

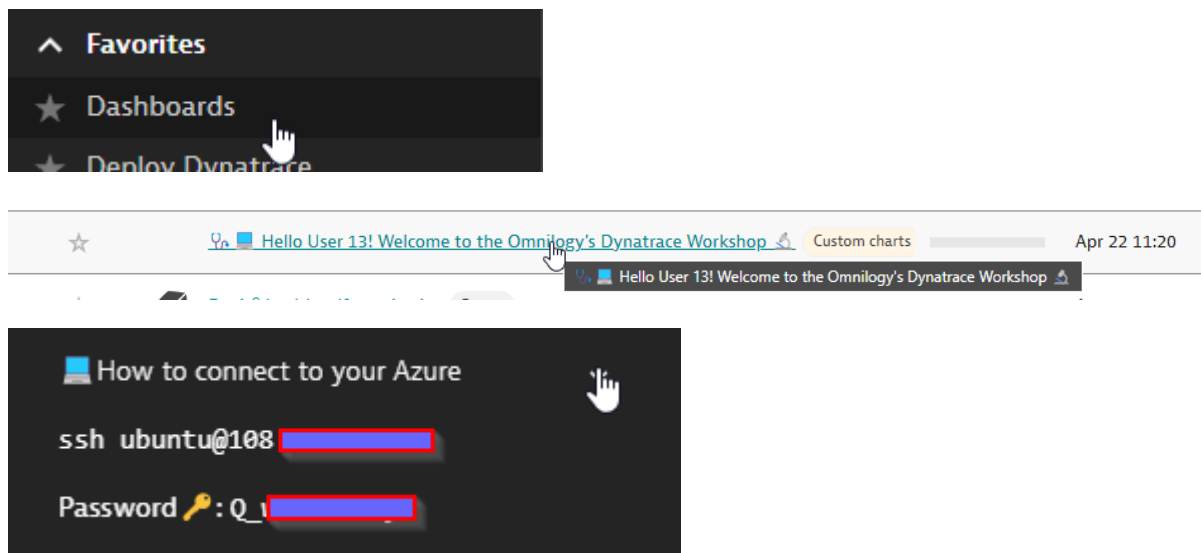
Część 2: Warsztaty i ćwiczenia (czas 3h 30min):

1. Omówienie środowiska oraz zakresu ćwiczeń - 15 min

Środowisko Dynatrace Managed: <https://dynatrace.omniloggy.pl>

Środowisko Dynatrace Playground: <https://wkf10640.apps.dynatrace.com/>

Serwer Linux: adres i dane dostępowe na dashboardzie Dynatrace Managed (prawy dolny róg):



Przydatne komendy:

Przejsście na root-a (wykonać po zalogowaniu):

```
sudo su -
```

Zatrzymanie aplikacji (stop procesów):

```
docker-compose down
```

Uruchomienie aplikacji (start procesów):

```
docker-compose up -d
```

Ćwiczenie 1

Instalacja i uruchomienia monitoringu - 30 min

Cel ćwiczenia:

Poznanie mechanizmów instalacji agenta Dynatrace oraz automatycznego skonfigurowania monitoringu dla aplikacji działającej na serwerze.

1. Instalacja agenta:

- Wybierz „Manage” -> „Deploy Dynatrace” z lewego menu
- Wybierz „Start installation” oraz odpowiedni system operacyjny
- Na kreatorze instalacji:
 - utwórz token,
 - ustaw Monitoring mode na Full stack,
 - ustaw Host Group – wartość EasyTravel (hint: wpisz wartość i wybierz ją po wpisaniu)
 - zweryfikuj, jak zmodyfikowało się polecenie instalacyjne w pkt.4
- Z poziomu linux jako root pobierz agenta poleceniem wget
- Sprawdź podpis pobranej paczki
- Zainstaluj agenta
- Hint – używaj przycisku kopiowania poleceń zamiast zaznaczania i Ctrt+C
- Zweryfikuj poprzez Deployment status, czy serwer pojawił się w Dynatrace. Rozwiń szczegóły o sprawdź, czy procesy Tomcat są monitorowane
- Wejdź w szczegóły serwera (link) i zweryfikuj zakres danych infrastrukturalnych widocznych na ekranie
- Wejdź w szczegóły jednego z procesów tomcat i zweryfikuj zakres danych jakie zbiera agent. Czy na górze ikonografiki w polu Service widać jakąś wartość?

2. Uruchom monitoring aplikacyjny

- Z poziomu linux jako root zatrzymaj aplikację (docker-compose down)
- Uruchom aplikację docker-compose up -d
- Poczekaj chwilę i ponownie zweryfikuj poprzez Deployment status, czy czy procesy Tomcat są monitorowane
- Wejdź w szczegóły serwera (link), wybierz dowolny proces Tomcat i sprawdź, czy tym razem widać serwisy.
- Kliknij w Service na ikonografice i wybierz jeden z widocznych serwisów.
- Sprawdź jakie dane widać na ekranie szczegółów Serwisu.

3. Smartscape

- Z poziomu Smartscape sprawdź widoczność połączeń między procesami i serwisami.
- Ile procesów jest uruchomionych na monitorowanym hoście?

- Do jakiego serwisu/serwisów łączy się automatycznie wykryta aplikacja?

Ćwiczenie 2

Detekcja i analiza anomalii - 60 min (<https://wkf10640.apps.dynatrace.com/>)

Cel ćwiczenia:

Poznanie mechanizmów autowykrywania anomalii oraz metod detalicznej diagnozy problemów

1. Problemy (<https://wkf10640.apps.dynatrace.com/>)

- Otwórz aplikację Problems Classic z sekcji Observe and explore
- Wyświetl problemy za ostatnie 24h
- Posortuj problemy wg Ilości dotkniętych komponentów (kolumna Affected)
- Czy dla problemów, które agregują anomalie z więcej niż jednego komponentu Dynatrace wskazuje źródło problemu?
- Wejdź w szczegóły problemu, który ma największą liczbę dotkniętych komponentów
- Dlaczego Dynatrace stwierdził, że problem dotyczy aplikacji i użytkowników końcowych? Ilu użytkowników mogło być dotkniętych problemem?
- Wejdź w aplikację dotkniętą problemem i zweryfikuj, czy aplikacja była niedostępna podczas wystąpienia problemu.
- Powróć do szczegółów problemu (hint: poprzez górną belkę)
- Czy problem dotyczy niedostępności, błędów czy spowolnienia?
- Czy Dynatrace wskazał źródło problemu? Jeżeli w źródle problemu wskazana jest usługa – wywołaj analizę czasów odpowiedzi i sprawdź czy za spowolnienie odpowiada inna usługa, wolna baza danych czy kod wykonywalny usługi, która została wskazana jako źródło problemu.
- Znajdź metodę w kodzie, która odpowiada za spowolnienie – wybierz Service execution -> Method hotspots. Zmień domyślnie wyświetlanie z Call hierarchy na Hotspots.

2. Wolne usługi (<https://wkf10640.apps.dynatrace.com/>)

- Otwórz aplikację Services Classic z sekcji Application Observability
- Wyświetl usługi za ostatnie 30 minut i posortuj po Response Time median (malejąco). Ustaw boczny filter na Technology: Spring (hint: wykorzystaj View more)
- Znajdź mikroserwis: [eks][unguard] MicroblogController i wejdź w jego szczegóły
- Czy Dynatrace znalazł jakieś hotspoty?

- Wyświetl szczegóły czasów odpowiedzi klikając na wykres Response time
 - Przetłącz się z widoku mediany na 90 percentyl (hint: Slowest 10% nad wykresem) i kliknij na wykresie w miejsce, gdzie jest najwyższy czas odpowiedzi.
 - Sprawdź, z czego wynikają czasy odpowiedzi (hint: Response time hotspots)
 - Jeżeli hotspoty wskażą na długi czas w sekcji Interaction with services and queues kliknij na tą sekcję a następnie wyświetl przepływ transakcji przez poszczególne komponenty (hint: przycisk View service flow na dole ekranu).
 - Czy w przepływie transakcji wykorzystywana jest baza danych? Jeżeli tak, ile procent czasu zajmują odpowiedzi z bazy danych
 - Zaznacz na flow pierwszy od lewej element i z prawej strony wybierz View distributed traces aby wyświetlić pojedyncze requesty
 - Kliknij na pojedynczy request w polu Name – czy jesteś w stanie wskazać zapytania SQL, które były wykonane w ramach tego requestu?
3. Błędy w działaniu usług (<https://wkf10640.apps.dynatrace.com/>)
- Otwórz aplikację Services Classic z sekcji Application Observability
 - Wyświetl usługi za ostatnie 30 minut i posortuj po Response Time median (malejąco). Ustaw boczny filter na Technology: Spring (hint: wykorzystaj View more)
 - Znajdź mikroservis: [eks][unguard] MicroblogController i wejdź w jego szczegóły
 - Sprawdź czy na usłudze występują błędy – wykres Failure rate. Kliknij na niego
 - Na wykresie kliknij w miejsce, gdzie widoczne są błędne requesty
 - Zobacz szczegóły błędów: View details of failures
 - Jaki jest powód wystąpienia błędu? Wyświetl szczegóły wyjątku
 - Jakie requesty kończą się błędem?
 - Czy dla błędnych requestów pojawiają się wpisy w logu?
 - Skąd przyszedł błędny request? Sprawdź to wybierając przycisk Backtrace w prawym górnym rogu.



Ćwiczenie 3

Analiza podatności w bibliotekach

Cel ćwiczenia:

Poznanie możliwości Dynatrace w obszarze automatycznego wykrywania podatności w bibliotekach zewnętrznych oraz w kodzie własnym.

