WCS in Yandex.Cloud

Server deployment from Yandex.Cloud Marketplace image

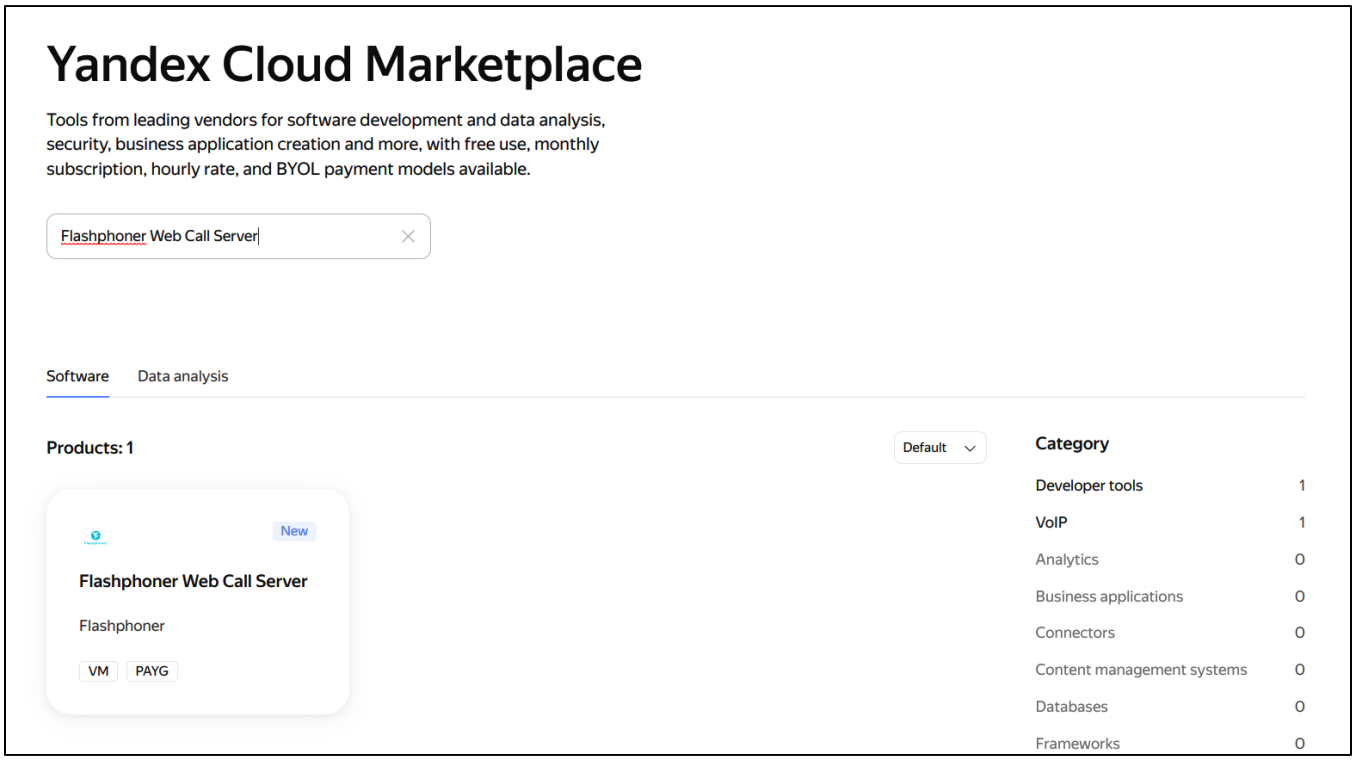
Flashphoner Web Call Server image with hourly billing is available in Yandex.Cloud Marketplace. This way is preferable for short term instances (weeks and months).

The following is necessary to be prepared before deploy:

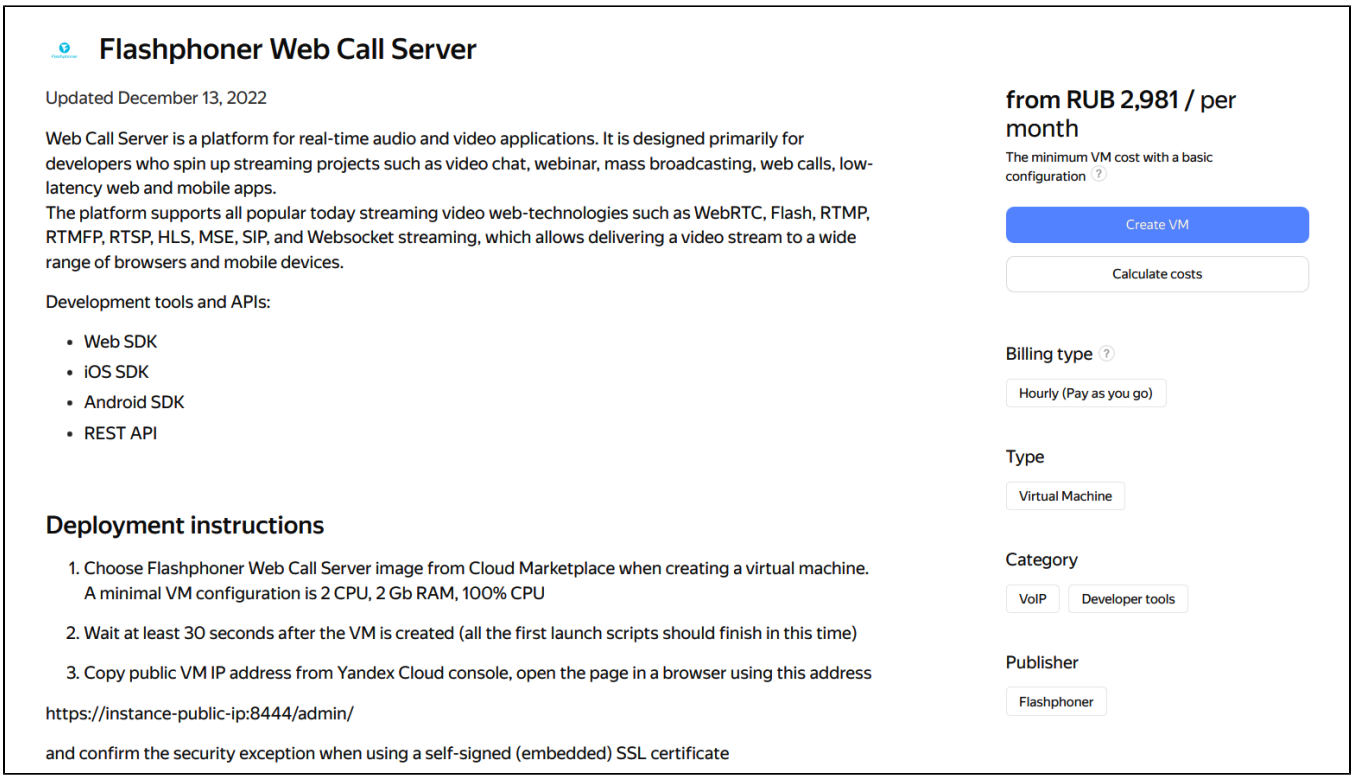
- active Yandex.Cloud account, a cloud and virtual private network in this account
- optionally, domain names to bind to bind to servers static IPs and corresponding SSL certificates

Create and launch VM instance

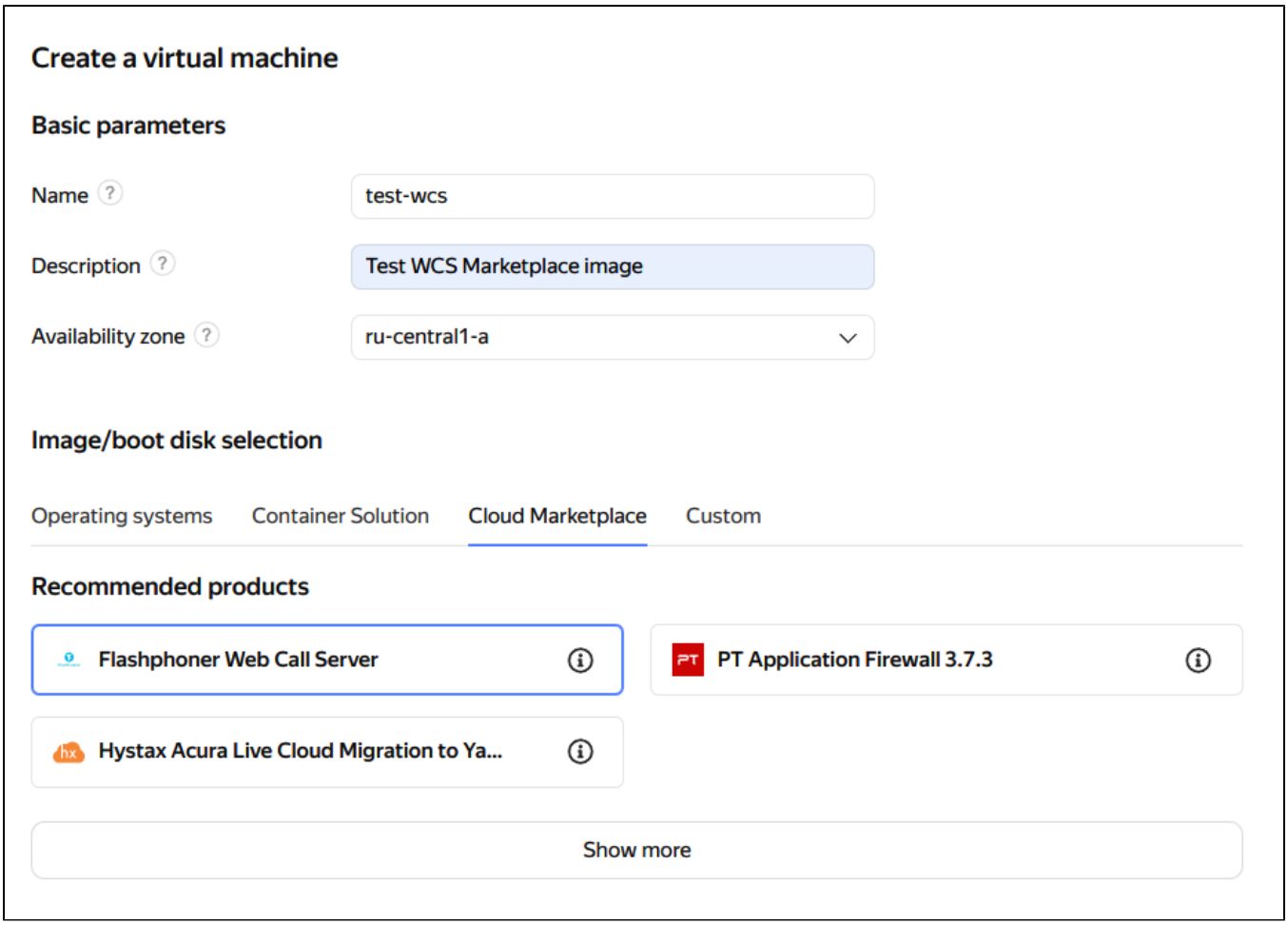
1. Find the product Flashphoner Web Call Server in Yandex.Cloud Marketplace or open the page in Yandex.Cloud Marketplace directly



