermost	Webhook	Enabled
☐ MS Teams	Webhook	Enabled
☐ Opsgenie	Webhook	Enabled
☐ PagerDuty	Webhook	Enabled
☐ Pushover	Webhook	Enabled
☐ Redmine	Webhook	Enabled
☐ ServiceNow	Webhook	Enabled
☐ SIGNAL4	Webhook	Enabled
☐ Slack	Webhook	Enabled
☐ SMS	SMS	Enabled
☐ Telegram	Webhook	Enabled
☐ Zammad	Webhook	Enabled
☐ Zendesk	Webhook	Enabled

Rysunek 4.31. Strona Media types z gotowymi typami mediów

16. Na razie zostawmy ustawienia domyślne. Przejdźmy jeszcze na stronę *Users/Users* i edytujmy użytkownika w grupie *Zabbix administrators* lub *Networking*. W tym przykładzie użyję konta *Admin* (rysunek 4.34).
17. Przejdź do zakładki *Media* i kliknij przycisk *Add*. Chcemy dodać informacje widoczne na rysunku 4.35, żeby otrzymywać na adres e-mail powiadomienia o wszystkich poziomach wyzwalacza.
18. Kliknij teraz niebieski przycisk *Add*, by zakończyć tworzenie mediów dla tego użytkownika.

Wyjaśnienie

Tak właśnie konfigurujemy alerty. Będziesz otrzymywać powiadomienia na adres e-mail zgodnie ze schematem zaprezentowanym na rysunku 4.36.

New media type

Media type Message templates Options

* Name

Type

Email provider

* SMTP server

SMTP server port

* Email

SMTP helo

Connection security **STARTTLS**

SSL verify peer ☐

SSL verify host ☐

Authentication **Username and password**

Username

Password

Message format **Plain text**

Description

Enabled ☒

Rysunek 4.32. Strona konfigurowania kanału Email

Message template

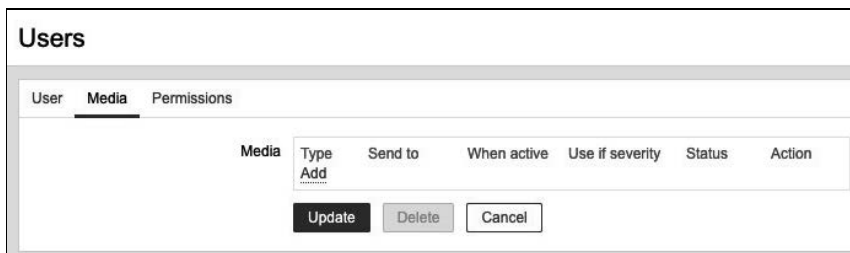
Message type

Subject

Message

Problem started at {EVENT.TIME} on {EVENT.DATE}
Problem name: {EVENT.NAME}
Host: {HOST.NAME}
Severity: {EVENT.SEVERITY}
Operational data: {EVENT.OPDATA}
Original problem ID: {EVENT.ID}
{TRIGGER.URL}

Rysunek 4.33. Domyślny szablon wiadomości informującej o problemach



The screenshot shows the 'Users' page in Zabbix with the 'Media' tab selected. The page has three sub-tabs: 'User', 'Media', and 'Permissions'. The 'Media' tab contains a table with columns: 'Type', 'Send to', 'When active', 'Use if severity', 'Status', and 'Action'. Below the table are three buttons: 'Update', 'Delete', and 'Cancel'.

Rysunek 4.34. Zakładka Media użytkownika Admin



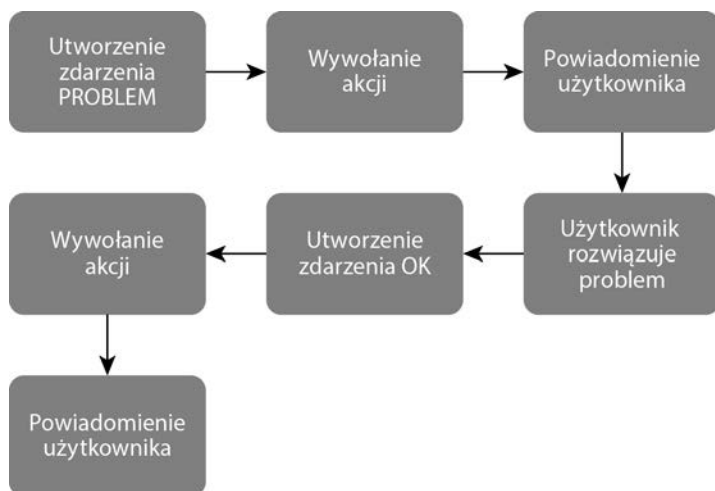
The screenshot shows the 'Media' configuration dialog for user Admin. It has a close button (X) in the top right corner. The 'Type' is set to 'Email'. The '* Send to' field contains 'nathan@oicsts.nl' with a 'Remove' button next to it. Below this is an 'Add' button. The '* When active' field contains '1-7,00:00-24:00'. The 'Use if severity' section has six checkboxes, all of which are checked: 'Not classified', 'Information', 'Warning', 'Average', 'High', and 'Disaster'. The 'Enabled' checkbox is also checked. At the bottom right are 'Add' and 'Cancel' buttons.