1. Konfiguracja początkowa – przygotowanie

- Przejdź do konfiguracji bezpieczeństwa aplikacji. Wejdź w Settings->Application Security
- Włącz wykrywanie podatności, Sekcja Vulnerability Analytics: General Settings
- Dla podatności w bibliotekach zewnętrznych Zaznacz Enable Third-party Vulnerability Analytics oraz wybierz z listy ustawienie Monitor
- Dla jakich technologii Dynatrace umożliwia monitorowanie podatności w bibliotekach? Zaznacz je wszystkie.
- Dla podatności w kodzie (Code-level Vulnerability Analytics) ustaw przełącznik w pozycji Enabled.
- Włącz monitorowanie dla Java, dla .NET pozostaw opcję nie uruchomioną.

2. Wykrywanie podatności w bibliotekach

- Smartscape
 - Wejdź z menu w Smartscape Topology i zmień domyślne ustawienie pokazywania anomalii z Show problems na Show third-party vulnerabilities.
 - Zweryfikuj, na poziomie jakich obiektów oznaczane są komponenty z podatnościami (hint: przełączaj się między poziomami: Applications, Services, Processes, Hosts). Dla jakich obiektów prezentowane są podatności?
- Sygnalizowanie wykrycia podatności
 - Zweryfikuj, ile podatności o statusie High lub/i Critical zostało wykrytych
 - Przejdź do listy wykrytych podatności poprzez sygnalizator problemów. Jakie filtry zostały ustawione?
- Analiza wykrytych podatności – filtrowanie najważniejszych problemów
 - Ile podatności jest obecnie otwartych? Ile z nich jest w statusie High lub Critical.
 - Odfiltruj tylko te, które mogą być wykorzystane poprzez atak z publicznego internetu. Jak zmieniła się liczba istotnych podatności?
 - Dodatkowo, odfiltruj tylko te, dla których są gotowe publiczne exploity. Jak zmieniła się liczba istotnych podatności?

- Sprawdź, jakie inne opcje filtrowania pozwalają znacząco poprawić priorytetyzację wykrytych podatności.
- Analiza wykrytych podatności – szczegóły najważniejszych problemów
 - Wybierz podatność o największym DSS (Davis Security Score) – usuń wcześniej filtry
 - Zweryfikuj czy DSS różni się od CVSS dla tej podatności. Zweryfikuj co wpłynęło na obecność lub brak różnicy.
 - Sprawdź, na czym polega podatność i jak ją można załatać.
 - Czy podatność ma taki sam priorytet na wszystkich procesach?
- Analiza wykrytych podatności – quick win
 - Co można poprawić, aby załatać jak najwięcej podatności?
- 3. Wykrywanie podatności w kodzie własnym
 - Przegląd podatności
 - Przejdź na środowisko: <https://wkf10640.apps.dynatrace.com/>
 - Otwórz aplikację Code-level vulnerabilities z sekcji Application Security
 - Jakie typy podatności są wychwytywane przez Dynatrace?
 - Wejdź w szczegóły jednej z podatności i zweryfikuj, w którym miejscu w kodzie istnieje podatność
- 4. Wykrywanie ataków – konfiguracja (<https://dynatrace.omnilogy.pl>)
 - Application Protection
 - Przejdź do konfiguracji bezpieczeństwa aplikacji. Wejdź w Settings->Application Security
 - Włącz wykrywanie ataków, Sekcja Application Protection: General Settings -> Enable
 - Dla Javy włącz opcję monitorowania i blokowania ataków
 - Dla .NET włącz opcję wyłącznie detekcji ataków, bez blokowania
- 5. Wykrywanie ataków – analiza (<https://wkf10640.apps.dynatrace.com/>)
 - Przegląd ataków
 - Otwórz aplikację Attacks z sekcji Application Security i wyświetl statystyki za ostatnie 24h
 - Ile ataków zostało wykrytych a ile zablokowanych?
 - Wyfiltruj ataki typu Command injection i wyświetl jeden z nich
 - Na jakim procesie miał miejsce ten atak
 - Jaki był punkt wejścia do systemu?
 - Na czym polegał atak
 - Zweryfikuj w logu, jaki był skutek wykonania ataku, który nie był zablokowany przez Dynatrace

Ćwiczenie 1

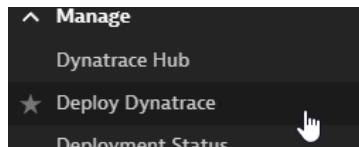
Instalacja i uruchomienia monitoringu - 30 min

Cel ćwiczenia:

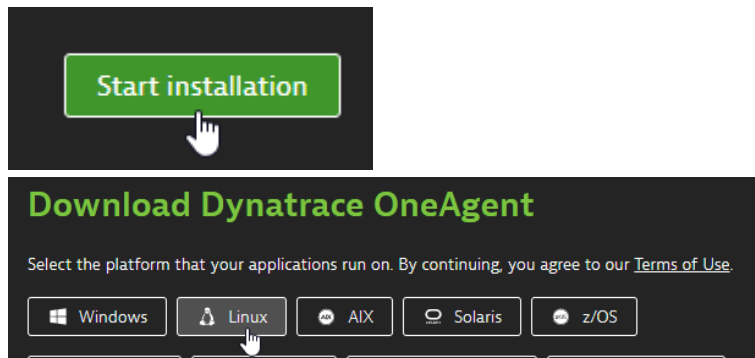
Poznanie mechanizmów instalacji agenta Dynatrace oraz automatycznego skonfigurowania monitoringu dla aplikacji działającej na serwerze.

4. Instalacja agenta:

- Wybierz „Manage” -> „Deploy Dynatrace” z lewego menu

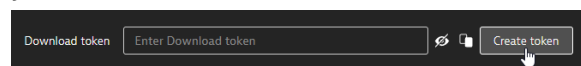


- Wybierz „Start installation” oraz odpowiedni system operacyjny

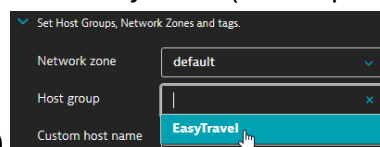


- Na kreatorze instalacji:

- utwórz token,
- ustaw Monitoring mode na Full stack,
- ustaw Host Group – wartość EasyTravel (hint: wpisz wartość i



wybierz ją po wpisaniu)



- zverifyfikuj, jak zmodyfikowało się polecenie instalacyjne w pkt.4

```
/bin/sh Dynatrace-OneAgent-Linux-1.309.66.20250401-150134.sh --set-monitoring-mode=fullstack --set-app-log-content-access=true --set-host-group=EasyTravel
```

- Z poziomu linux jako root pobierz agenta poleceniem wget

```
2. Download the installer using this command on the target host.

wget -O Dynatrace-OneAgent-Linux-1.309.66.20250401-150134.sh
"https://dynatrace.omnily.pl/e/421fed4-2271-433e-8d7d-8e35bfaa8e4a/api/v1/deployment/installer/agent/unix/default/latest?arch=x86" --header="Authorization: Api-Token
....."
....."
```

- Sprawdź podpis pobranej paczki

3. Verify signature:

```
wget https://ca.dynatrace.com/dt-root.cert.pem ; ( echo 'Content-Type: multipart/signed; protocol="application/x-pkcs7-signature"; micalg="sha-256"; boundary="--SIGNED-INSTALLER"' ; echo ; echo ; echo '----SIGNED-INSTALLER' ; cat Dynatrace-OneAgent-Linux-1.309.66.20250401-150134.sh ) | openssl cms -verify -CAfile dt-root.cert.pem > /dev/null
```

- Zainstaluj agenta

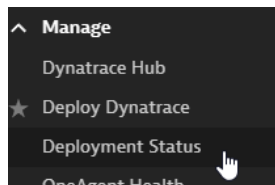
4. Run the installer with root rights.

```
/bin/sh Dynatrace-OneAgent-Linux-1.309.66.20250401-150134.sh --set-monitoring-mode=fullstack --set-app-log-content-access=true --set-host-group=EasyTravel
```



Hint – używaj przycisku kopiowania poleceń zamiast zaznaczania i Ctrt+C

- Zweryfikuj poprzez Deployment status,



czy serwer pojawił się w Dynatrace.

Showing 1 host.

Host unit consumption: 0.25 (for 1 total host)

<input type="checkbox"/>	OS	Host name
<input type="checkbox"/>		OMNI13-vm.jsxtrew1ibtu3kmuf5jyc1eiaa.ax.internal.cloudapp.net

Host is monitored. Dynatrace OneAgent is up-to-date on version 1.309.66.
Host units: 0.25 Host group: [EasyTravel](#) IP addresses: 10.0.1.4 Monitoring mode: Full stack Currently used ActiveGates none (direct communication)

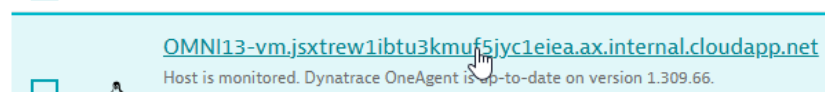
Details



Rozwiń szczegóły i sprawdź, czy jakieś procesy są monitorowane

No automatically detected processes that support Service insights.

- Wejdź w szczegóły serwera (link) i zweryfikuj zakres danych infrastrukturalnych widocznych na ekranie



Hosts [OMNI13-vm.jsxtrew1ibtu3kmuf5jyc1eiaa.ax.internal.cloudapp.net](#)

OMNI13-vm.jsxtrew1ibtu3kmuf5jyc1eiaa.ax.internal.cloudapp.net
Overview of the Host
Monitoring mode Full stack Host group [EasyTravel](#)

Properties and tags ✓ No problems ⚠ Not analyzed 98.32 % Availability 0 SLOs 👤 Owners

✓ Uptime 3.37 h	✓ CPU 0.459 %	✓ Available memory 83.6 %	✓ Network 25.3 kbit/s
------------------------------	----------------------------	--	------------------------------------

Incoming connections
Contains 0 Hosts.