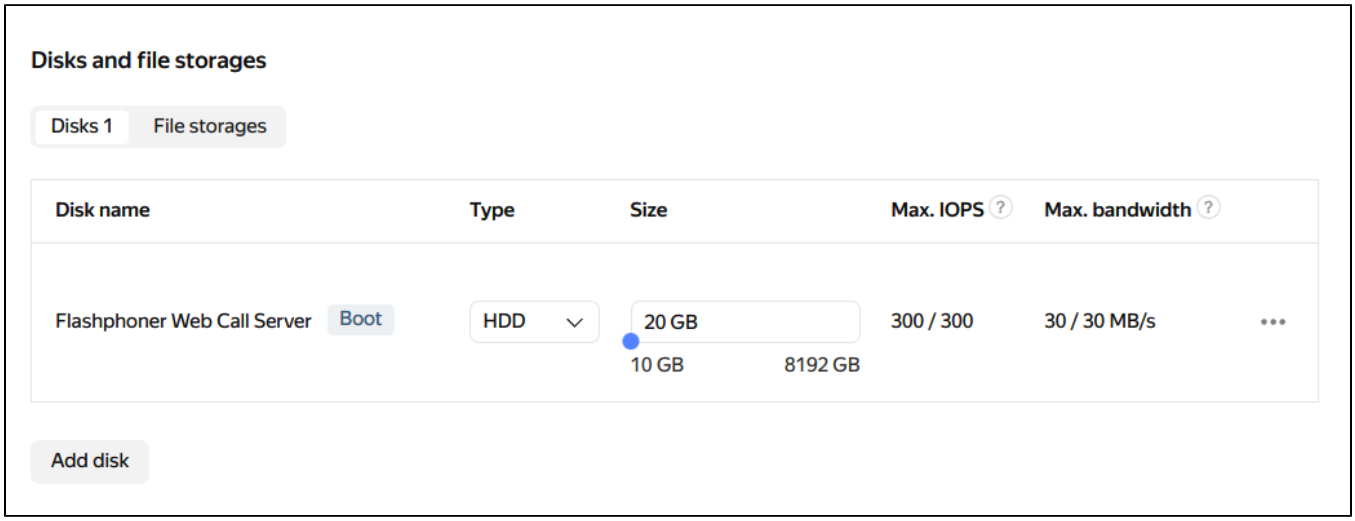
2. Click `Create VM` on the product page



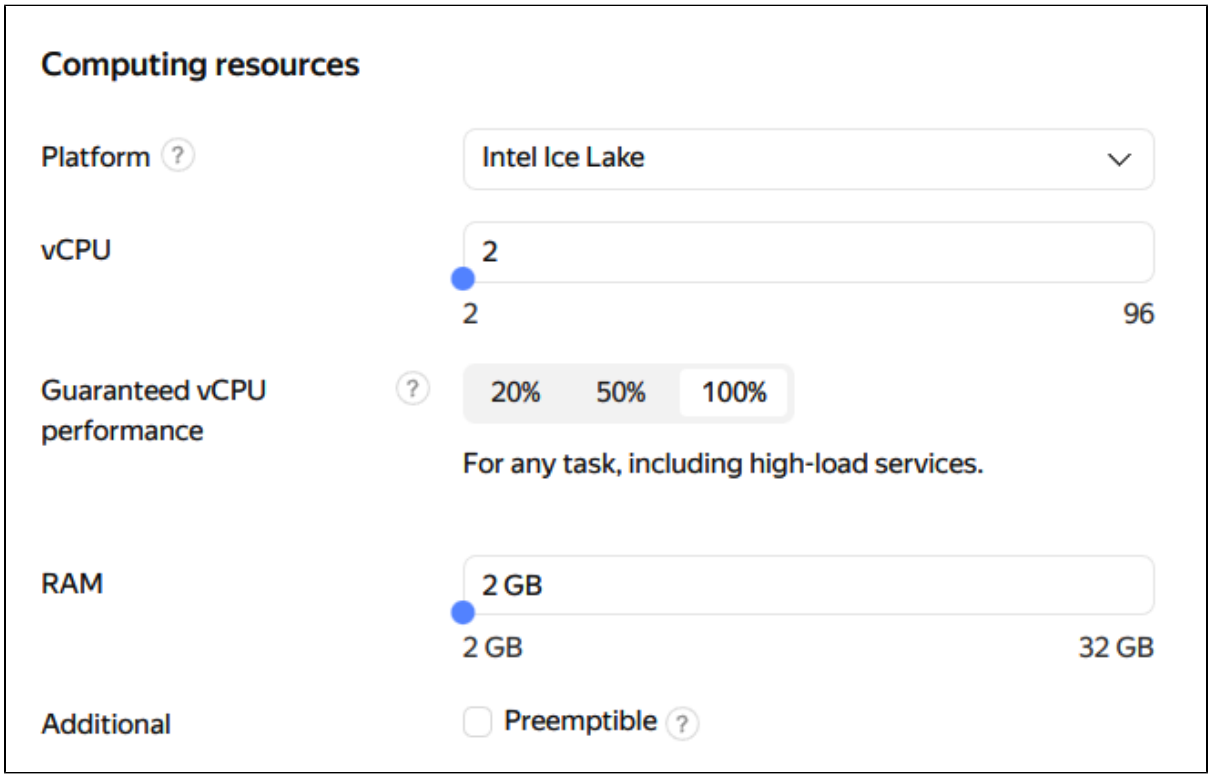
3. Enter server name, description and choose datacenter region



