Architektura Zabbixa: serwer, proxy, agent, frontend

Poziom: podstawy **Dotyczy: Zabbix 6.0 LTS / 6.4** **Czas: ~20 min**

Zanim zaczniemy dodawać hosty i szablony, sprawdzamy z jakich elementów Zabbix w ogóle się składa i którędy faktycznie płyną dane od monitorowanej maszyny do wykresu na pulpicie.

Skróty użyte na tej stronie (7)

Skrót	Co znaczy
API	Application Programming Interface: sposób, w jaki programy rozmawiają ze sobą
CPU	Central Processing Unit: procesor
DMZ	Demilitarized Zone: osobna sieć za zaporą dla usług dostępnych z internetu, odgrodzona od sieci wewnętrznej; w FortiGate 100F tak jest opisany jeden z portów
LTS	Long Term Support: wersja z długim wsparciem poprawkami
PHP	PHP: Hypertext Preprocessor, język programowania, w którym napisany jest frontend Zabbixa i wiele innych aplikacji webowych
SNMP	Simple Network Management Protocol: protokół odczytu stanu urządzeń sieciowych
TCP	Transmission Control Protocol: protokół sieciowy z potwierdzaniem dostarczenia

Z życia wzięte

Z ŻYCIA WZIĘTE

O 7:50 Tomek dostaje powiadomienie, że host `app01` jest niedostępny w Zabbixie, chociaż aplikacja na nim działa bez żadnych skarg od użytkowników. Sprawdza kierunek połączenia w `zabbix_agentd.conf` i widzi, że ktoś dzień wcześniej dopisał `ServerActive`, ale zapomniał

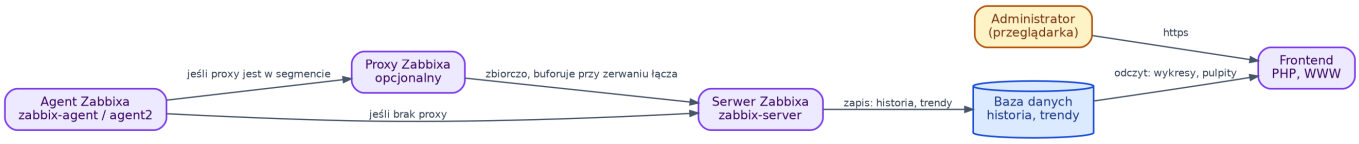
otworzyć na firewallu portu 10051 w stronę serwera. Po dodaniu reguły agent od razu zaczyna wysyłać zaległe dane, a status hosta wraca do zielonego bez żadnej zmiany po stronie samej aplikacji.

Czym jest Zabbix

Pojęcie: Zabbix. System monitoringu typu open source, zbierający metryki (dostępność usługi, zużycie zasobów, liczniki aplikacyjne) z monitorowanych hostów, oceniający je względem zdefiniowanych progów i wyzwalający akcje — powiadomienie, zgłoszenie, komendę zdalną — gdy próg zostanie przekroczony. W odróżnieniu od narzędzi typu „sprawdź czy port odpowiada" Zabbix trzyma pełną historię zebranych wartości, więc odpowiada nie tylko na pytanie „czy działa teraz", ale też „jak to wyglądało tydzień temu".

Cztery elementy i jak przepływają dane

Zanim wejdziemy w konfigurację, rozkładamy Zabbixa na cztery role — każda odpowiada za inny fragment drogi, jaką pokonuje pojedyncza zebrana wartość.



Wartość płynie od agenta albo proxy do serwera, dalej do bazy danych, a frontend czyta ją dopiero z bazy (kliknięcie otwiera pełny rozmiar)

Pojęcie: serwer Zabbixa (zabbix-server). Centralny proces, który przechowuje konfigurację (hosty, szablony, wyzwalacze), planuje zbieranie danych, ocenia wyzwalacze i uruchamia akcje. Sam serwer niczego nie wystawia użytkownikowi — to proces działający w tle, komunikujący się z bazą danych i z agentami albo proxy.

> **Pojęcie: baza danych Zabbixa.** Miejsce, w którym serwer trzyma zarówno konfigurację, jak i zebrane dane — historię pojedynczych wartości i trendy, czyli zagregowane godzinowe minimum/maksimum/średnią. To ta baza rośnie najszybciej ze wszystkich elementów instalacji, temat rozwijamy na stronie Utrzymanie serwera Zabbix.