Rysunek 4.35. Strona konfigurowania mediów dla użytkownika Admin

Gdy coś się psuje, nasza konfiguracja wyzwalacza wywołuje w Zabbixie stan **PROBLEM**. Zdarzenie to wywoła **akcję** (ang. *action*) polegającą na wykorzystaniu danego typu mediów i konfiguracji mediów użytkownika do powiadomienia naszego użytkownika. Użytkownik następnie rozwiązuje problem (na przykład uruchamia ponownie serwer), po czym zostaje wygenerowane zdarzenie OK. Zostaje ponownie wywołana akcja, która przesyła odpowiedni komunikat do użytkowników.

Wskazówka

Przed skonfigurowaniem takich alertów przygotuj najpierw dla siebie samego taki przepływ informacji jak na rysunku 4.36 i tylko wyznacz grupy oraz użytkowników, którzy mają otrzymywać powiadomienia. W ten sposób będziesz mógł się przekonać, czy system wysyłania alertów spełnia Twoje wymagania.



Rysunek 4.36. Schemat ukazujący przepływ informacji o problemie w Zabbixie

To nie wszystko

Istnieje mnóstwo typów mediów i sposobów ich integracji, a my zaledwie ruszyliśmy wierzchołek góry lodowej, przeglądając listę gotowych rozwiązań. Przejrzyj listę gotowych sposobów integracji na stronie <https://www.zabbix.com/integrations> albo stwórz własne rozwiązanie za pomocą skryptów webhook Zabbixa i innych rozszerzeń.

Utrzymywanie alertów w dobrym stanie

Ważne jest utrzymywanie alertów w dobrym stanie, aby użytkownicy nie otrzymywali ani za dużo, ani za mało powiadomień. W tym celu zmienimy wyzwalacz i typ mediów *Email* na coś odpowiedniejszego.

Przygotowanie

Będziemy używać pierwszego wyzwalacza z pierwszego przepisu, a także domyślnego typu mediów e-mail w Zabbixie.

Poza tym, oczywiście, będzie nam potrzebny serwer Zabbix.

Wykonanie

Przygotujemy efektywne alerty w następujący sposób:

1. Zaczniemy od naszego pierwszego wyzwalacza, utworzonego w recepturze „Konfigurowanie wyzwalaczy”. Przejdź do hosta `lar-agent-simple` na stronie *Data collection/Hosts* i kliknij jego zakładkę *Trigger*.
2. Niektórzy podają tutaj inną nazwę wyzwalacza, jak widać na rysunku 4.37.

Rysunek 4.37. Pierwszy wyzwalacz z pierwszej receptury w tym rozdziale

Nawet po użyciu w wyzwalaczu makra `{HOST.NAME}` wciąż pozostaje on prosty, dlatego na szczęście nie trzeba tu wiele zmieniać. Jeżeli użyłś nazwy hosta w nazwie wyzwalacza, możesz ją zmienić, żeby komunikat był czytelniejszy.

3. Podaj krótką i opisową nazwę wyzwalacza, na przykład tak jak na rysunku 4.38.

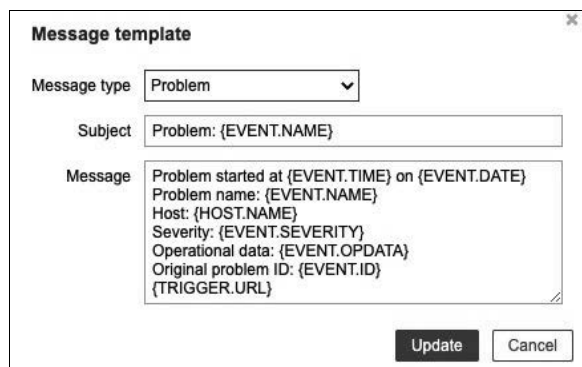
* Name

Rysunek 4.38. Nowa nazwa pierwszego wyzwalacza

4. Przejdź teraz do zakładki *Tags* i dodaj znacznik zapewniający uporządkowane nazewnictwo wyzwalaczy. Może wyglądać to tak, jak na rysunku 4.39.

Rysunek 4.39. Znaczniki pierwszego wyzwalacza

5. Innym doskonałym sposobem utrzymywania porządku jest zmiana komunikatu w mediach. Zmieńmy go na lepiej odpowiadający naszym potrzebom. Otwórz stronę *Alerts/Media types* i wybierz nasz typ mediów *Email*.
6. Wybierz *Message templates* i kliknij przycisk *Edit* obok naszego pierwszego problemu. Zostanie wyświetlone okno widoczne na rysunku 4.40.