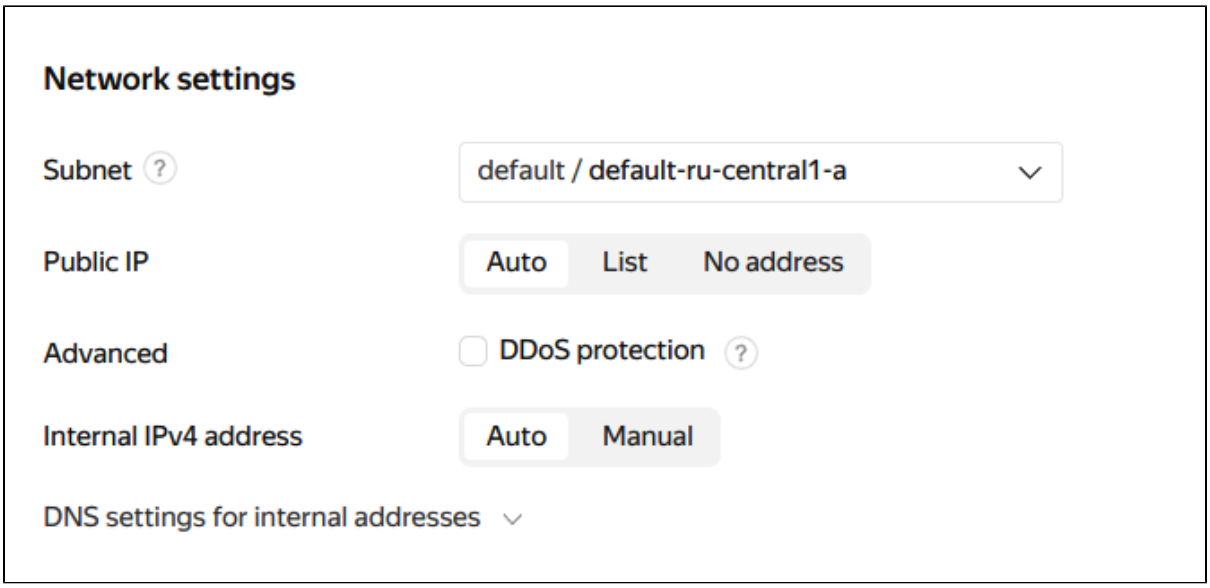
4. Choose storage type and size in `Disks` section



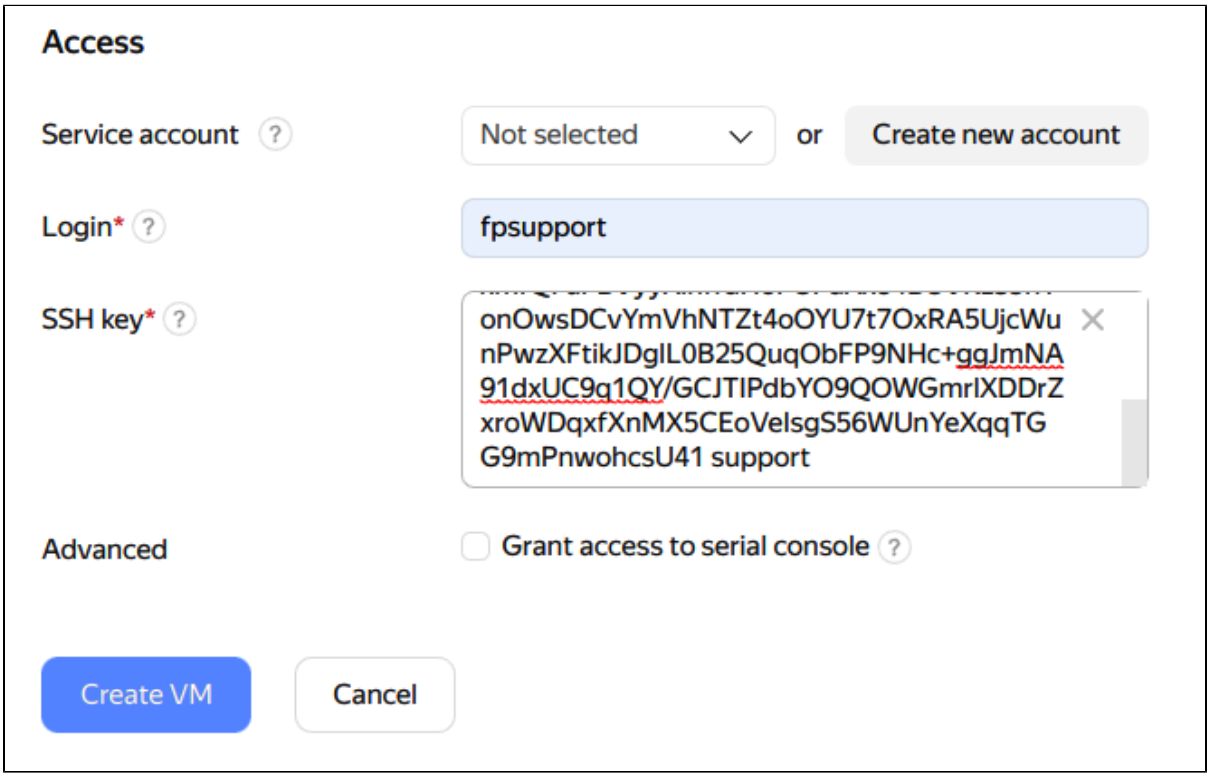
5. Choose CPU type and count, adjust RAM size in `Computing resources` section. A minimal required parameters are set by default. Note that `Guaranteed vCPU performance` parameter must be `100%`