> **Pojęcie: frontend Zabbixa.** Aplikacja webowa (PHP), przez którą konfigurujemy hosty i szablony oraz przeglądamy wykresy i pulpity. Frontend czyta i zapisuje bezpośrednio do tej samej bazy danych co serwer — nie rozmawia z serwerem przez żadne własne API w trakcie zwykłej pracy, poza kilkoma wyjątkami jak ręczne uruchomienie akcji.

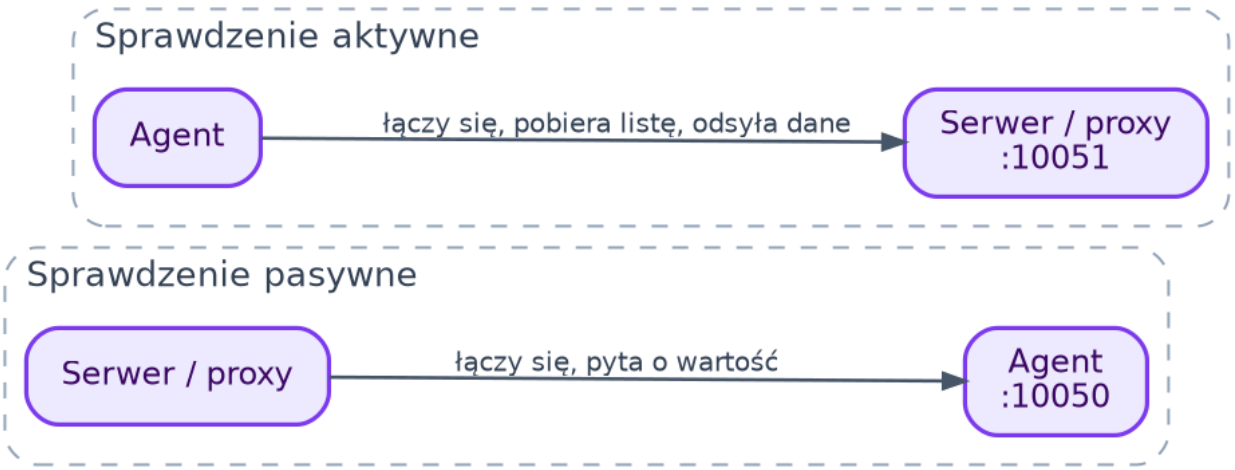
> **Pojęcie: agent Zabbixa (zabbix-agent / zabbix-agent2).** Lekki proces instalowany na monitorowanym hoście, który udostępnia lokalne metryki (obciążenie CPU, zajętość dysku, status usługi systemd) na żądanie serwera albo proxy. Instalację i konfigurację agenta opisujemy osobno na stronie Agent — instalacja i konfiguracja.

> **Pojęcie: proxy Zabbixa (zabbix-proxy).** Opcjonalny pośrednik między agentami a serwerem głównym, zbierający dane lokalnie i przesyłający je do serwera zbiorczo. Proxy nie ma własnego frontendu ani osobnej konfiguracji hostów — cała konfiguracja proxy jest kopią fragmentu konfiguracji serwera głównego, pobieraną automatycznie.

Droga pojedynczej wartości wygląda więc tak: agent (albo urządzenie SNMP, albo sam serwer wykonujący sprawdzenie zewnętrzne) → proxy, jeśli jest w danym segmencie zdefiniowane → serwer → baza danych → frontend, który dopiero z tej bazy rysuje wykres. Żaden z tych elementów nie omija bazy danych poza samym momentem zbierania — nawet frontend nie pyta serwera „jaka jest aktualna wartość", tylko czyta to, co serwer już zapisał.

Agent pasywny kontra agent aktywny

To jest rozróżnienie, które trzeba mieć jasne, zanim zabierzemy się za konfigurację `zabbix_agentd.conf` — bo od niego zależy, kto się do kogo łączy, i to determinuje wymagania na firewallu w obie strony.



Przy sprawdzeniu pasywnym serwer albo proxy łączy się do agenta, przy aktywnym to agent łączy się do serwera albo proxy (kliknięcie otwiera pełny rozmiar)

Pojęcie: sprawdzenie pasywne (passive check). Serwer albo proxy sam, w zaplanowanym momencie, łączy się do agenta i pyta o wartość konkretnego klucza. To agent nasłuchuje, a serwer inicjuje połączenie — model „serwer pyta, agent odpowiada".

Pojęcie: sprawdzenie aktywne (active check). Agent sam, cyklicznie, łączy się do serwera albo proxy, pobiera od niego listę elementów, jakie ma dla tego hosta zbierać, i sam z własnej inicjatywy odsyła zebrane wartości. To odwraca kierunek połączenia — serwer nie musi mieć żadnej trasy sieciowej do agenta, wystarczy że agent ma trasę do serwera.