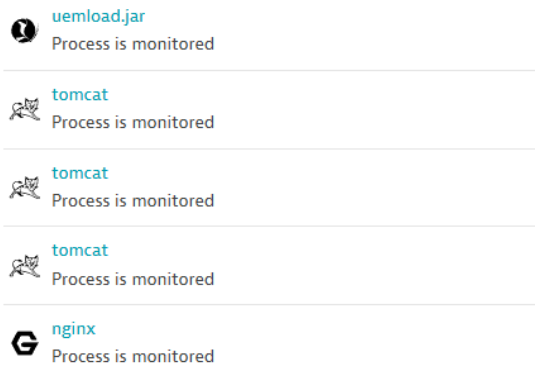
Outgoing connections
Contains 0 Hosts.

5. Uruchom monitoring aplikacyjny

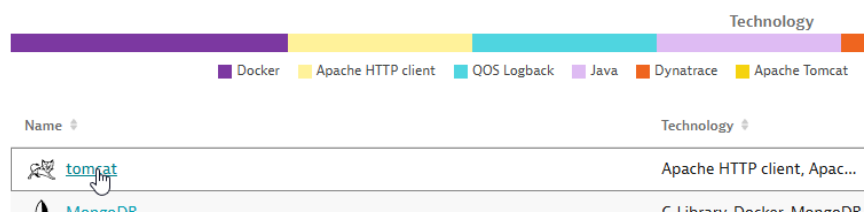
- Z poziomu linux jako root uruchom aplikację docker-compose up -d
- Poczekaj chwilę i ponownie zweryfikuj poprzez Deployment status, czy procesy Tomcat są monitorowane

Uwaga – to trochę potrwa. Czekać na pojawienie się procesów wykonaj punkt Konfiguracja początkowa – przygotowanie z ćwiczenia 3.

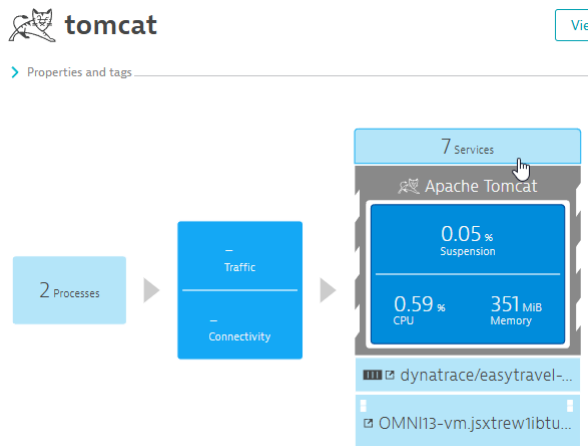
Automatically detected processes that support Service insights:



- Wejdź w szczegóły serwera, wybierz dowolny proces Tomcat



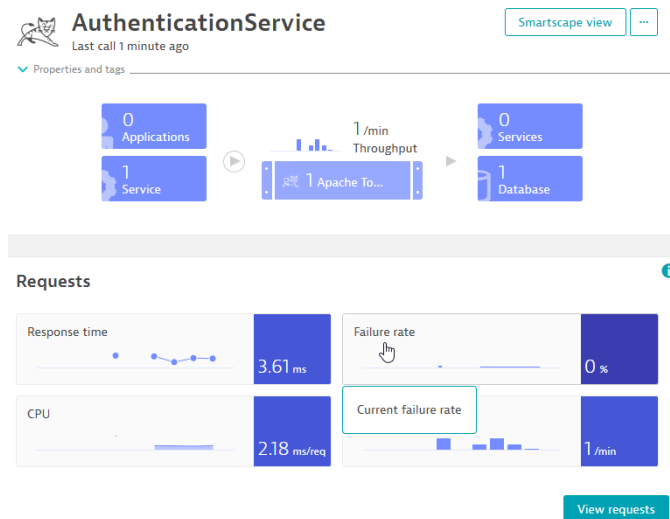
i sprawdź, czy na ikonografice widać serwisy.



- Kliknij w Service na ikonografice i wybierz jeden z widocznych serwisów.

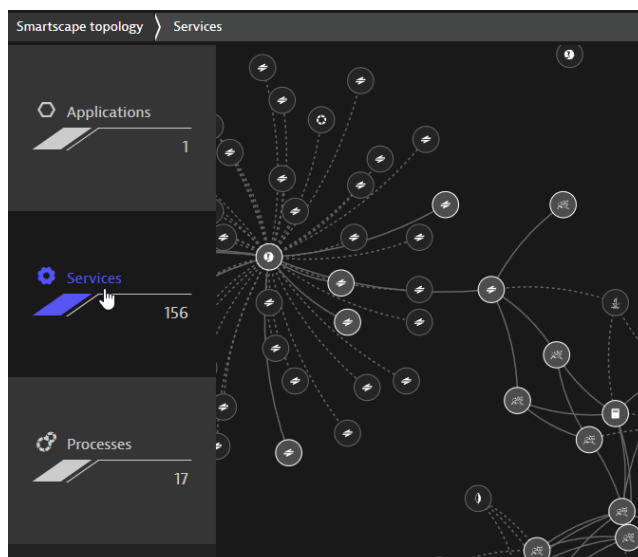
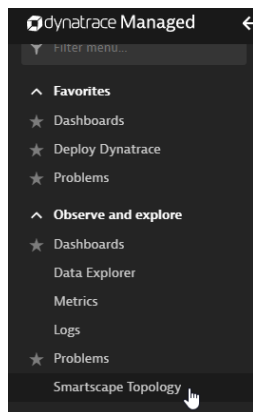


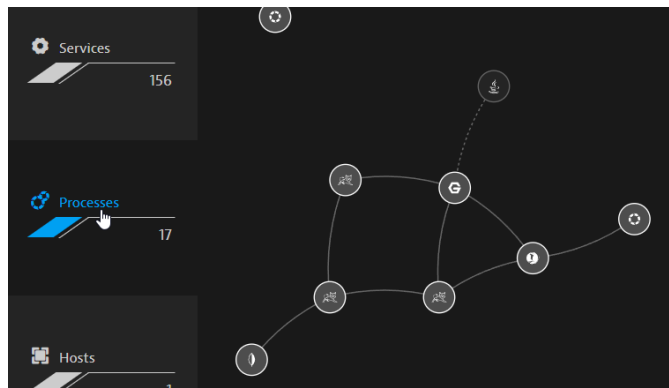
- Sprawdź jakie dane widać na ekranie szczegółów Serwisu.



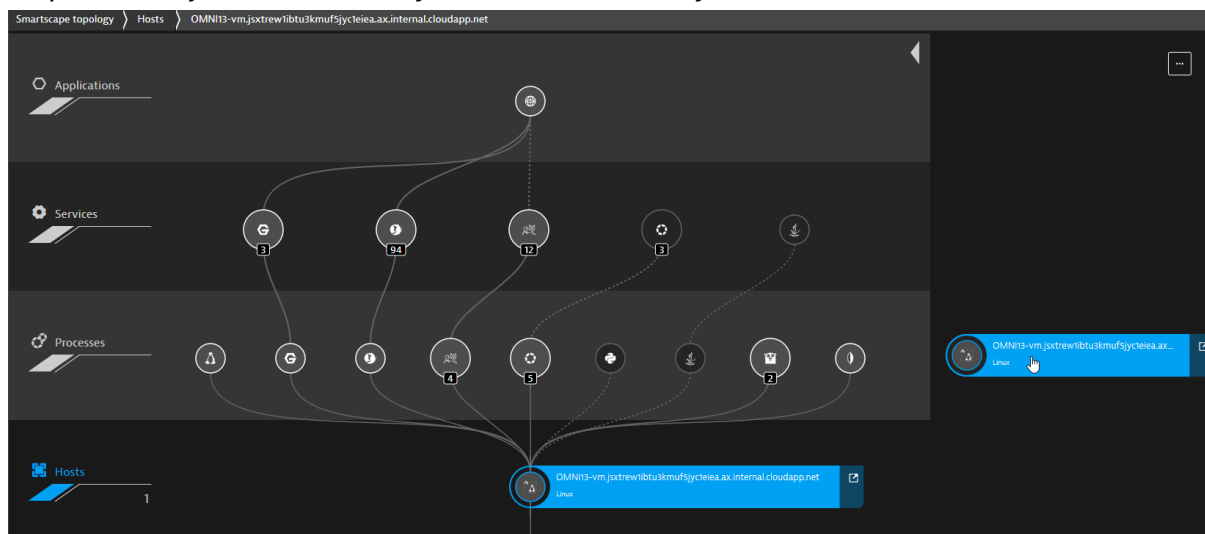
6. Smartscape

- Z poziomu Smartscape sprawdzić widoczność połączeń między procesami i serwisami.

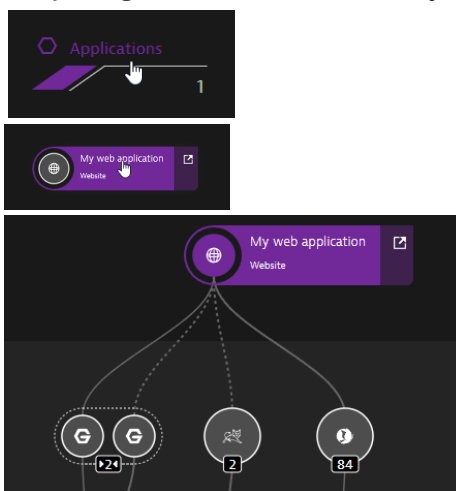




- Ile procesów jest uruchomionych na monitorowanym gościu?



- Do jakiego serwisu/serwisów łączy się automatycznie wykryta aplikacja?