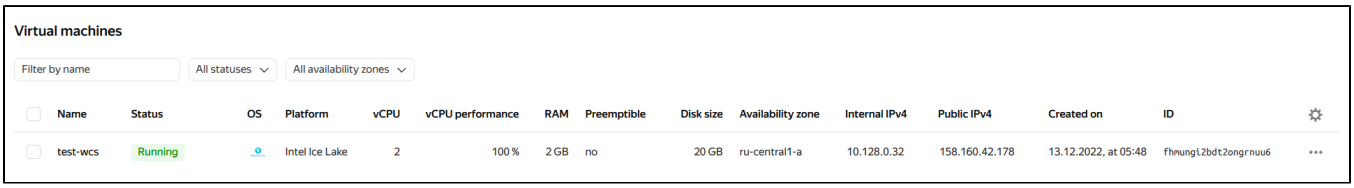
6. Choose available subnet, set manual IP addresses if necessary in `Network settings` section



7. Set user name and public SSH access key in `Access` section, then click `Create VM`

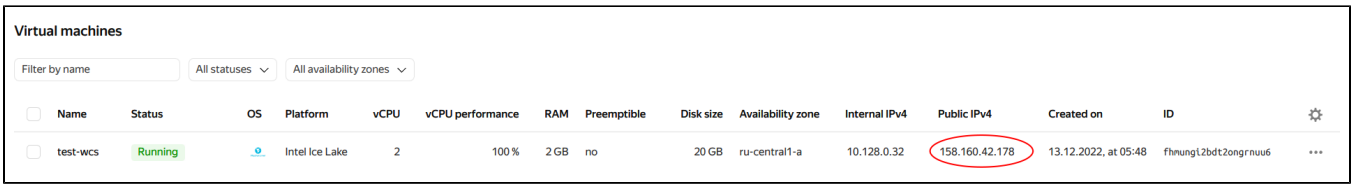


8. Wait for VM changes its state to `Running` (page refresh may be required)

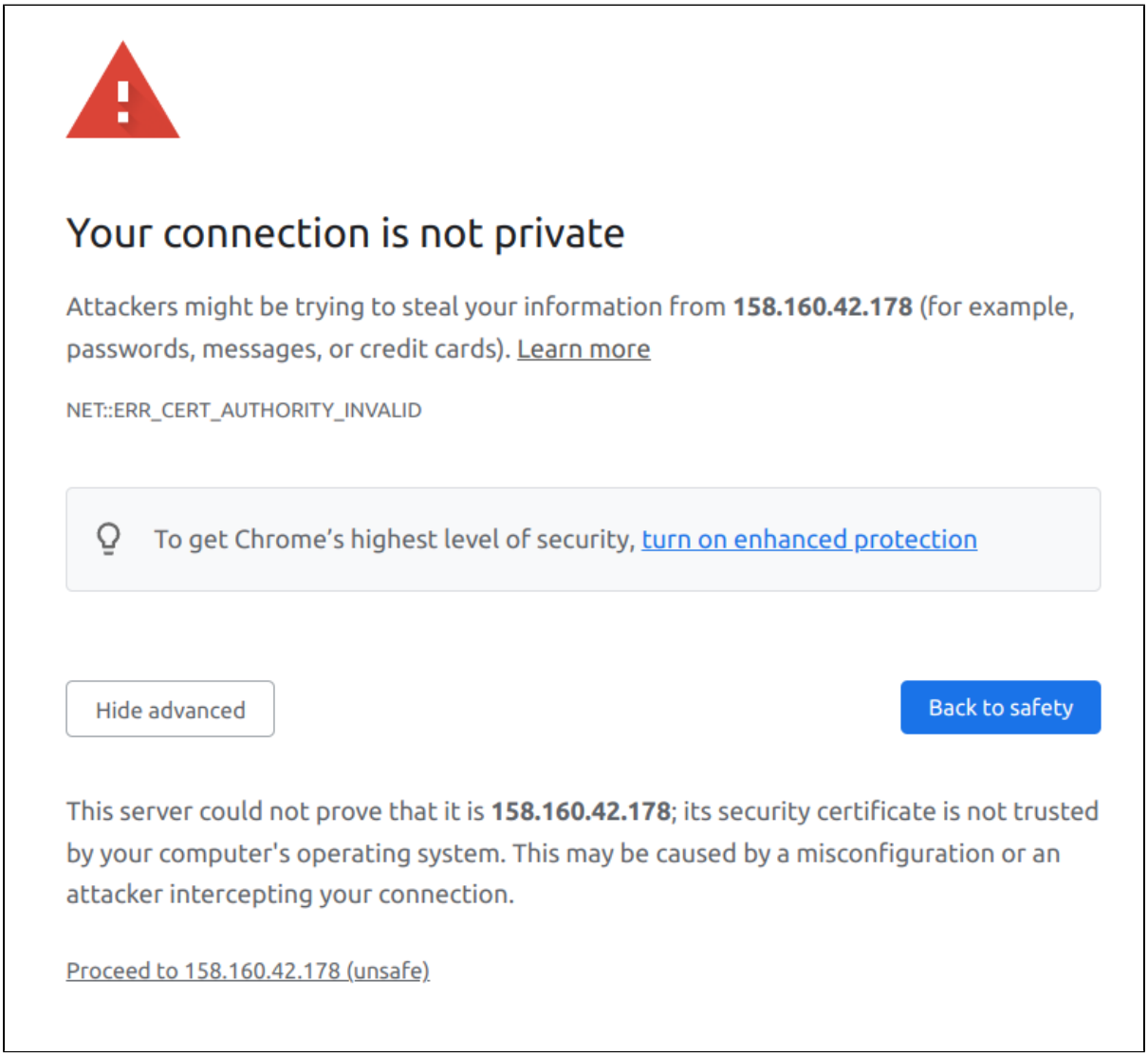


Testing WCS instance