The screenshot shows a 'Message template' dialog box with a close button (X) in the top right corner. It contains the following fields:

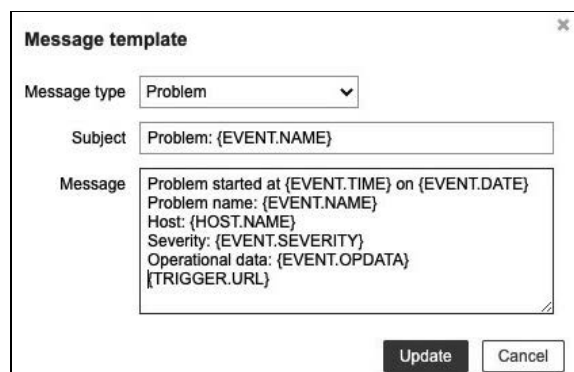
- Message type:** A dropdown menu with 'Problem' selected.
- Subject:** A text field containing 'Problem: {EVENT.NAME}'.
- Message:** A text area containing the following text:
Problem started at {EVENT.TIME} on {EVENT.DATE}
Problem name: {EVENT.NAME}
Host: {HOST.NAME}
Severity: {EVENT.SEVERITY}
Operational data: {EVENT.OPDATA}
Original problem ID: {EVENT.ID}
{TRIGGER.URL}

At the bottom right, there are two buttons: 'Update' and 'Cancel'.

Rysunek 4.40. Domyślna wiadomość typu mediów Email

Obecnie w Zabbixie wysyłany jest domyślny komunikat. Jeżeli jednak chcemy to zmienić, możemy utworzyć własną wiadomość. Domyślny komunikat został zaprezentowany na rysunku 4.40.

7. Możemy zmienić wiadomość w typie mediów. Na przykład jeśli nie chcesz wysyłać oryginalnego identyfikatora problemu lub potrzebujesz bardziej zmodyfikowanego powiadomienia, wystarczy usunąć odpowiedni wiersz, co ukazano na rysunku 4.41.



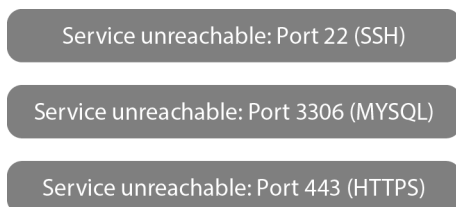
This screenshot is identical to the one in Rysunek 4.40, showing the 'Message template' dialog box with the default template. The text in the 'Message' field is: 'Problem started at {EVENT.TIME} on {EVENT.DATE}', 'Problem name: {EVENT.NAME}', 'Host: {HOST.NAME}', 'Severity: {EVENT.SEVERITY}', 'Operational data: {EVENT.OPDATA}', 'Original problem ID: {EVENT.ID}', and '{TRIGGER.URL}'.

Rysunek 4.41. Niestandardowa wiadomość typu mediów Email

Wyjaśnienie

W tej recepturze wykonaliśmy dwie czynności: zmieniliśmy nazwę wyzwalacza i dodaliśmy do niego znacznik.

Przemyślane, uporządkowane nazewnictwo wyzwalaczy jest istotne dla utrzymania dobrej organizacji środowiska Zabbix. Zmieniliśmy nazwę Port 22 SSH down on {HOST.NAME} po to, aby zapewnić naszej architekturze standaryzację, dzięki czemu możemy tworzyć użyteczne struktury, na przykład za pomocą przyszłych wyzwalaczy (rysunek 4.42).



Rysunek 4.42. Schemat struktury utworzonej z wyzwalaczy

Nasze wyzwalacze są jasno opisane i od razu dowiadujemy się z nich, który host, port i usługa nie działają.

Ponadto dodaliśmy znacznik do niedziałającej usługi, dzięki czemu usługa ta będzie teraz wyraźnie wyświetlana oraz zostaniemy poinformowani o sytuacji (rysunek 4.43).

Host	Problem	Duration	Ack	Actions	Tags
lar-book-agent_simple	Service unreachable: Port 22 (SSH)	1s	No		component: tcp services scope: availability

Rysunek 4.43. Uporządkowany wyzwalacz informujący o niedziałającej usłudze

W wersji Zabbix 6 wprowadzono nowe zasady tworzenia znaczników. Spełniliśmy wymogi tych nowych standardów, najpierw tworząc używaną w wyzwalaczu pozycję ze znacznikiem składowej (*component*), a przed chwilą dodając również znacznik zakresu (*scope*). Jeżeli spojrzymy na rysunek 4.43, od razu zauważymy, że problem dotyczy dostępności usługi SSH. W zakresie *scope* zazwyczaj umieszczamy jedną z pięciu wartości: *availability* (dostępność), *performance* (wydajność), *notification* (powiadomienie), *security* (zabezpieczenia) i *capacity* (pojemność).

Więcej informacji na temat tych nowych zasad tworzenia znaczników znajdziesz w dokumentacji:

<https://blog.zabbix.com/tags-in-zabbix-6-0-lts-usage-subfiltersand-guidelines/19565/>

Usunęliśmy również makro {HOST.NAME}, jeśli korzystaliśmy z niego wcześniej. Wiemy z kolumny *Host*, który host uaktywnił wyzwalacz, dlatego nie potrzebujemy tego makra.

Nazwy wyzwalaczy powinny być krótkie oraz czytelne, a makra nazwy hosta powinniśmy wstawiać w sekcji *Media* lub używać jedynie pola dostępnego we frontendzie.