Ćwiczenie 2

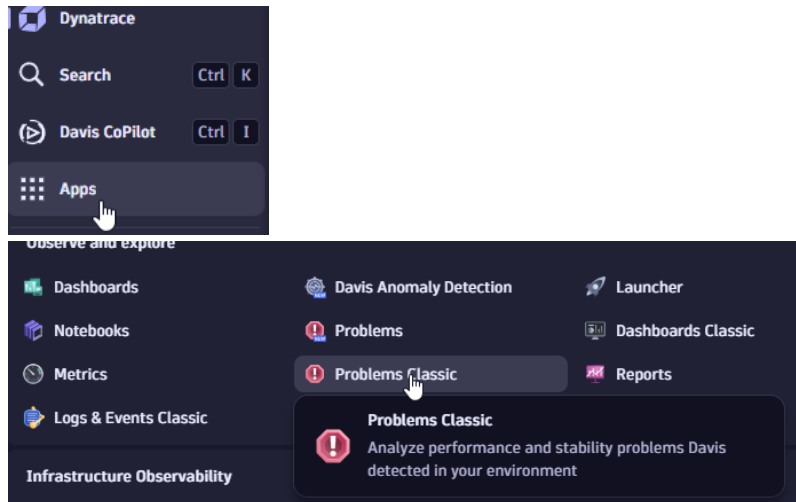
Detekcja i analiza anomalii - 60 min (<https://wkf10640.apps.dynatrace.com/>)

Cel ćwiczenia:

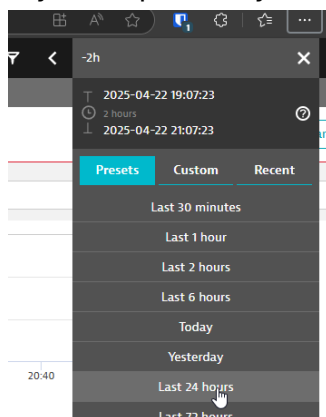
Poznanie mechanizmów autowykrywania anomalii oraz metod detalicznej diagnozy problemów

4. Problemy (<https://wkf10640.apps.dynatrace.com/>)

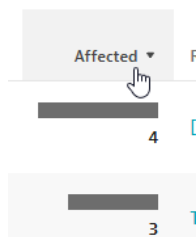
- Otwórz aplikację Problems Classic z sekcji Observe and explore



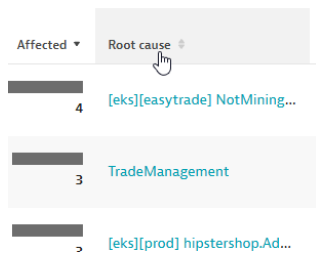
- Wyświetl problemy za ostatnie 24h



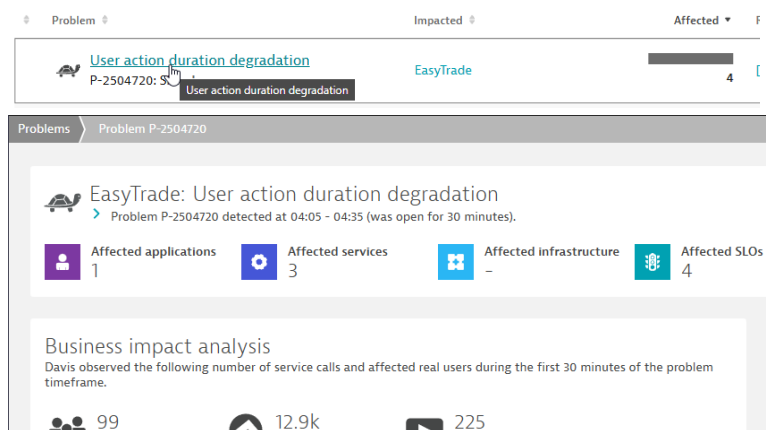
- Posortuj problemy wg Ilości dotkniętych komponentów (kolumna Affected)



- Czy dla problemów, które agregują anomalie z więcej niż jednego komponentu Dynatrace wskazuje źródło problemu?



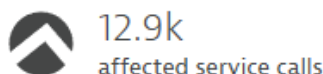
- Wejść w szczegóły problemu, który ma największą liczbę dotkniętych komponentów



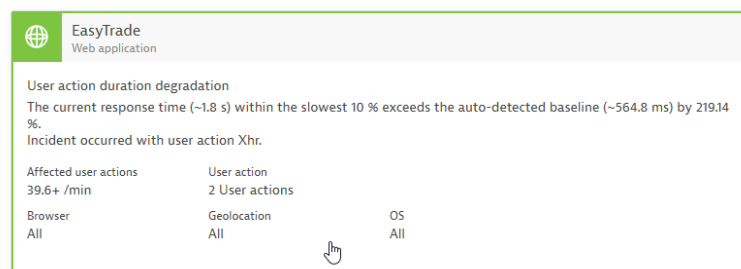
- Dlaczego Dynatrace stwierdził, że problem dotyczy aplikacji i użytkowników końcowych? Ilu użytkowników mogło być dotkniętych problemem?

Business impact analysis

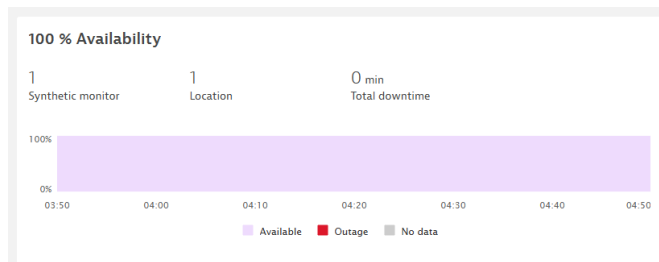
Davis observed the following number of service calls and affected users during the first 30 minutes of the problem timeframe.



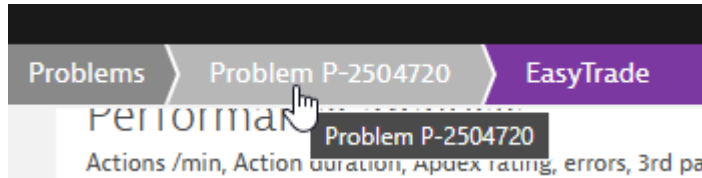
1 impacted application
39.6+ User actions per minute impacted



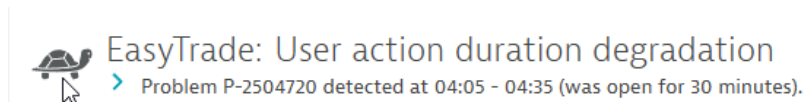
- Wejść w aplikację dotkniętą problemem i zweryfikuj, czy aplikacja była niedostępna podczas wystąpienia problemu.



- Powrót do szczegółów problemu (hint: poprzez górną belkę)



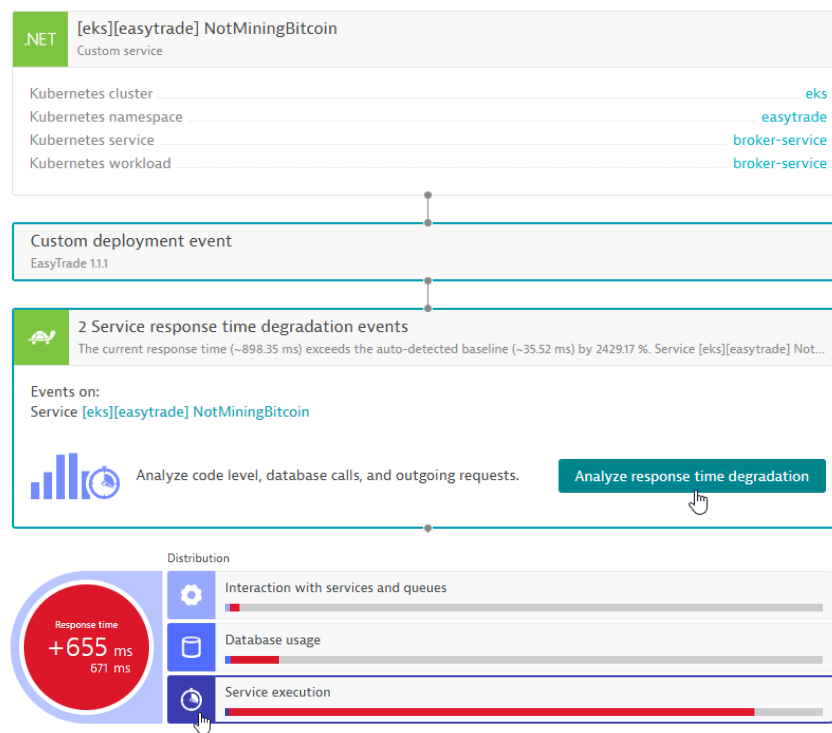
- Czy problem dotyczy niedostępności, błędów czy spowolnienia?



- Czy Dynatrace wskazał źródło problemu? Jeżeli w źródle problemu wskazana jest usługa – wywołaj analizę czasów odpowiedzi i sprawdź czy za spowolnienie odpowiada inna usługa, wolna baza danych czy kod wykonywalny usługi, która została wskazana jako źródło problemu.

Root cause

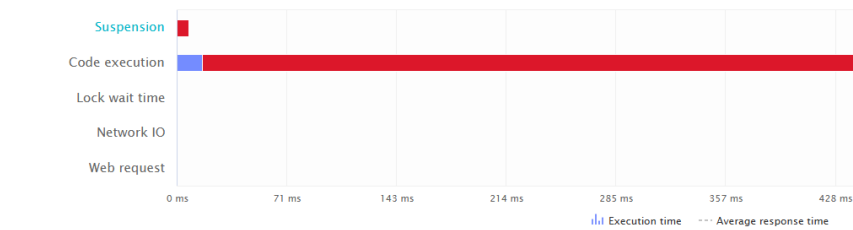
Based on our dependency analysis all incidents have the same root cause



- Znajdź metodę w kodzie, która odpowiada za spowolnienie – wybierz Service execution -> Method hotspots. Zmień domyślnie wyświetlanie z Call hierarchy na Hotspots.