1. Wait at least for 30 seconds after VM changes its state to `Running` for all the first launch scripts to work. Then copy a public IP address

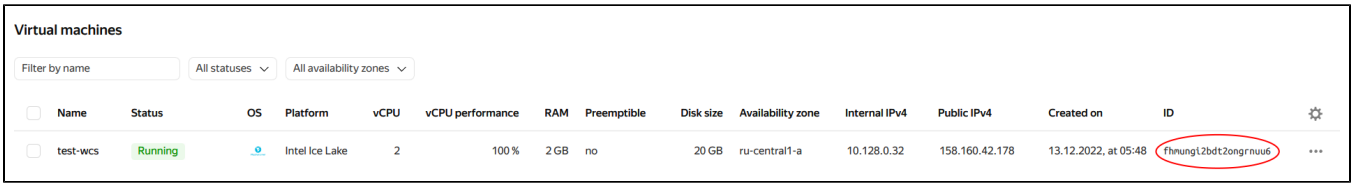


2. In a new browser tab, open the URL `https://public-ip:8444/admin/`, where `public-ip` - public IP address copied above. Accept the security exception (WCS uses self signed SSL certificates by default)

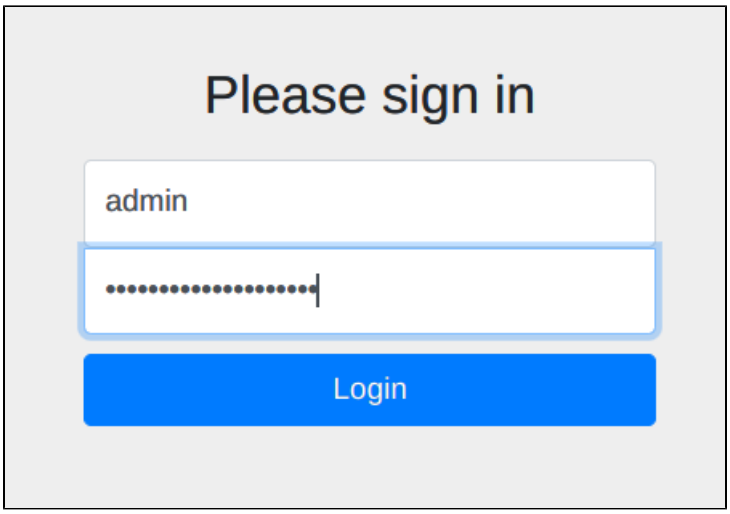


WCS web interface login page will open.

