# Zabezpieczenie systemów i usług sieciowych
# Laboratorium 5

**Wprowadzenie**

Celem ćwiczenia jest zapoznanie z podstawowymi metodami zarządzania dużą grupą serwerów. Jako przykład użyty zostanie serwer salt. Jednak zachęcam do zapoznania się z innymi podobnymi narzędziami (Puppet, Chef, Ansible) i wybrania tego które najlepiej spełnia nasze potrzeby. Przed przystąpieniem do zajęć należy zainstalować pakiet **p7zip**.

## Zadanie 1 (*)

Celem zadania jest uruchomienie jednego serwera który pozwoli na kontrolowanie podłączonych do niego maszyn. Aby maksymalnie uprościć przygotowanie środowiska zarówno serwer jak i podłączone maszyny uruchomimy jako kontenery docker. Aby wszystkie nasze "maszyny" mogły komunikować się bez problemów utworzymy wewnętrzną sieć pomiędzy kontenerami:

```
docker network create zsius
```

Teraz importujemy odpowiedni obraz dockera:

```
curl https://zsius-pliki.justdoit.tech/docker_salt.tar.xz | 7zr x -txz -si -so | docker load
```

Obraz jest dodatkowo skompresowany dlatego rozpakowujemy go przy pomocy programu 7zip. Następnie uruchamiamy serwer sterujący:

```
docker run -d --name=salt --net=zsius salt salt-master
```

Po czym uruchamiamy kilka przykładowych "maszyn" którymi będziemy sterować:

```
docker run -d --name=minion-1 --net=zsius salt salt-minion
docker run -d --name=minion-2 --net=zsius salt salt-minion
docker run -d --name=minion-3 --net=zsius salt salt-minion
```

Ze względu na ograniczony czas zajęć wszystkie pliki konfiguracyjne będziemy tworzyć ręcznie na serwerze sterującym. W rzeczywistym środowisku zmiany powinny być wersjonowane i umieszczane automatycznie na serwerze w reakcji na zmianę kodu w repozytorium. Aby dostać się do naszego serwera sterującego używamy komendy docker exec:

```
docker exec -it salt /bin/bash
```

Salt master komunikuje się z minionami (serwerami nad którymi ma pełną kontrolę) poprzez dedykowane szyfrowane połączenie. Aby zostało poprawnie nawiązane konieczna jest akceptacja kluczy publicznych wszystkich maszyn. W tym celu wydajemy komendę:

```
salt-key -A
```

Czasem musimy odczekać chwilę zanim wszystkie maszyny zgłoszą się do kontrolera. Aby sprawdzić czy wszystko działa poprawnie uruchamiamy:

```
salt '*' test.ping
```

Jako wynik powinniśmy otrzymać:

*7d7e24e4b9b3:*

*True*
*7d47edab64f6:*
*True*
*2ab6cf3dbed9:*
*True*

Możemy też wykonać dowolną komendę na wszystkich serwerach, na przykład:

```
salt '*' cmd.run 'ls -la'
```

'*' oznacza tutaj wszystkie serwery, w rzeczywistości mogłaby to być nazwa pojedynczego serwera lub selektor grupy serwerów np 'prod-web-*' lub 'test-db-*'. Daje to możliwość kontrolowania wielu maszyn jednocześnie z jednego punktu bez konieczności logowania się na poszczególne serwery. Jeśli jakiś "minion" zwraca błąd odczekujemy kilka minut i wydajemy komendę salt-key -A ponownie, czasem pomaga również restart "miniona" (docker restart minion-1)

## Zadanie 2

Celem zadania jest zapisanie pożądanej konfiguracji serwera w postaci pliku. Pozwoli to na traktowanie konfiguracji naszych serwerów podobnie jak kodu naszych aplikacji. Możliwe będzie audytowanie, wersjonowanie zmian oraz szybka instalacja nowych maszyn. W pierwszej kolejności dodajmy konto użytkownika na wszystkich naszych maszynach. Skorzystamy w tym celu z przygotowanej wcześniej konfiguracji znajdującej się w katalogu /srv. /srv/salt zawiera instrukcję w jaki sposób skonfigurować nasze serwery. Na przykład w pliku: /srv/salt/user/init.sls znajdziemy instrukcję jakiej użyje salt do założenia nowego konta. Natomiast katalog /srv/pillar zawiera konkretne dane potrzebne do wykonania tych instrukcji. Dodajmy teraz nowego użytkownika analogicznie do istniejącego usera. Aby tego dokonać edytujemy plik ('apt-get update && apt-get install vim' aby zainstalować edytor tekstu) /srv/pillar/users.sls. Po wpisaniu w nim danych nowego użytkownika wysyłamy nową konfigurację do naszych serwerów:

```
salt '*' state.highstate
```

Po chwili nasze serwery zaczną raportować postępy w wykonaniu zadania. Aby potwierdzić, że wszystko się udało wpisujemy:

```
salt '*' cmd.run 'getent passwd loginnowegousera'
```

Powinniśmy otrzymać taką samą informację zwrotną (zawierającą dane dodanego użytkownika) z wszystkich 3 serwerów.

## Zadanie 3

Celem zadania jest dodanie kolejnego serwera do puli zarządzanych maszyn. W tym celu otwieramy nowe okno putty i logujemy się do naszego serwera ćwiczeniowego. Następnie wydajemy komendę:

```
docker run -d --name=minion-4 --net=zsius salt salt-minion
```

Po czym wracamy do okna w którym mamy nasz serwer zarządzający i wpisujemy:

```
salt-key
```

Tym razem w odpowiedzi powinniśmy otrzymać coś podobnego do:
*Accepted Keys:*
*0d18cce56871*
*1343b7e09c16*
*19799c9491fd*
*Denied Keys:*
*Unaccepted Keys:*
*52f6789509ba*
*Rejected Keys:*

Akceptujemy klucz nowego serwera komendą:

```
salt-key -a 52f6789509ba
```

Oczywiście zmieniając nazwę serwera na właściwą. Powtarzając komendę z poprzedniego ćwiczenia:

```
salt ‘*’ cmd.run ‘getent passwd loginnowegousera’
```

Zobaczymy, że nowy serwer nie ma jeszcze naszego użytkownika. Zmuszamy go, więc do przeładowania konfiguracji poprzez:

```
salt '52f6789509ba' state.highstate
```

Teraz getent powinien już zwracać identyczny rezultat dla wszystkich 4 serwerów.

## Podsumowanie

Obecnie dostępne narzędzia administracyjne są wyjątkowo zaawansowane oraz łatwe w instalacji i utrzymaniu. Dzisiejszy przykład dotyczył jedynie dodania nowego konta użytkownika, jednak bez problemu moglibyśmy zmienić plik konfiguracyjny dowolnej usługi lub też zainstalować dowolny pakiet oprogramowania równocześnie na wszystkich serwerach. Lista gotowych modułów salt jest dość pokaźna: https://docs.saltstack.com/en/latest/ref/modules/all/index.html Nawet gdyby nie było jeszcze modułu rozwiązującego nasz problem napisanie własnego nie stanowi większego wyzwania. Dlatego też szczerze polecam wykorzystywanie tego typu narzędzi w codziennej pracy. Pomagają one znaleźć czas na rozwiązanie problemów bardziej istotnych niż ręczna mozolna konfiguracja wszystkich serwerów.