Breakdown of service execution time



View method hotspots

Call hierarchy Hotspots

Top hotspots

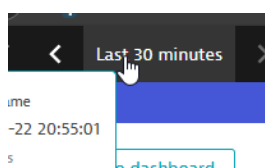
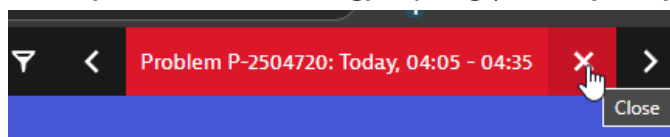
Method	Contribution breakdown	Overall contribution
HighCpuUsageMiddleware.NorthMinimaBitness.com		78.8 %
.NET EasyTradeBrokerService.ProblemPattern		
HighCpuUsage		
HighCpuUsageMiddleware.NorthMinimaBitness.com		

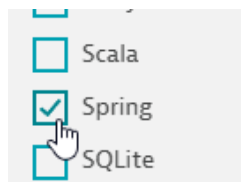
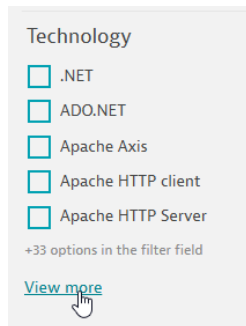
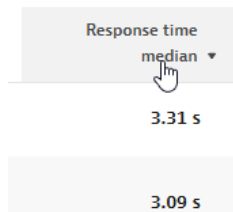
5. Wolne usługi (<https://wkf10640.apps.dynatrace.com/>)

- Otwórz aplikację Services Classic z sekcji Application Observability

The screenshot shows the Dynatrace Application Observability interface. The 'Services Classic' section is highlighted, showing a list of services. The 'Services' section is expanded, showing a table of services with columns for Name, Response time median, Slowest 10%, Failure rate, Requests, and Actions. The table lists several services, including 'jako-playground[ingress-nginx]-80,443' and 'jako-playground[ingress-nginx]-35246'.

- Wyświetl usługi za ostatnie 30 minut (hint: zamknij poprzez X timeframe z problemem) i posortuj po Response Time median (malejąco). Ustaw boczny filter na Technology: Spring (hint: wykorzystaj View more)

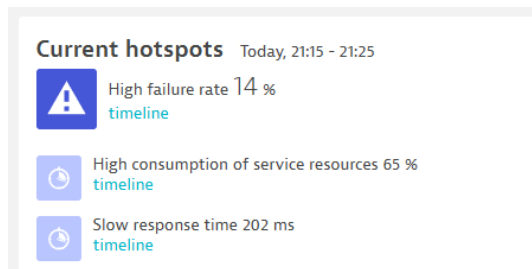




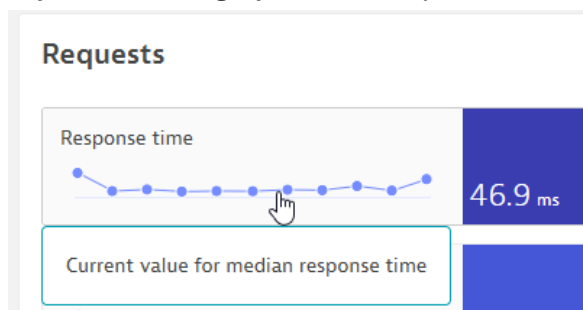
- Znajdź mikroservis: [eks][unguard] MicroblogController i wejdź w jego szczegóły

[eks][unguard] MicroblogController	70.1 ms	144 ms	13.3 %
SpringBoot.org.dynatrace.R... [eks][unguard] MicroblogController blog-service-*			

- Czy Dynatrace znalazł jakieś hotspoty?



- Wyświetl szczegóły czasów odpowiedzi klikając na wykres Response time

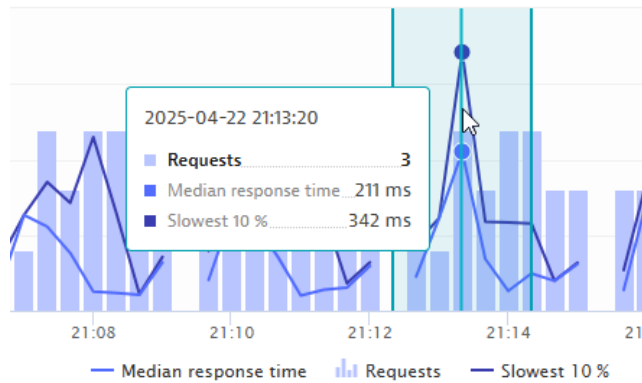


- Przetłącz się z widoku mediany na 90 percentyl (hint: Slowest 10% nad wykresem) i kliknij na wykresie w miejsce, gdzie jest najwyższy czas odpowiedzi.

Details ?

Response time

View Median Slowest 10 % Slowest 5 %

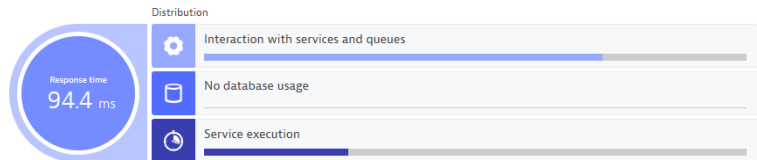


- Sprawdź, z czego wynikają czasy odpowiedzi (hint: Response time hotspots)

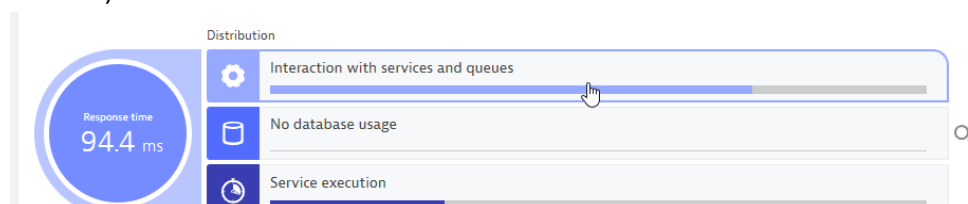


Analyze code level, database calls, and outgoing requests.

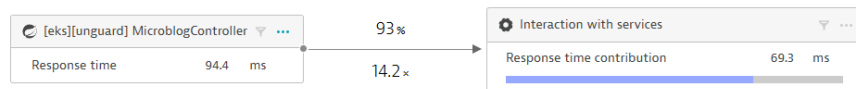
View response time hotspots



- Jeżeli hotspoty wskażą na długi czas w sekcji Interaction with services and queues kliknij na tą sekcję a następnie wyświetl przepływ transakcji przez poszczególne komponenty (hint: przycisk View service flow na dole ekranu).

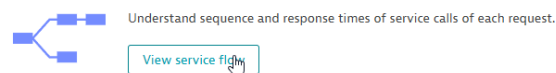


Interaction with services and queues

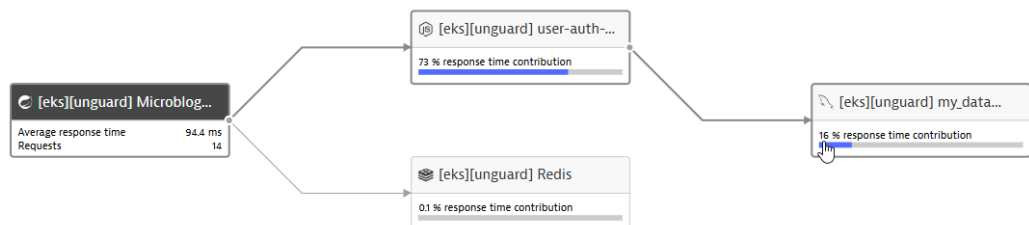


93% of the requests call other services (averaging 14.2 calls per request).

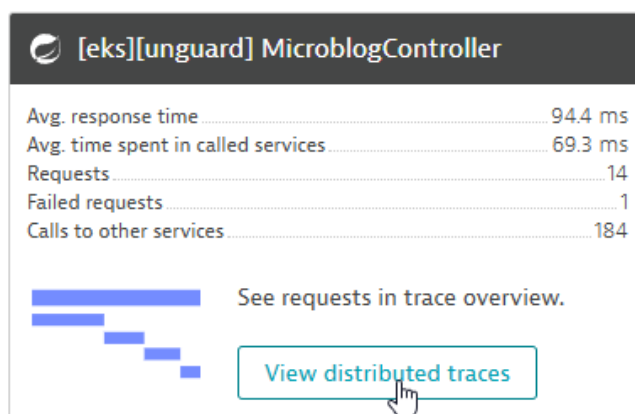
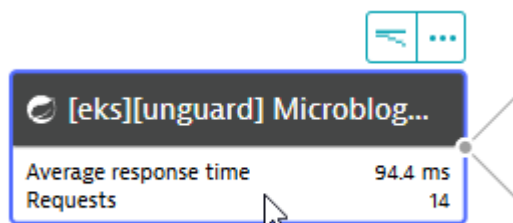
Analyze '[eks][unguard] MicroblogController' requests

