WCS in Yandex.Cloud

- Server deployment
  - Create and launch VM instance
  - Firewall rules setup
  - WCS installation and configuration
  - WCS starting and testing
- Default admin credentials

Since build 5.2.759, WCS can be deployed in Yandex.Cloud as separate media server or low latency streaming CDN node.

The following is necessary to be prepared before deploy:

- active Yandex.Cloud account, a cloud and virtual private network in this account
- a WCS license to activate on server/servers
- optionally, domain names to bind to bind to servers static IPs and corresponding SSL certificates

Server deployment

Create and launch VM instance

1. In Yandex,Cloud console go to "Compute Cloud - Virtual machines" section and click "Create VM" to begin VM instance creation

2. Enter server name, description and choose datacenter region

3. In "Computing resources" section choose processor type and count, memory size. Set the parameter "Guaranteed vCPU performance" to "100%"

4. In "Image/boot disk selection" section choose Centos, version 7 (other operating systems listed here are allowed too)

5. In "Disks" section choose disk type and size

6. In "Network settings" section choose available subnet, Set manual IP addresses if necessary

7. In "Access" set user name and public SSH access key

then click "Create VM"

8. VM instance created will appear in VMs list

9. Click VM instance string, copy pulic IP address from "Network" section to access the server

10. Connect to the instance by SSH

Firewall rules setup

Yandex.Cloud does not support security groups now (the feature is in Preview state), therefore ii is necessary to set up firewall on the instance itself:

iptables_setup.sh Expand source

```
#!/bin/bash
#
export IPT="iptables"

# External interface
export WAN=eth0

# Clean iptables
$IPT -F
$IPT -F -t nat
$IPT -F -t mangle
$IPT -X
$IPT -t nat -X
$IPT -t mangle -X

# Set default policies
$IPT -P INPUT ACCEPT
$IPT -P OUTPUT ACCEPT
$IPT -P FORWARD ACCEPT

# Allow local traffic
$IPT -A INPUT -i lo -s 127.0.0.0/8 -d 127.0.0.0/8 -j ACCEPT
$IPT -A OUTPUT -o lo -s 127.0.0.0/8 -d 127.0.0.0/8 -j ACCEPT

# Allow outgoing connections
$IPT -A OUTPUT -o $WAN -j ACCEPT

# Allow already established connections
$IPT -A INPUT -p all -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A OUTPUT -p all -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A FORWARD -p all -m state --state ESTABLISHED,RELATED -j ACCEPT

# Enable packet fragmentation
#$IPT -I FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu

# Drop invalid packets
$IPT -A INPUT -m state --state INVALID -j DROP
$IPT -A FORWARD -m state --state INVALID -j DROP
$IPT -A INPUT -p tcp ! --syn -m state --state NEW -j DROP
```

```
$IPT -A OUTPUT -p tcp ! --syn -m state --state NEW -j DROP

# Allow pings
$IPT -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type destination-unreachable -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type time-exceeded -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type echo-request -j ACCEPT

# Allow SSH
$IPT -A INPUT -p tcp --dport 22 -j ACCEPT
# Allow DNS
#$IPT -A INPUT -i $WAN -p udp --dport 53 -j ACCEPT
# Allow NTP
#$IPT -A INPUT -i $WAN -p udp --dport 123 -j ACCEPT

# Allow WCS ports
$IPT -A INPUT -p tcp --dport 80 -j ACCEPT
$IPT -A INPUT -p tcp --dport 443 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8888 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8443 -j ACCEPT
$IPT -A INPUT -p tcp --dport 1935 -j ACCEPT
$IPT -A INPUT -p udp --dport 1935 -j ACCEPT
$IPT -A INPUT -p tcp --dport 554 -j ACCEPT
$IPT -A INPUT -p tcp --dport 3478 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8080 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8081 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8084 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8082 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8085 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8445 -j ACCEPT
$IPT -A INPUT -p tcp --dport 8444 -j ACCEPT
$IPT -A INPUT -p tcp --dport 10000:50000 -j ACCEPT
$IPT -A INPUT -p udp --dport 10000:50000 -j ACCEPT
$IPT -A INPUT -p tcp --dport 50999 -j ACCEPT

$IPT -A INPUT -j DROP
$IPT -A FORWARD -j DROP

# Save the rules to file
/sbin/iptables-save  > /etc/sysconfig/iptables
```

WCS installation and configuration

1. Install JDK. It is recommended to use JDK 12 or 14 if high load is planning

```
#!/bin/bash
sudo rm -rf jdk*
curl -s
https://download.java.net/java/GA/jdk12.0.2/e482c34c86bd4bf8b56c0b35558996b9/10/
12.0.2_linux-x64_bin.tar.gz | tar -zx
[ ! -d jdk-12.0.2/bin ] && exit 1
sudo mkdir -p /usr/java
[ -d /usr/java/jdk-12.0.2 ] && sudo rm -rf /usr/java/jdk-12.0.2
sudo mv -f jdk-12.0.2 /usr/java
[ ! -d /usr/java/jdk-12.0.2/bin ] && exit 1
```

```
sudo rm -f /usr/java/default
sudo ln -sf /usr/java/jdk-12.0.2 /usr/java/default
sudo update-alternatives --install "/usr/bin/java" "java" "/usr/java/jdk-
12.0.2/bin/java" 1
sudo update-alternatives --install "/usr/bin/jstack" "jstack" "/usr/java/jdk-
12.0.2/bin/jstack" 1
sudo update-alternatives --install "/usr/bin/jcmd" "jcmd" "/usr/java/jdk-
12.0.2/bin/jcmd" 1
sudo update-alternatives --install "/usr/bin/jmap" "jmap" "/usr/java/jdk-
12.0.2/bin/jmap" 1
sudo update-alternatives --set "java" "/usr/java/jdk-12.0.2/bin/java"
sudo update-alternatives --set "jstack" "/usr/java/jdk-12.0.2/bin/jstack"
sudo update-alternatives --set "jcmd" "/usr/java/jdk-12.0.2/bin/jcmd"
sudo update-alternatives --set "jmap" "/usr/java/jdk-12.0.2/bin/jmap"
```