Zmieniliśmy tu również akcję. Modyfikowanie komunikatów wysyłanych przez różne typy mediów stanowi doskonały sposób utrzymywania porządku w naszych kanałach nadawczych. Czasami chcemy otrzymywać mniej lub więcej informacji przez określone kanały, a jednym ze sposobów osiągnięcia tego jest zmiana struktury wysyłanych komunikatów.

Możemy również tworzyć niestandardowe wiadomości z poziomu *Action*, co powoduje zmianę wszystkich komunikatów wysyłanych wybranymi kanałami.

To nie wszystko

W tej recepturze próbuję wyjaśnić, że chociaż łatwo jest skonfigurować środowisko Zabbix, to o wiele trudniej przygotować dobrą architekturę monitorowania w ramach Zabbixa — dotyczy to właściwie każdej platformy monitorowania — bez odpowiedniej dozy planowania. Ostrożnie zaplanuj strukturę wyzwalaczy, zanim zaczniesz tworzyć strukturę systemu monitorowania w Zabbixie.

Inżynier pracujący na ustrukturyzowanych danych, który poświęci czas na stworzenie przemyślanej struktury monitorowania, zaoszczędzi w przyszłości mnóstwo czasu, ponieważ pozna istotę problemu przed innymi.

Dostosowywanie alertów

Wysyłanie alertów to bardzo przydatna funkcja, pozwalająca zachować porządek w strukturach platformy monitorującej, zwłaszcza w połączeniu z dotychczas opisanymi w tej książce sztuczkami. Czasami jednak potrzebujemy od alertów możliwości wykraczających poza gotowe rozwiązania.

W tej recepturze odrobinę dostosujemy alerty do naszych potrzeb.

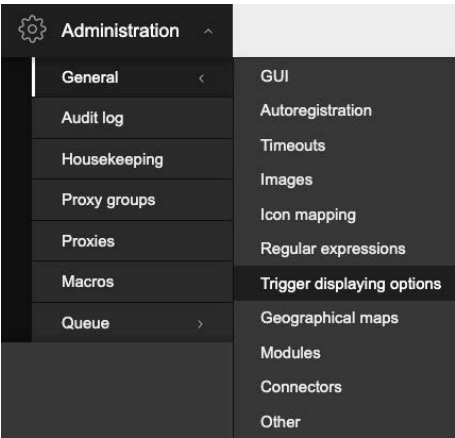
Przygotowanie

W tej recepturze będzie nam potrzebny jedynie nasz serwer Zabbix.

Wykonanie

Aby dostosowywać alerty, należy wykonać następujące czynności:

1. Stwórzmy niestandardowe poziomy wyzwalacza, odzwierciedlające potrzeby naszej organizacji. Otwórz stronę *Administration/General* i wybierz *Trigger displaying options* w bocznym menu (rysunek 4.44).



Rysunek 4.44. Opcje Administration/General w bocznym menu

Po kliknięciu przejdziemy na nową stronę. Znajdziemy tam domyślne poziomy wyzwalacza, co widać na rysunku 4.45.

Use custom event status colors ☐

* Unacknowledged PROBLEM events

blinking

* Acknowledged PROBLEM events

blinking

* Unacknowledged RESOLVED events

blinking

* Acknowledged RESOLVED events

blinking

* Display OK triggers for

5m

* On status change triggers blink for

2m

* Not classified

Not classified

* Information

Information

* Warning

Warning

* Average

Average

* High

High

* Disaster

Disaster

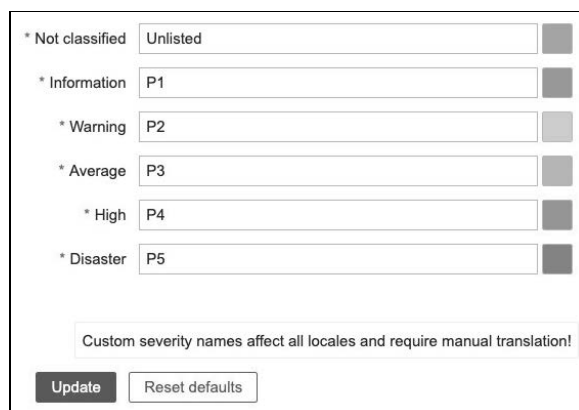
Custom severity names affect all locales and require manual translation!

Update

Reset defaults

Rysunek 4.45. Strona z domyślnymi poziomami wyzwalacza

2. Możemy tu zmodyfikować te domyślne poziomy wyzwalacza, na przykład tak jak na rysunku 4.46.



* Not classified	Unlisted	
* Information	P1	
* Warning	P2	
* Average	P3	
* High	P4	
* Disaster	P5	

Custom severity names affect all locales and require manual translation!

Rysunek 4.46. Niestandardowe ustawienia poziomów wyzwalacza

3. Nie zapomnij wcisnąć niebieskiego przycisku *Update* u dołu strony, aby zapisać zmiany.

Wyjaśnienie

Nie wszystkie firmy lubią stosować takie pojęcia jak *High* czy *Disaster* do opisu poziomów zagrożenia; wolą używać określeń typu *P1* czy *P2*. Dzięki możliwości modyfikowania poziomów wyzwalacza możemy bardziej dostosować Zabbixa do własnych potrzeb i na przykład wprowadzić nomenklaturę stosowaną w innych narzędziach.