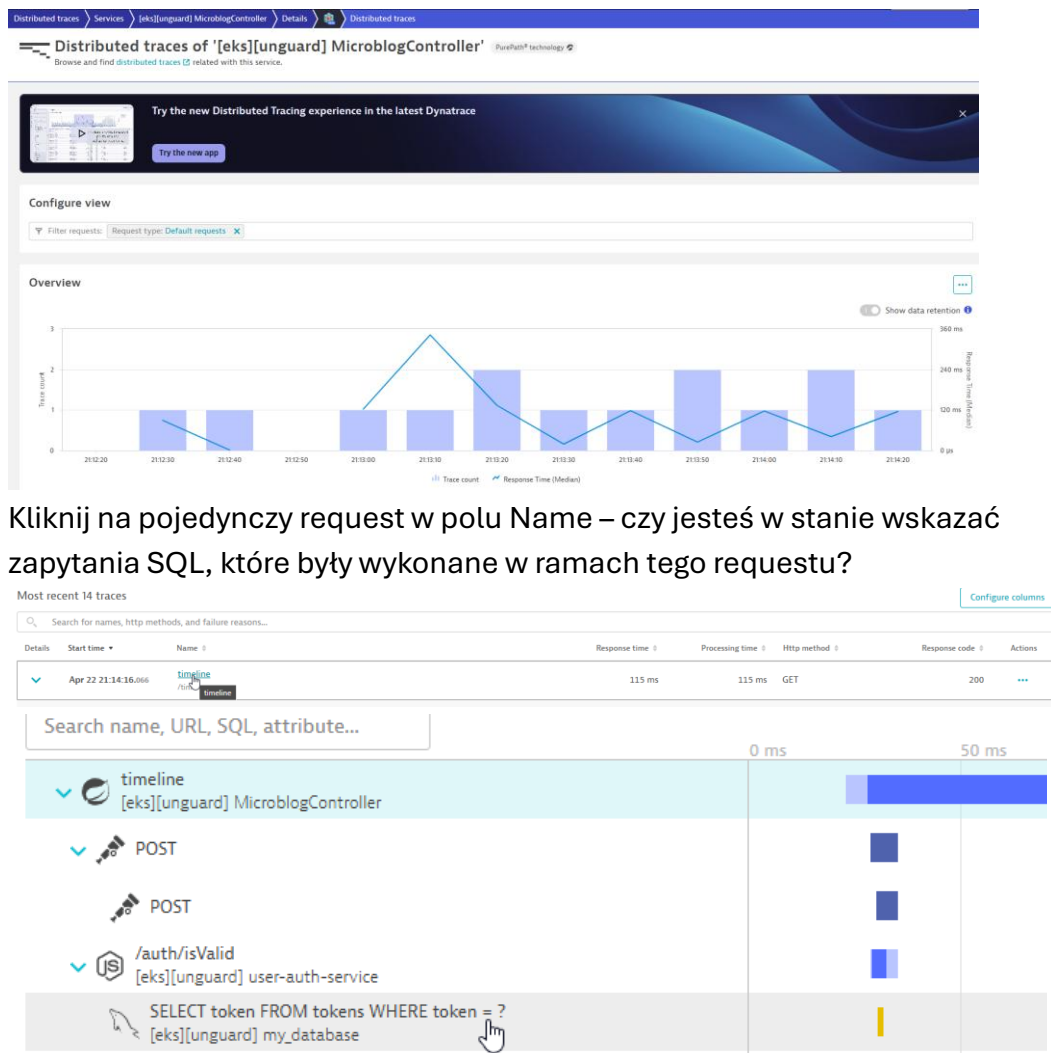


- Czy w przepływie transakcji wykorzystywana jest baza danych? Jeżeli tak, ile procent czasu zajmują odpowiedzi z bazy danych



- Zaznacz na flow pierwszy od lewej element i z prawej strony wybierz View distributed traces aby wyświetlić pojedyncze requesty





- Kliknij na pojedynczy request w polu Name – czy jesteś w stanie wskazać zapytania SQL, które były wykonane w ramach tego requestu?

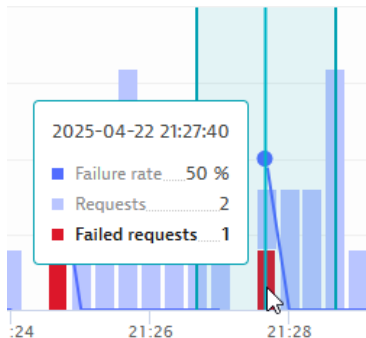
6. Błędy w działaniu usług (<https://wkf10640.apps.dynatrace.com/>)

- Otwórz aplikację Services Classic z sekcji Application Observability
- Wyświetl usługi za ostatnie 30 minut i posortuj po Response Time median (malejąco). Ustaw boczny filter na Technology: Spring (hint: wykorzystaj View more)
- Znajdź mikroservis: [eks][unguard] MicroblogController i wejdź w jego szczegóły
- Sprawdź czy na usłudze występują błędy – wykres Failure rate. Kliknij na niego





- Na wykresie kliknij w miejsce, gdzie widoczne są błędne requesty



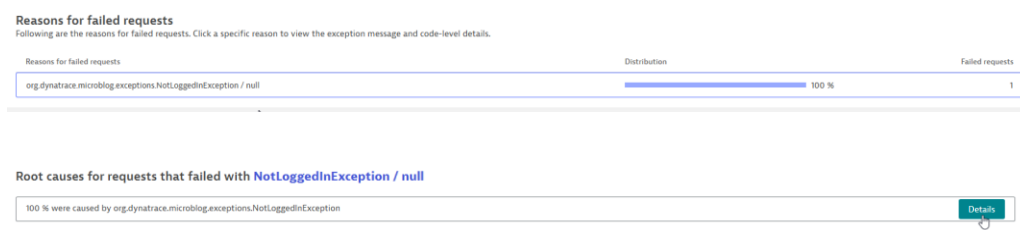
- Zobacz szczegóły błędów: View details of failures



Understand which requests fail and the underlying root causes and errors.


[View details of failures](#)

- Jaki jest powód wystąpienia błędu? Wyświetl szczegóły wyjątku



- Jakie requesty kończą się błędem?


Requests that failed with `NotLoggedInException / null`

Name 

timeline

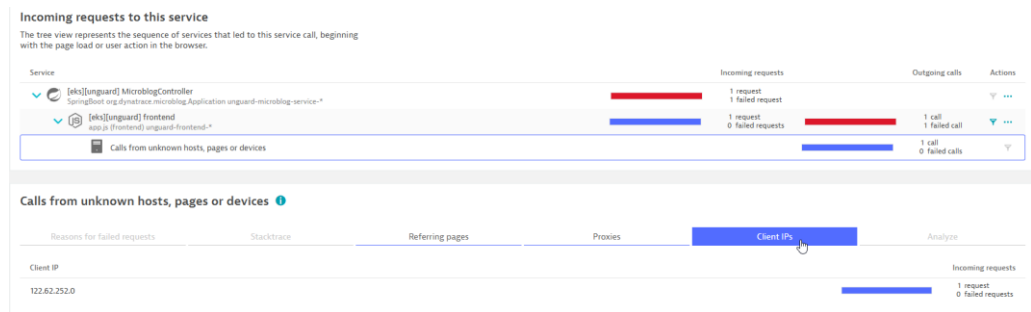
- Czy dla błędnych requestów pojawiają się wpisy w logu?

Related error logs

Filter log by  content or status

- Skąd przyszedł błędny request? Sprawdź to wybierając przycisk Backtrace w prawym górnym rogu.





Ćwiczenie 3

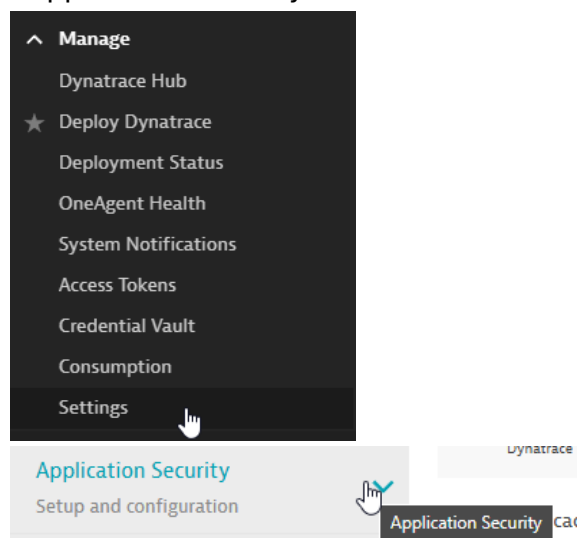
Analiza podatności w bibliotekach

Cel ćwiczenia:

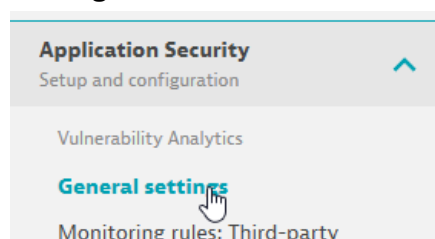
Poznanie możliwości Dynatrace w obszarze automatycznego wykrywania podatności w bibliotekach zewnętrznych oraz w kodzie własnym.

6. Konfiguracja początkowa – przygotowanie

- Przejdź do konfiguracji bezpieczeństwa aplikacji. Wejdź w Settings->Application Security



- Włącz wykrywanie podatności, Sekcja Vulnerability Analytics: General Settings

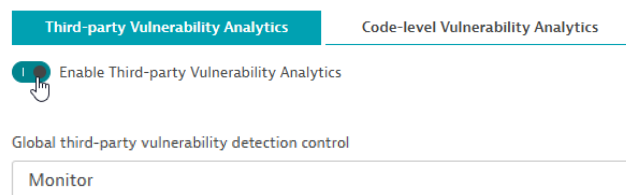




- Dla podatności w bibliotekach zewnętrznych Zaznacz Enable Third-party Vulnerability Analytics oraz wybierz z listy ustawienie Monitor

Vulnerability Analytics: General settings

Automated [Runtime Vulnerability Analytics](#) helps you quickly and completely understand each detected vulnerability in your environment and how to remediate it, allowing you to prioritize which vulnerabilities to fix first. Note: Enabling Third-party or Code-level Vulnerability Analytics consumes Application Security units. For details, see the [Application Security Monitoring documentation](#).