2. Install accessory tools and libraries

```
sudo yum install -y tcpdump mc iperf3 fontconfig
```

3. Disable SELinux

```
sudo setenforce 0
```

4. Install WCS

```
curl -OL
https://flashphoner.com/downloads/builds/WCS/5.2/FlashphonerWebCallServer-
5.2.xxx.tar.gz
tar -xzf FlashphonerWebCallServer-5.2.xxx.tar.gz
cd FlashphonerWebCallServer-5.2.xxx
sudo ./install.sh
```

Where xxx is WCS actual build number

5. Activate your license

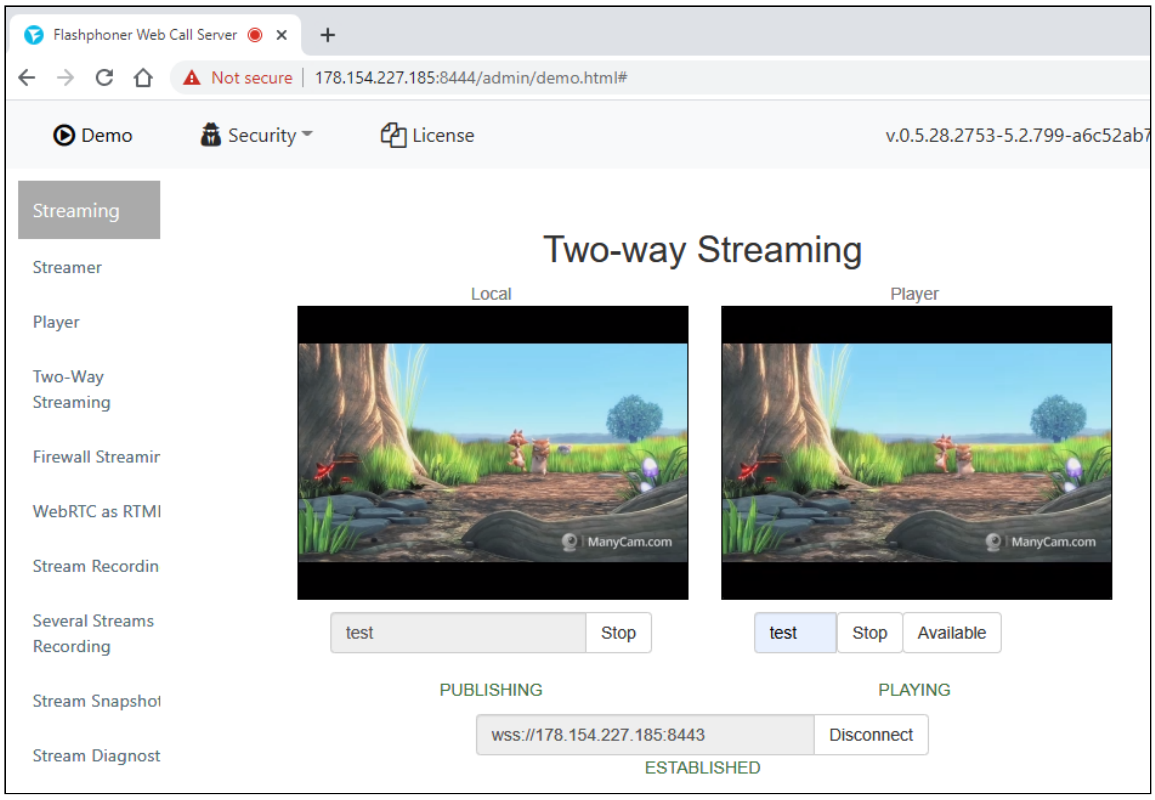
```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./activation.sh
```

WCS starting and testing

1. Start WCS

```
sudo systemctl start webcallserver
```

2. Enter to WCS web interface, open Two Way Streaming example, publish and play test stream



Default admin credentials

The running instance data can be received in Yandex.Cloud by two ways: using Google Cloud API endpoints or AWS EC2 API endpoints. Therefore, WCS detects cloud environment as Amazon-like since build 5.2.921.

In its turn, Amazon requires to use an unique admin password for every instance, and WCS sets admin password in Amazon-like cloud environment by unique instanceId available via API or in EC2 console.

Therefore, since build 5.2.921 WCS sets admin password to instanceId on first launch in Yandex.Cloud. However, this parameter may not be displayed in Yandex.Clouid console. To get instanceId, connect to the instance via SSH and use the following command

```
curl http://169.254.169.254/latest/meta-data/instance-id
```

Attachments:

- yandex_create_vm.png (image/png)
- yandex_create_vm_choose_name.png (image/png)
- yandex_create_vm_choose_cpu.png (image/png)
- yandex_create_vm_choose_os.png (image/png)
- yandex_create_vm_choose_hdd.png (image/png)

- yandex_create_vm_choose_network.png (image/png)
- yandex_create_vm_choose_user_and_create.png (image/png)
- yandex_vm_list.png (image/png)
- yandex_vm_ip_address.png (image/png)
- yandex_vm_ssh.png (image/png)
- yandex_vm_test.png (image/png)