Cecha	Pasywne	Aktywne
Kto inicjuje połączenie	serwer/proxy → agent	agent → serwer/proxy
Kierunek ruchu na firewallu	do hosta monitorowanego	z hosta monitorowanego
Parametr w `zabbix_agentd.conf`	`Server`	`ServerActive`
Typowe zastosowanie	hosty w tej samej sieci co serwer	hosty za NAT, laptopy, segmenty bez dostępu przychodzącego
Obciążenie agenta	niskie, odpowiada na żądanie	agent sam utrzymuje harmonogram i kolejkę

W praktyce w jednej instalacji obie metody współistnieją na tym samym hoście — nic nie stoi na przeszkodzie, żeby część elementów szablonu była pasywna, a inne (typowo `zabbix_agent.hostname`, `log[]`, elementy niskopoziomowego wykrywania) aktywne, bo agent i tak musi się zgłosić do serwera po listę, żeby wiedzieć co monitorować aktywnie.

Porty 10050 i 10051

Port 10050 to port, na którym nasłuchuje **agent** — zarówno przy sprawdzeniach pasywnych (serwer łączy się na ten port, żeby zapytać o wartość), jak i jako port, spod którego agent sam wychodzi przy sprawdzeniach aktywnych.

Port 10051 to port, na którym nasłuchuje **serwer i proxy** — agent przy aktywnym sprawdzeniu łączy się na ten port, żeby odebrać listę elementów i odesłać zebrane dane. Na ten sam port przychodzi też ruch z proxy do serwera nadrzędnego.

Uważaj: pomylenie tych dwóch portów jest najczęstszą przyczyną „host niedostępny" przy pierwszym podłączaniu nowego serwera — reguła firewalla otwarta w złym kierunku (np. przepuszczony ruch wychodzący 10050 zamiast przychodzącego, albo odwrotnie) daje dokładnie taki sam objaw jak brak reguły w ogóle.

Po co stawiamy proxy

Proxy nie jest wymagane w małej, płaskiej sieci — dodajemy je świadomie, z jednego z dwóch powodów.

Odciążenie serwera głównego. Odpytywanie tysięcy elementów danych z setek hostów generuje ruch sieciowy i obciążenie procesora na samym akcie zbierania (parsowanie odpowiedzi SNMP, nawiązywanie połączeń TCP). Proxy przejmuje tę pracę lokalnie w swoim segmencie i do serwera głównego wysyła już tylko zebrane, gotowe wartości w mniejszej liczbie większych paczek.

Monitorowanie segmentu bez bezpośredniej trasy do serwera. Oddział, sieć DMZ albo środowisko chmurowe z ograniczonym ruchem wychodzącym — zamiast otwierać firewalla dla każdego hosta osobno do centralnego serwera, otwieramy jedną regułę dla proxy, a reszta hostów rozmawia wyłącznie z lokalnym proxy.

Dodatkowa własność proxy, która bywa równie ważna jak odciążenie: proxy buforuje dane lokalnie, jeśli traci łączność z serwerem głównym. Krótka awaria łącza między lokalizacjami nie oznacza dziury w historii — proxy dosyła zaległe dane, gdy połączenie wraca.

Baza danych Zabbixa i jej przyrost

Baza danych trzyma dwa rodzaje danych pomiarowych o zupełnie różnym tempie wzrostu, co rozwijamy szerzej przy retencji na stronie Utrzymanie serwera Zabbix: **historię** (`history*`) — każdą pojedynczą zebraną wartość z osobnym znacznikiem czasu, oraz **trendy** (`trends*`) — zagregowane dane godzinowe (min/max/avg), trzymane znacznie dłużej, bo zajmują ułamek

miejsca historii.

Rozmiar tabel historii rośnie wprost proporcjonalnie do trzech rzeczy naraz: liczby elementów danych, częstości ich odpytywania (interwału) i długości okresu retencji. Podwojenie liczby monitorowanych hostów albo skrócenie interwału o połowę podwaja przyrost bazy — to jest pierwsza rzecz, którą sprawdzamy, gdy dysk pod bazę Zabbixa zaczyna się kończyć szybciej niż zakładano.