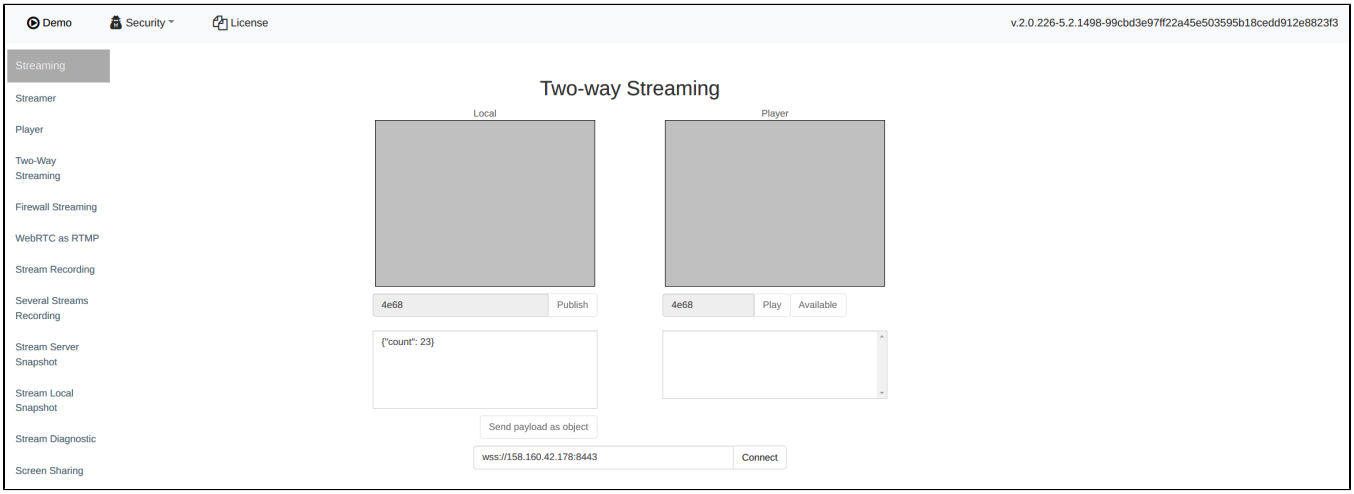
3. Copy `ID` from VM parameters



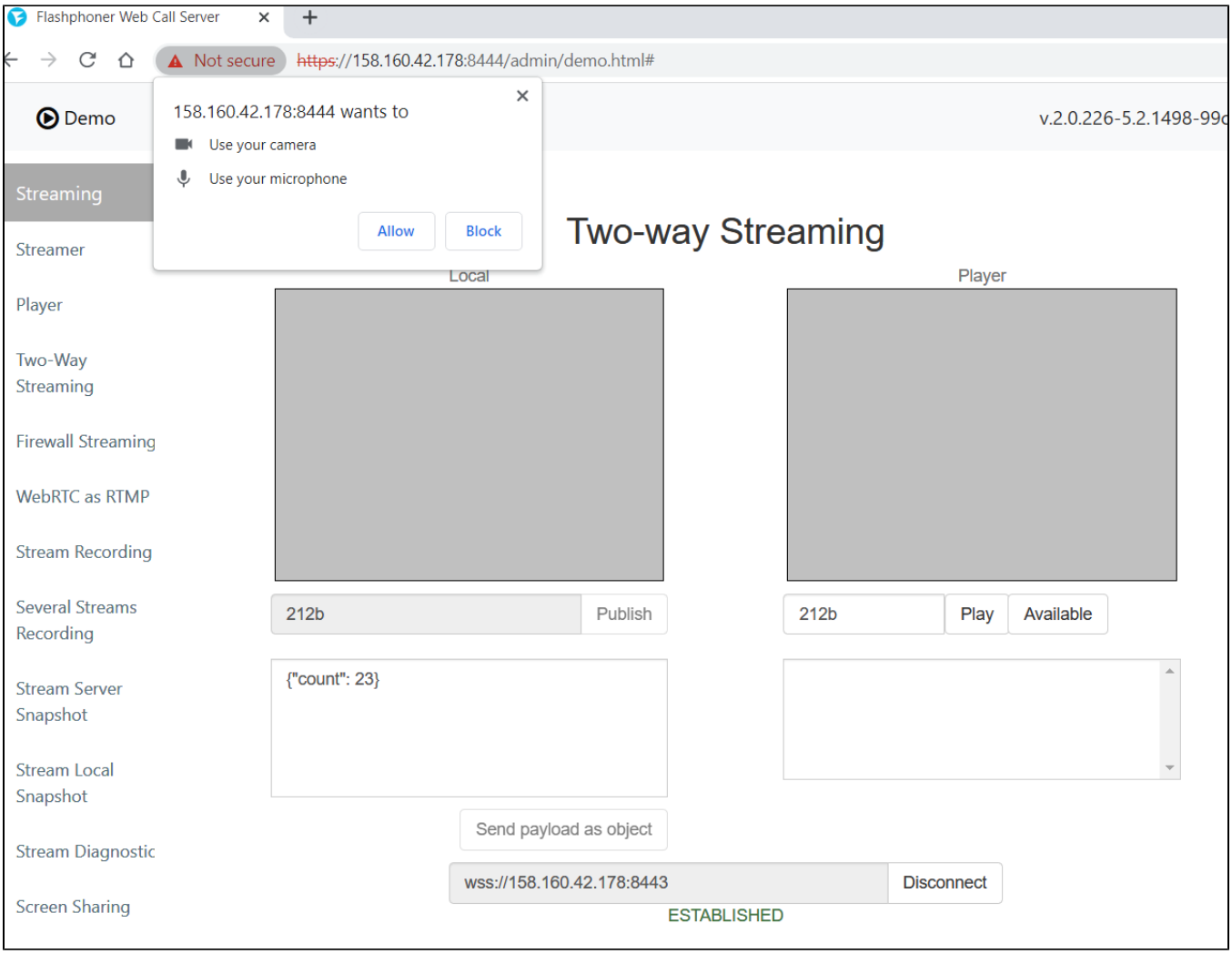
4. On WCS web interface login page enter `admin` user name, use `ID`copied above as password