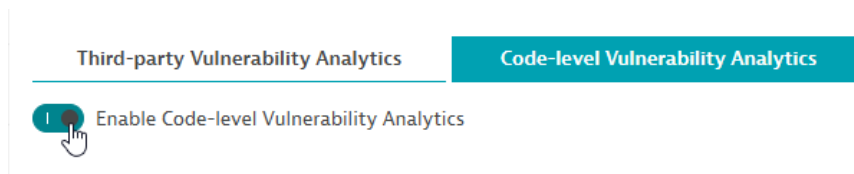


- Dla jakich technologii Dynatrace umożliwia monitorowanie podatności w bibliotekach? Zaznacz je wszystkie.

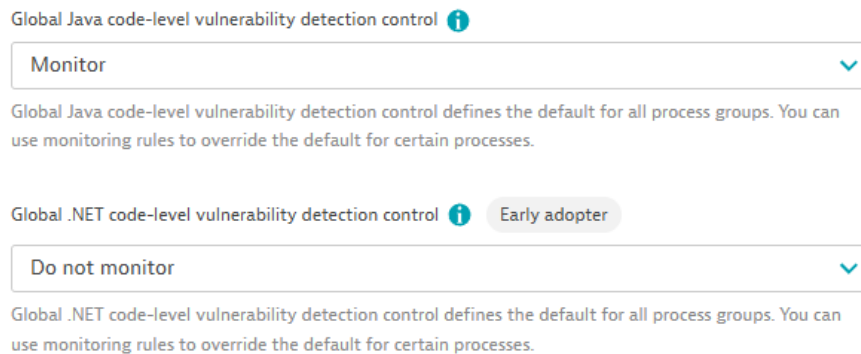
Technologies
Vulnerability Analytics can be enabled/disabled per supported technology.



- Dla podatności w kodzie (Code-level Vulnerability Analytics) ustaw przełącznik w pozycji Enabled.

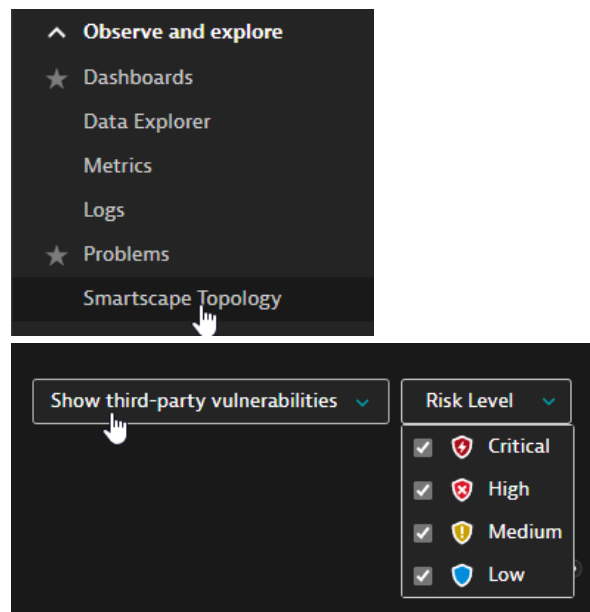


- Włącz monitorowanie dla Java, dla .NET pozostaw opcję nie uruchomioną.

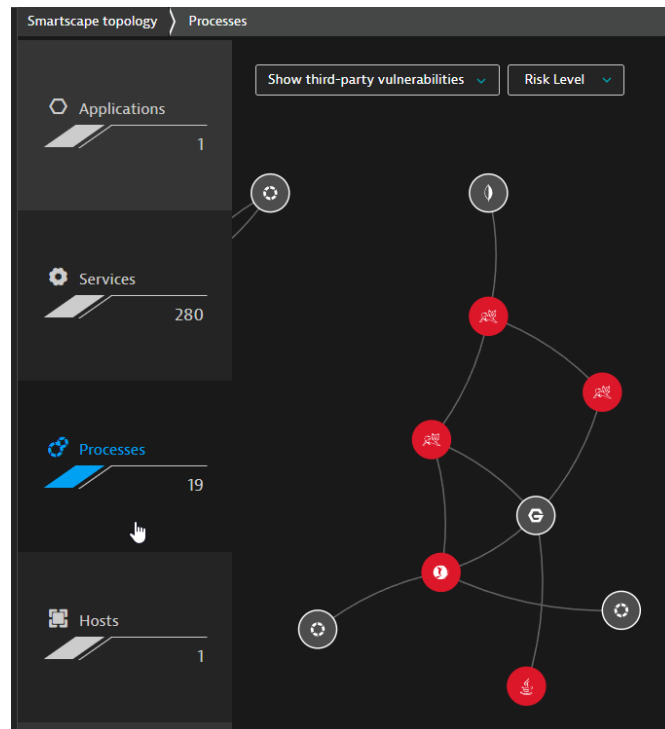


7. Wykrywanie podatności w bibliotekach

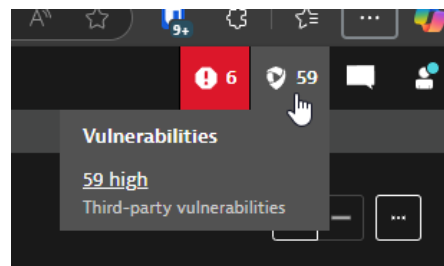
- Smartscape
 - Wejdź z menu w Smartscape Topology i zmień domyślne ustawienie pokazywania anomalii z Show problems na Show third-party vulnerabilities.



- Zweryfikuj, na poziomie jakiś obiektów oznaczane są komponenty z podatnościami (hint: przełączaj się między poziomami: Applications, Services, Processes, Hosts). Dla jakich obiektów prezentowane są podatności?



- Sygnalizowanie wykrycia podatności
 - Zweryfikuj, ile podatności o statusie High lub/i Critical zostało wykrytych



- Przejdź do listy wykrytych podatności poprzez sygnalizator problemów. Jakie filtry zostały ustawione?

Third-party vulnerabilities

Third-party vulnerabilities Insights by snyk
Detect vulnerabilities in your environment at runtime.

89 Open	0 Critical	59 High	Monitor
------------	---------------	------------	---------

Davis® Security Advisor

Top recommended fixes

Below are the most effective actions you can take right now to improve the security of your environment.

Upgrade jackson-databind
Impacts 43 high (46 vulnerabilities total)
[Add as filter](#)

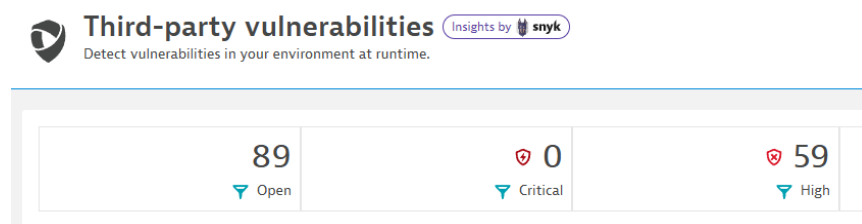
Upgrade tomcat-catalina
Impacts 4 high (8 vulnerabilities total)
[Add as filter](#)

Upgrade to
Impacts 4 high
[Add as filter](#)

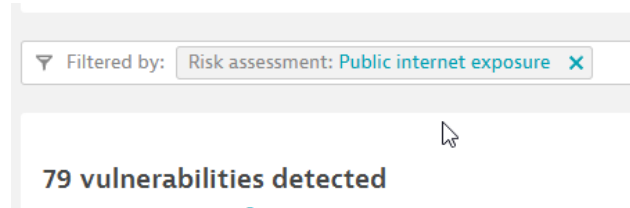
Filtered by: Risk level: High x Status: Open x

- Analiza wykrytych podatności – filtrowanie najważniejszych problemów

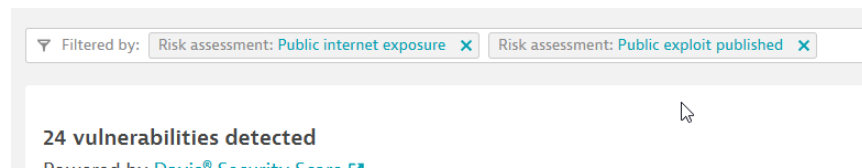
- Ile podatności jest obecnie otwartych? Ile z nich jest w statusie High lub Critical?



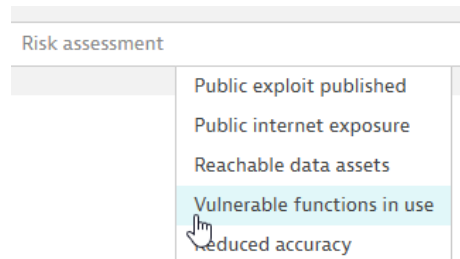
- Odfiltruj tylko te, które mogą być wykorzystane poprzez atak z publicznego internetu. Jak zmieniła się liczba istotnych podatności?



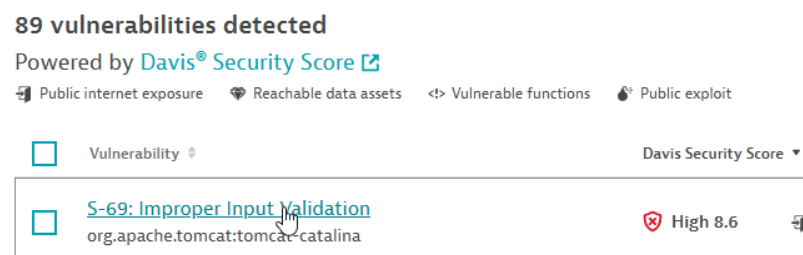
- Dodatkowo, odfiltruj tylko te, dla których są gotowe publiczne exploity. Jak zmieniła się liczba istotnych podatności?



- Sprawdź, jakie inne opcje filtrowania pozwalają znacząco poprawić priorytetyzację wykrytych podatności.



- Analiza wykrytych podatności – szczegóły najważniejszych problemów
 - Wybierz podatność o największym DSS (Davis Security Score) – usuń wcześniej filtry



- Zweryfikuj czy DSS różni się od CVSS dla tej podatności.
Zweryfikuj co wpłynęło na obecność lub brak różnicy.