Zabbix a monitoring aplikacyjny wyższego poziomu

Zabbix odpowiada dobrze na pytania o dostępność, zasoby i proste metryki aplikacyjne wystawiane przez `UserParameter` (temat rozwijamy na stronie Monitorowanie typowych usług), ale nie jest narzędziem do śledzenia pojedynczego żądania przechodzącego przez wiele mikroserwisów naraz — do tego służą dedykowane systemy śledzenia rozproszonego (distributed tracing), poza zakresem tej strony. Rozróżnienie to jest praktyczne przy planowaniu monitoringu nowej aplikacji: pytanie „czy usługa odpowiada i ile zużywa zasobów” trafia do Zabbixa, pytanie „dlaczego akurat to jedno żądanie trwało trzy sekundy, przechodząc przez pięć usług” wymaga innego narzędzia.

Częste błędy

Traktowanie proxy jako serwera zapasowego. Proxy nie ma własnej konfiguracji hostów ani szablonów niezależnej od serwera głównego — awaria serwera głównego zatrzymuje ewaluację wyzwalaczy i wysyłkę powiadomień w całej instalacji, proxy tylko buforuje surowe dane do czasu, aż serwer wróci.

Otwarcie firewalla tylko w jedną stronę dla sprawdzeń aktywnych. Sprawdzenie aktywne wymaga ruchu wychodzącego z hosta monitorowanego na port 10051 serwera/proxy — samo otwarcie portu 10050 na hoście monitorowanym (jak przy pasywnym) nic tu nie da.

Mylenie „agent nie odpowiada" z „host niedostępny w Zabbiksie". To drugie jest stanem hosta ustalanym przez serwer na podstawie nieudanych prób połączenia — objaw bywa identyczny zarówno przy złej regule firewalla, złym adresie w konfiguracji hosta, jak i przy zatrzymanym procesie agenta; diagnostykę rozkładamy na czynniki pierwsze na stronie Agent — instalacja i konfiguracja.

□□ Ćwiczenia

> Poniższe ćwiczenia wykonujemy na dowolnym hoście testowym z zainstalowanym agentem Zabbixa.

Ćwiczenie 1 — rozpoznanie kierunku połączenia

Mając plik `zabbix_agentd.conf` z wypełnionym tylko parametrem `Server=10.20.10.51` (bez `ServerActive`), określamy: czy to sprawdzenie pasywne czy aktywne, i w którą stronę musi być otwarty firewall.

Odpowiedź

To sprawdzenie pasywne — parametr `Server` ogranicza, z jakich adresów agent przyjmie połączenie przychodzące na port 10050. Firewall na hoście monitorowanym musi przepuszczać ruch przychodzący z adresu serwera na port 10050; sam host monitorowany nie musi mieć żadnej reguły wychodzącej dla Zabbixa.

Pytania kontrolne

1. Dlaczego host za NAT-em bez publicznego adresu prawie zawsze wymaga sprawdzeń aktywnych, a nie pasywnych?

Odpowiedź

Bo sprawdzenie pasywne wymaga, żeby serwer mógł zainicjować połączenie do agenta — bez publicznego adresu i przekierowania portu serwer fizycznie nie ma jak się połączyć. Sprawdzenie aktywne odwraca kierunek: to agent łączy się na zewnątrz do serwera, co działa zza NAT-u bez żadnej dodatkowej konfiguracji sieciowej po stronie hosta monitorowanego.

2. Co się stanie z danymi zbieranymi przez proxy, jeśli łączność między proxy a serwerem głównym zerwie się na godzinę?

Odpowiedź

Proxy nie przestaje zbierać danych od swoich hostów — buforuje je lokalnie we własnej bazie i dosyła zaległość do serwera głównego, gdy połączenie wróci. Wyzwalacze zależne od tych danych zostają jednak ocenione z opóźnieniem, dopiero po dotarciu zaległych wartości do serwera.

3. Dlaczego frontend Zabbixa przestaje pokazywać nowe dane, jeśli padnie sam proces serwera, mimo że baza danych i frontend nadal działają?

Odpowiedź

Frontend czyta wyłącznie to, co już leży w bazie danych — to serwer jest jedynym elementem odpowiedzialnym za zbieranie nowych wartości z agentów i zapisywanie ich do tej bazy. Bez działającego serwera baza danych po prostu przestaje otrzymywać nowe wpisy, więc frontend wciąż poprawnie wyświetla historię sprzed awarii, ale nic nowszego.

Powiązane: Agent — instalacja i konfiguracja · Host, element danych, wyzwalacz, akcja: jak to się składa · Utrzymanie serwera Zabbix

Źródła: dokumentacja Zabbix 6.0, rozdział „Concepts" i „Proxies" · `man zabbix_agentd`

Revision #37
Created 2026-09-25 07:40:29 UTC by Tomasz Konieczny
Updated 2026-10-26 14:09:44 UTC by Tomasz Konieczny