Zmiana poziomów wyzwalacza nie jest w żadnym wypadku obowiązkowa, ale stanowi dobry sposób na szybsze przyzwyczajenie się do Zabbixa, jeśli dotychczas korzystałeś z innych narzędzi.

Skorowidz |

A

- AD, Active Directory, 83
- agent, 99
 - aktywny, 102, 105
 - automatyczna rejestracja, 301, 356
 - instalacja interfejsu, 99
 - konfiguracja wykrywania, 290
 - monitorowania HTTP, 143
 - pasywny, 100, 105
- alerty
 - dostosowywanie, 193
 - efektywne, 190
 - konfiguracja, 182
 - Microsoft Teams, 378
 - Slack, 366
 - tworzenie, 183
- aplikacja korporacyjna
 - dodawanie grup, 86
 - dodawanie użytkowników, 86
 - przydzielanie ról, 86
 - strona, 85
 - tworzenie, 85
 - wybór protokołu SAML, 87
- aprowizacja JIT, 92
- architektura Zabbixa
 - baza danych, 29
 - frontend, 30
 - serwer, 29
- Atlassian Opsgenie
 - alert Zabbixa, 403
 - integracja z Zabbixem, 396
- automatyczna rejestracja
 - aktywnego agenta, 301, 356
 - hosta, 305
- AWS
 - monitorowanie usługi, 481
- Azure AD SAML, 81

B

- baza danych
 - konfiguracja, 37
 - okno konfiguracji, 44
- CMDB, 257
- MySQL, 450
 - konfigurowanie partycjonowania, 457
 - strojenie, 450, 454
 - zabezpieczenia, 470
- PostgreSQL TimescaleDB, 465
 - konfiguracja, 467
- boty Telegram, 386

C

- certyfikaty SLL, 478

D

- Docker
 - monitorowanie kontenerów, 498
- dostawca usług zarządzanych, MSP, 346

E

- edytowanie hosta, 343
- ekran
 - Add widget, 53
 - Administration/General/Other, 258
 - Administration/Housekeeping, 463, 469
 - Administration/Proxies, 336–339, 341, 350
 - Administration/Proxy groups, 350
 - Alerts/Actions, 299
 - Alerts/Scripts, 422, 423
 - Data collection/Discovery, 297, 298
 - Data collection/Host, 343
 - Data collection/Hosts, 356, 359, 361
 - Data collection/Maintenance, 428, 429
 - Inventory/Hosts, 258
 - logowania, 48

ekran

- Media types, 186
- Monitoring/Discovery, 295, 300
- Monitoring/Hosts/Graphs, 231
- Monitoring/Latest, 206, 315
- Monitoring/Latest data, 359, 362
- Monitoring/Maps, 423
- powitalny, 32
- pulpitu globalnego, 251
- Reports/Action log, 267
- Reports/Audit, 267
- Reports/Availability report, 266
- Reports/Notifications, 268
- Reports/Scheduled reports, 272
- Reports/System information, 264
- Reports/Trigger top 100, 266
- Service/Service, 280–285
- Trigger actions, 182
- tworzenia nowej pozycji, 156, 159
- Users/API tokens, 409
- Users/User groups, 407
- Users/Users, 395

F

frontend, 47

- Global view, 49
- konfiguracja, 30
- logowanie, 48
- monitorowanie serwera proxy, 362, 363
- panel boczny, 56
- widok globalny, 48
- widżety, 48–55
- zakładki, 56–62

funkcja

- timeleft, 175, 179
- trendavg, 174, 178

funkcje matematyczne, 177, 179

I

identyfikator obiektu, OID, 203, 221, 296

instalowanie

- interfejsu agent 2, 99
- serwera, 26

interfejs

- agent 2, 99
- API Zabbixa, 405, 411
 - działanie, 415
 - tworzenie jumphosta, 415
- pośredniczący ODBC, 142
- użytkownika, 56

inwentarz, 257

J

język JSONPath, 329

JIT, Just In Time, 81

JMX, Java Management Extensions, 135, 289, 312

jump host

- schemat wykorzystania skryptu, 420
- tworzenie, 415

K

komunikacja

- serwer–host, 126
- w architekturze Zabbixa, 29, 35
- z agentem aktywnym, 105
- z agentem pasywnym, 105
- z hostem SNMP, 111
- z interfejsem ODBC, 142
- z obydwoma typami agentów, 106
- z serwerem Java, 138
- ze skryptem zewnętrznym, 135

konfigurowanie

- agenta monitorującego, 99
- agenta monitorującego HTTP, 143
- alertów, 182
- bazy danych, 37, 43, 467
- frontendu, 30
- hosta, 108, 113, 121, 137