8.6 High risk vulnerability
Davis Security Score

8.6 High risk vulnerability
CVSS as a base

Analyzed with Davis

Public internet exposure

Exposure	Impact on score	Risk level
Public network	No change	High risk

Reachable data assets

Affected	Impact on score	Risk level
None within range	No change	High risk

- Sprawdź, na czym polega podatność i jak ją można załatać.

Vulnerability details

[org.apache.tomcat:tomcat-catalina](#) is a Tomcat Servlet Engine Core Classes and Standard implementations.

Affected versions of this package are vulnerable to Improper Input Validation due to the improper parsing of HTTP trailer headers. An attacker can manipulate the server into treating a single request as multiple requests by sending a trailer header that exceeds the header size limit. This could lead to request smuggling when the server is behind a reverse proxy.

For more information visit [SNYK](#)

Fix recommendation

Upgrade `org.apache.tomcat:tomcat-catalina` to version 8.5.96, 9.0.83, 10.1.16, 11.0.0-M10 or higher.

- Czy podatność ma taki sam priorytet na wszystkich procesach?

Most affected process groups

Process group	Status
tomcat 1/1 process affected	Affected
tomcat 1/1 process affected	Affected
tomcat 1/1 process affected	Affected
tomcat 1/1 process affected	Affected


Public internet exposure

- Analiza wykrytych podatności – quick win
 - Co można poprawić, aby załatać jak najwięcej podatności?


Davis® Security Advisor

Top recommended fixes

Below are the most effective actions you can take right now to improve the security of your environment.



Upgrade jackson-databind
Impacts 43 high (46 vulnerabilities total)
[Add as filter](#)

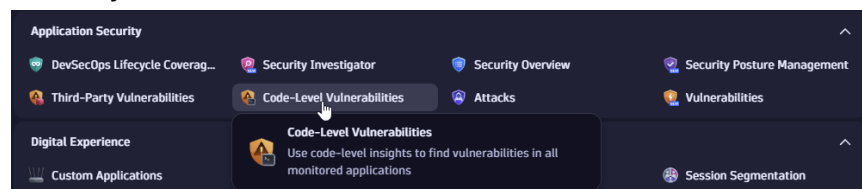


Upgrade tomcat-catalina
Impacts 4 high (8 vulnerabilities total)
[Add as filter](#)

8. Wykrywanie podatności w kodzie własnym

(<https://wkf10640.apps.dynatrace.com/>)

- Przegląd podatności
 - Przejdź na środowisko: <https://wkf10640.apps.dynatrace.com/>
 - Otwórz aplikację Code-level vulnerabilities z sekcji Application Security



- Jakie typy podatności są wychwytywane przez Dynatrace?
- Wejdź w szczegóły jednej z podatności i zweryfikuj, w którym miejscu w kodzie istnieje podatność

Vulnerability Risk level ▾

S-360: SQL injection at MembershipController+<SetMembershipStatus>d_4.MoveNext... Critical

MembershipService.dll unguarded Membership-service-*

Code location

```
MembershipService.Controllers.MembershipController+<SetMembershipStatus>d__4.MoveNext()
```

Vulnerable function

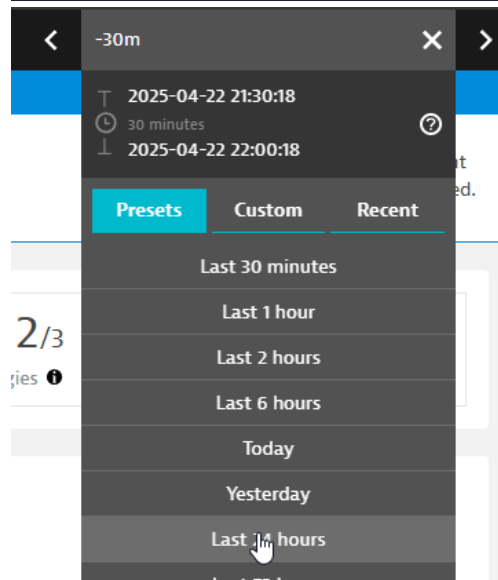
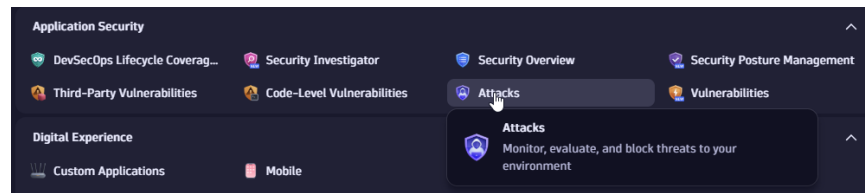
```
System.Data.Common.DbCommand.ExecuteNonQueryAsync()
```

9. Wykrywanie ataków – konfiguracja (<https://dynatrace.omnilog.pl>)

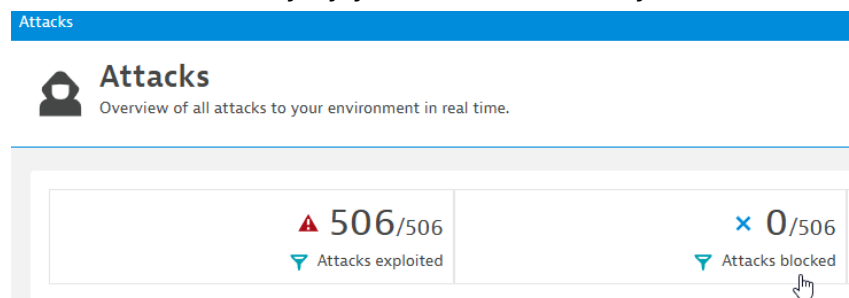
- Application Protection*
 - Przejdź do konfiguracji bezpieczeństwa aplikacji. Wejdź w *Settings->Application Security*
 - Włącz wykrywanie ataków, Sekcja *Application Protection: General Settings -> Enable*
 - Dla Javy włącz opcję monitorowania i blokowania ataków
 - Dla .NET włącz opcję wyłącznie detekcji ataków, bez blokowania

10. Wykrywanie ataków – analiza (<https://wkf10640.apps.dynatrace.com/>)

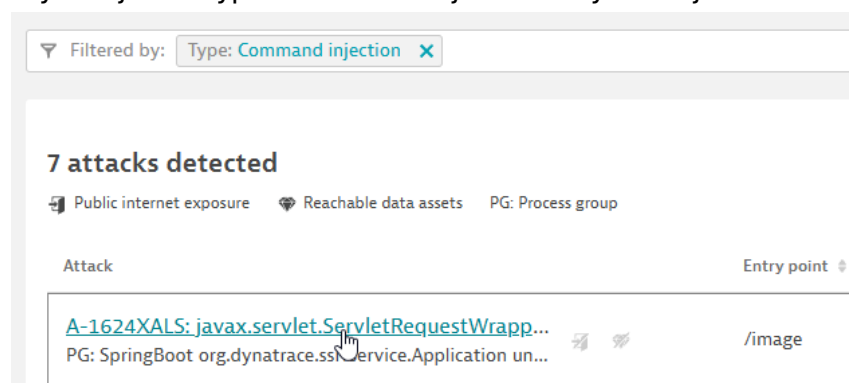
- Przegląd ataków
 - Otwórz aplikację Attacks z sekcji Application Security i wyświetl statystyki za ostatnie 24h



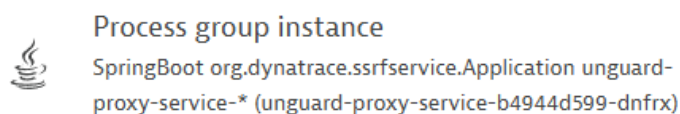
- Ile ataków zostało wykrytych a ile zablokowanych?



- Wyfiltruj ataki typu Command injection i wyświetl jeden z nich



- Na jakim procesie miał miejsce ten atak



- Jaki był punkt wejścia do systemu?

Entry point

URL path
/image

Entry point function

```
javax.servlet.ServletRequestWrapper.getParameterValues(String)
```

Payload

HTTP parameter url

HTTP parameter value example.com && whoami #

- Na czym polegał atak

Command

 Highlighted text indicates malicious input.

```
/bin/sh -c curl --silent -S example.com && whoami # --max-time 10 --output /tmp/img-1745351530131.jpg
```

- Zweryfikuj w logu, jaki był skutek wykonania ataku, który nie był zablokowany przez Dynatrace

```
java.io.FileNotFoundException: /tmp/img-1745351530131.jpg (No such file or directory) at java.io.FileInputStream.open(Native Method) ~[na:1.8.0_111] at java.io.FileInputStream.open(FileInputStream.java:195) ~[na:1.8.0_111] at java.io.FileInputStream.<init>(FileInputStream.java:130) ~[na:1.8.0_111] at org.dynatrace.service.ProxyController.proxyUrlWithProxyController.java:190 ~[classes/:na] at sun.reflect.GeneratedMethodAccessor71.invoke(Unknown Source) ~[na:na] at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) ~[na:1.8.0_111] at java.lang.reflect.Method.invoke(Method.java:498) ~[na:1.8.0_111] at org.springframework.web.method.support.InvocableHandlerMethod.doInvoke(InvocableHandlerMethod.java:190) ~[spring-web-5.2.2.RELEASE.jar/5.2.2.RELEASE] at org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerMethod.java:138) ~[spring-web-5.2.2.RELEASE.jar/5.2.2.RELEASE] at org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter.invokeHandlerMethod(AnnotationMethodHandlerAdapter.java:206) ~[spring-webmvc-5.2.2.RELEASE.jar/5.2.2.RELEASE] at org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter.invokeHandlerMethod(AnnotationMethodHandlerAdapter.java:180) ~[spring-webmvc-5.2.2.RELEASE.jar/5.2.2.RELEASE] at
```