Dockera, 499

WWW, 147

interfejsu SNMP, 114

inwentarza Zabbix, 257

kanału Email, 187

makr hosta, 141

mediów, 188

monitorowania

bazodanowego, 139

JMX, 135

usługi AWS, 481

usługi Microsoft Azure, 489

odbiornika pozycji przechwytywania, 124

okresów konserwacji, 428

partycjonowania bazy danych, 457

połączenia z bazą danych, 33

pozycji, 128–131, 135, 141, 144

dla klucza, 115, 116

dla sprawdzania, 121

równoważenia obciążenia, 349

serwera proxy, 332

tokenów API, 406

uwierzytelniania LDAP, 95

- węzłów klastra serwerów, 39
- wykrywania agenta, 290
- wykrywania sieci SNMP, 316
- wysokiej dostępności, 36, 40
- wyzwalaczy, 163, 273
- zaawansowanych wyzwalaczy, 173
- znacznika odbiornika, 125
- znaczników na poziomie szablonu, 200
- konserwacja, 428
- kopia zapasowa konfiguracji, 430

L

- liczniki wydajności, 306
- LLD, Low-Level Discovery, 196, 289
- logowanie, 48

Ł

- ładunek, payload, 414

M

- makra
 - globalne, 109
 - LLD, 211, 329
 - na poziomie hosta, 212
 - na poziomie szablonu, 209
 - użytkownika, 212
 - wbudowane, 211, 213
 - wyrażeń, 211
- mapa sieci lokalnej, 236
- mapy, 234
 - edycja łącza, 240
 - elementy, 238
 - etykiety, 241
 - tworzenie, 237
 - włączanie i wyłączanie hosta, 420
- maszyna wirtualna, virtual machine, VM, 327
- MIB, Management Information Base, 203
- Microsoft Azure
 - monitorowanie usługi, 489
- Microsoft Teams
 - konfigurowanie wysyłania alertów, 378
- modernizowanie
 - backendu, 434
 - dystribucje na silniku Debian, 437
 - dystribucje na silniku RHEL8, 435
 - bazy danych, 438
 - dystribucje na silniku Debian, 440
 - dystribucje na silniku RHEL, 439

- konfiguracji, 442
 - dystribucje na silniku Debian, 444
 - dystribucje na silniku RHEL, 443
- monitorowanie
 - aktywnych hostów przeglądarki, 150, 151
 - bazodanowe
 - konfiguracja, 139
 - czasu działania przeglądarki, 149
 - Dockera, 498
 - hostów
 - za pomocą serwerów proxy, 342
 - HTTP, 143
 - JMX, 135
 - konfiguracja agenta, 99
 - serwerów proxy, 357, 362, 363
 - SNMP, 106, 316
 - konfiguracja, 112
 - stron internetowych, 146
 - symulacja użytkownika, 146
 - usługi
 - AWS, 481
 - biznesowej, 273, 287
 - Microsoft Azure, 489
 - SSH, 163
 - za pomocą obiektów JMX, 316
 - zdalne serwera proxy, 360, 363

N

- narzędzie wysyłające, sender, 120

O

- obiekty JMX
 - wykrywanie, 312
- ODBC, 142
- OID, Object Identifier, 203
- okno
 - Create action/New, 302
 - Create proxies, 339, 341
 - dodawania hostów do konserwacji, 429
 - edycji elementu mapy, 238, 239
 - edytowania pozycji dla hosta, 123
 - konfiguracji
 - bazy danych, 43, 44
 - Host permissions, 69
 - hosta, 147
 - konta użytkownika, 78
 - makr hosta, 149
 - pozycji, 148
 - roli użytkownika, 71
 - Template permissions, 67, 69

okno

konfiguracji

uprawnień użytkownika, 79, 80, 82

User group, 66–68

User roles, 71

użytkownika, 81, 82

znacznika pozycji, 148

logowania, 90, 96

mapowania wartości, 123

New proxy group, 350

szczegółów grupy, 84

tworzenia akcji, 299

tworzenia hosta, 359

Users, 77

Users and groups, 83

ustawień serwera, 44

wyboru protokołu, 87

OpenLDAP, 92

P

panel boczny, 56

plik

/etc/hosts, 419

konfiguracyjny agenta, 157

pliki JSON, 323

polecenie

cat, 414

GetBulk, 323

ifconfig, 158, 160

snmptranslate, 204

snmpwalk, 117, 119, 204

system.run, 158

połączenie

aktywne serwera, 342

pasywne z serwerem proxy, 338

pozycja

identyfikatora OID, 205

przechwytywania, trapper, 120, 126

sprawdzania portu 22, 122

sprawdzania portu SSH, 122

system.run agenta, 159

pozycje

modyfikowanie wartości, 155

nadrzędne, 130

obliczone, calculated items, 127

konfiguracja, 127

schemat, 132

przeglądarki, Browser items, 146

zależne, dependent items, 127

konfiguracja, 129

schemat, 133

proces

odbiorczy agenta, 118

odbiorczy agenta HTTP, 118

odbiorczy SNMP, 118

porządkujący, housekeeper, 448, 453

protokół

LDAP, 92

SAML, 81

SMTP, 185

SNMP, 106

prototyp

pozycji, 232, 235, 320

pozycji LLD, 217–219, 221

wykresu, 233

wyzwalacza LLD, 220, 222

prototypy znaczników, tag prototypes, 203

przechwytywanie, 120, 123

przetwarzanie wstępne, 320, 321

pulpit

dla monitorowanych hostów, 242

hosta, 253–255

ekran, 256

szablon, 253

tworzenie, 242, 243

widoku globalnego, 49

Python

tworzenie jumpha, 415

R

raporty

tworzenie, 263

w formacie PDF, 269

zaplanowane, 269

reguły, *Patrz także* makra

LLD, 196

tworzenie hostów, 323

w szablonach, 213

wykrywania, 355

AWS by HTTP, 488

Azure by HTTP, 496

Discover JMX, 313

hostów, 291

sieci, 319

role

domyślne użytkownika, 79

użytkownika, 70

S

SAML, Security Assertion Markup Language, 81

schemat

- aktywnego serwera proxy, 344, 345

- integracji Zabbixa

 - z aplikacją Microsoft Teams, 386

 - z aplikacją Slack, 378

 - z platformą Opsgenie, 404

 - z Telegramem, 395

- komunikacji, 29, 35

- komunikacji agenta HTTP, 145

- pasywnego serwera proxy, 344

- pozycji zależnej, 132, 133

- przepływu informacji o problemie, 189

- serwerowego odbiornika pułapek, 126

- struktury Zabbixa, 433

- uwierzytelniania

 - LDAP JIT, 97

 - SAML, 90

 - SAML JIT, 91

serwer proxy

- aktywny, 338, 343

- konfiguracja, 332

- monitorowanie, 357, 362

 - hostów, 342

 - z poziomu frontendu, 362, 363

- pasywny, 335, 338, 343

- równoważenie obciążenia, 349

- szyfrowane połączenia, 346, 349

- wersje, 334

- zdalne monitorowanie, 360, 363

serwery

- informacje systemowe, 45

- instalacja, 26

- konfiguracja węzłów klastra, 39

- konfiguracja wysokiej dostępności, 40

- okno ustawień, 44

- struktura, 46

- wysoka dostępność, HA, 36

skrypty, 405

- ręczne wprowadzanie danych, 425

- schemat wykonywania, 424

- tworzenie, 422, 423

Slack

- konfigurowanie wysyłania alertów, 366

- powiadomienia Zabbixa, 377

SMTP, Simple Mail Transfer Protocol, 185

sprawdzenia

- proste, simple checks, 120, 125

- zewnętrzne, 133, 134

statystyki ruchu internetowego, 157, 159

strona, *Patrz także* ekran

- Zabbixa, 57

struktura Zabbixa, 433

szablony

- AWS, 488

- Azure, 497

- domyślne, 199

- konfigurowanie makr, 209

- Linux by SNMP, 109

- nazwy, 199

- pozycji, 203

- pulpitów, 253

- reguły LLD, 213

- reguły wykrywania, 488

- struktura zagnieżdżania, 225

- tworzenie, 197

- wiadomości, 187

- wyzwalaczy, 207

- zagnieżdżanie, 223

szyfrowanie, 478

- połączenia, 346

- PSK, 348

- schematy, 479

T

Telegram

- powiadomienia Zabbixa, 386, 394

tokeny API, 406, 409

- konfiguracja, 406

tworzenie

- akcji, 183, 299, 302

- aplikacji korporacyjnej, 85

- grup użytkowników, 65

- hosta, 323, 325

- jumphosta, 415

- konserwacji Patch Tuesday, 428

- kopii zapasowej konfiguracji, 430

- mapy, 237

- okresu konserwacji Patch Tuesday, 429

- pozycji

 - dla hosta, 326

 - ICMP, 229

 - przechwytywania, 123

 - szablону, 203

- prototypu

 - hosta, 327

 - pozycji, 232, 235, 314

 - pozycji LLD, 217–219

- pulpitu hosta, 253

- tworzenie
 - raportów, 263
 - reguł LLD, 215, 216, 326
 - skryptu
 - Disable, 423
 - Enable, 422
 - sprawdzeń
 - prostych, 120
 - zewnętrznych, 133
 - szablonów, 197, 224
 - tokena API, 409
 - uprawnień
 - hostów, 407
 - szablonów, 407
 - użytkownika API, 408
 - użytkowników, 76
 - widżetu, 244
 - widżetu wykresu, 245
 - wykresów, 230
 - wyzwalacza, 164–166
 - wyzwalaczy szablonu, 207
 - zaplanowanych raportów, 269
 - znacznika
 - pozycji, 326
 - prototypu, 218, 219
- typ Secret text, 109

U

- uprawnienia
 - hostów, 407
 - szablonów, 407
- usługi
 - chmurowe, 480
 - zewnętrzne, 365
- ustawienia
 - SAML, 87–89
 - użytkownika Admin, 94
- utrzymywanie wydajności, 446
- uwierzytelnianie
 - LDAP, 92
 - konfiguracja, 95
 - schemat, 97
 - z aprowizacją JIT, 95, 97
 - SAML, 81
 - wieloskładnikowe, 67
 - wybór domyślnej metody, 94
 - za pomocą protokołu SAML, 90
- użytkownicy
 - aprowizacja JIT, 81
 - grupy, 65
 - role, 70

- symulacja, 146
- tworzenie, 76
- uwierzytelnianie, 81
- użytkownik Admin, 94

W

- webhook Zabbix, 378
- widżet
 - Clock, 52
 - Gauge, 250
 - Geomap, 55, 259–63
 - Graph, 54, 254, 255
 - Host availability, 51
 - Item value, 55, 246, 254
 - Pie chart, 251
 - Problems, 255
 - Problems by severity, 51
 - System information, 49
 - Top hosts, 246–249
 - wyświetlający problemy, 244
- wirtualna sieć prywatna, VPN, 346
- wskaźniki z hosta, 322
- wstępne przetwarzanie wartości pozycji, 155
- wydajność serwera
 - proces porządkujący, 448, 453
 - procesy odbiorcze, 455
 - procesy Zabbix, 447, 452
 - strojenie bazy danych MySQL, 450, 454
- wykresy, 228
 - tworzenie, 230
- wykrywanie, 354
 - hostów z agentem, 290–294
 - liczników wydajności, 306
 - niskopoziomowe, LLD, 289
 - obiektów JMX, 312
 - reguły, 355
 - sieci SNMP, 296, 316
- wyrażenie regularne, regex, 161
- wysyłanie alertów
 - Microsoft Teams, 378
 - Slack, 366
- wyzwalacze, triggers, 163
 - dla nowej wersji Zabbixa, 165, 170
 - dla szablonów, 207, 210, 211
 - domyślne poziomy, 194
 - konfiguracja, 163
 - korzystanie z wielu pozycji, 166, 171
 - LLD
 - definiowanie prototypu, 220
 - tworzenie znacznika prototypu, 220

- monitorujące usługi SSH, 163, 168
- monitorujące interfejs, 222
- poziomy, 167
- schemat struktury, 192
- trigger top 100, 266
- tworzenie, 164–166
- ustawienia poziomów, 195
- zaawansowane, 173
 - funkcja timeleft, 175, 179
 - funkcja trendavg, 174, 178
 - przesunięcie czasowe, 177, 179, 181
- zmiana nazwy, 190
- znaczniki, 190

Z

zakładka

- Administration, 57, 62
- Alerts, 57, 61
- Data collection, 57, 60
- dodawania makr, 206

- Encryption, 337
- Inventory, 57, 59
- Item tags, 320
- LDAP settings, 94
- LLD macros, 327
- Macros, 114
- Media użytkownika Admin, 188
- Monitoring, 56, 58
- Permissions, 409
- Preprocessing, 117, 232, 236
- Reports, 57, 59
- Services, 57, 58
- Tags, 116, 122, 128, 201
- Users, 57, 61
- Value mapping, 114
- zarządzanie
 - bazami danych, 456
 - użytkownikami, 64
- znaczniki
 - konfiguracja, 200

PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

Monitoring czy chaos? Prosty wybór!

Wybierz Zabbix – pełną kontrolę!

Systemy monitorujące są często bagatelizowane, tymczasem to właśnie dzięki nim możesz uniknąć problemów. Spośród dostępnych na rynku opcji Zabbix jest atrakcyjnym rozwiązaniem: pozwala na monitoring dowolnie dużej infrastruktury składającej się z wielu różnych komponentów i na daleko idącą automatyzację zadań, zapewnia też rozbudowane wizualizacje. W dodatku można z niego korzystać zupełnie za darmo.

Ten praktyczny przewodnik zawiera receptury uwzględniające nowe funkcje środowiska Zabbix. Znajdziesz tu informacje potrzebne do konfiguracji Zabbixa z wbudowanym trybem wysokiej dostępności. Dowiesz się też, jak korzystać z aprowizacji użytkowników LDAP JIT, implementować funkcję niskopoziomowego wykrywania hostów i tworzyć zaawansowane wyzwalacze. Każda receptura została opracowana z myślą o różnych typach monitorowania i korzystania z serwerów proxy Zabbix. Ponadto nauczysz się modyfikować serwer i bazę danych Zabbix, a także zarządzać nimi za pomocą interfejsu API. Poznasz również rozwiązania problemów, na które możesz natrafić podczas pracy z Zabbixem.

W książce:

- implementacja infrastruktury w trybie wysokiej dostępności
- szablony monitorowania
- skalowanie środowiska Zabbix
- niestandardowe integracje i interfejsy, a także zaawansowane wyzwalacze i alerty
- zaawansowane operacje zarządzania bazami danych Zabbix
- monitorowanie usług chmurowych, takich jak Amazon Web Services, Azure czy Docker

Nathan Liefing jest konsultantem i trenerem. Ma duże doświadczenie w zarządzaniu sieciami. Obecnie pracuje w Opensource ICT Solutions BV. Tworzy profesjonalne składniki Zabbixa dla największych firm z całego świata.

Brian van Baake w 2017 roku uzyskał certyfikat Zabbix Certified Trainer. Rok później założył dwie firmy koncentrujące się głównie na tworzeniu środowisk Zabbix. Jest też właścicielem pięknego kota o imieniu... Zabbix.

	KOD KORZYŚCI Sięgnij po więcej! ▶ 
 helion.pl  HELION S.A. ul. Kościuszki 1c 44-100 Gliwice tel.: 32 230 98 63 helion@helion.pl	ISBN 978-83-289-2476-5  9 788328 924765
Cena: 139,00 zł	

<packt>