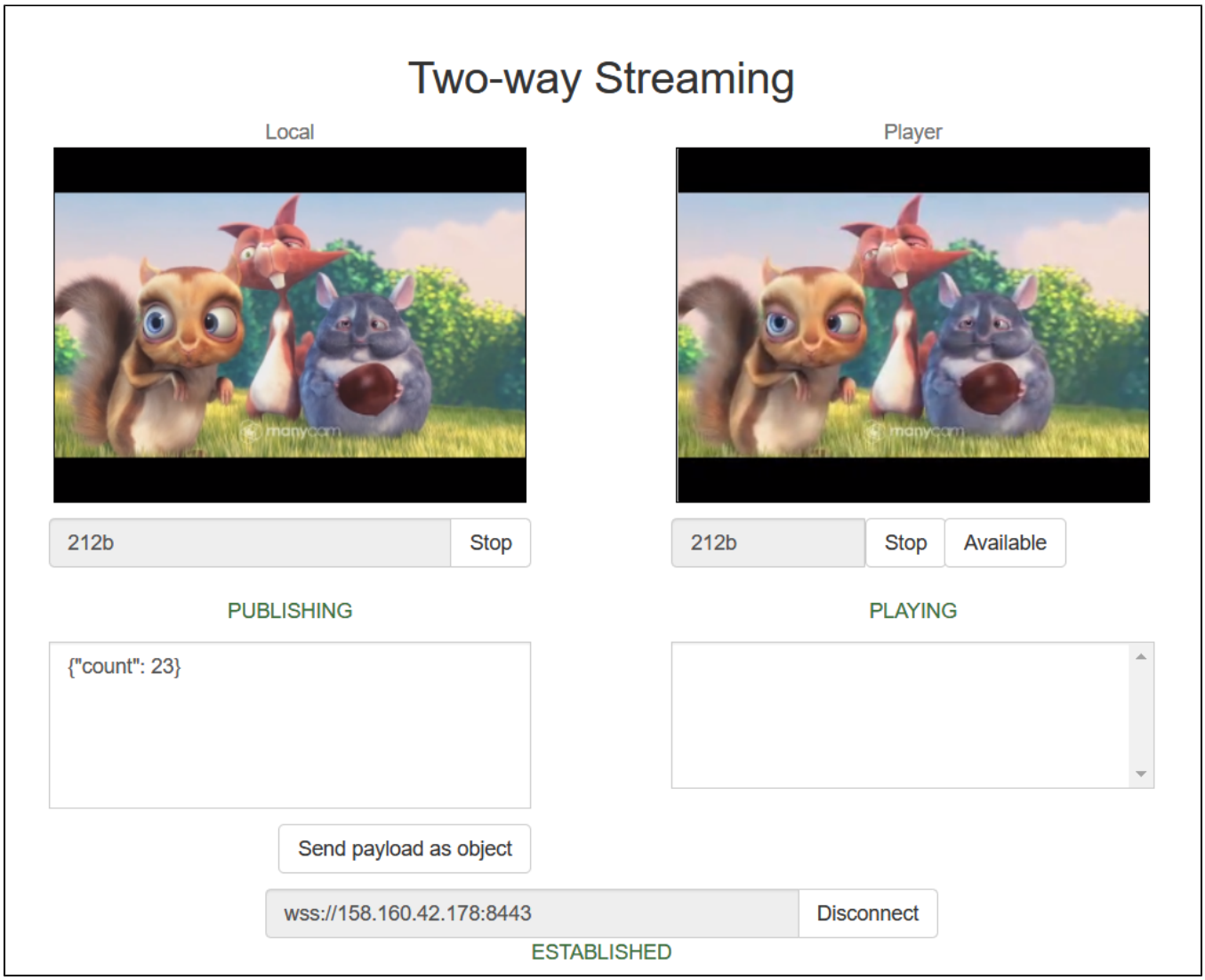
5. Choose Two Way Streaming example



6. Click `Connect`, then `Publish`. Allow camera and microphone access



7. Click `Play` when `PUBLISHING`, is displayed under the `Local` window



The browser sends media stream to the server and plays it from the server. The WCS instance is working.

Server deployment from the scratch using one of the standard Linux images

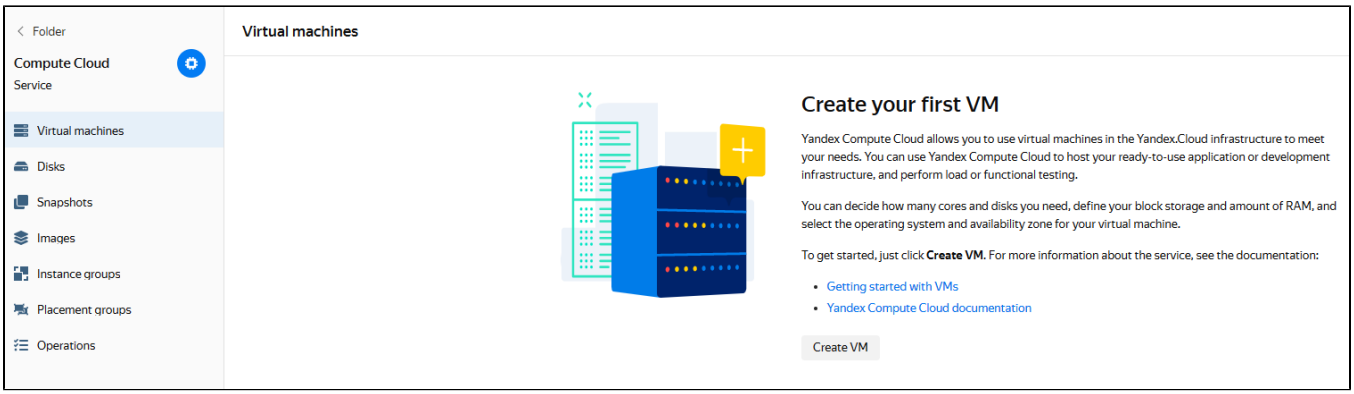
Since build5.2.759,WCS can be deployed in Yandex.Cloud using one of the standard Linux images as separate media server or low latency streaming CDN node. This way is preferable for long term server instances (from year and more).

The following is necessary to be prepared before deploy:

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

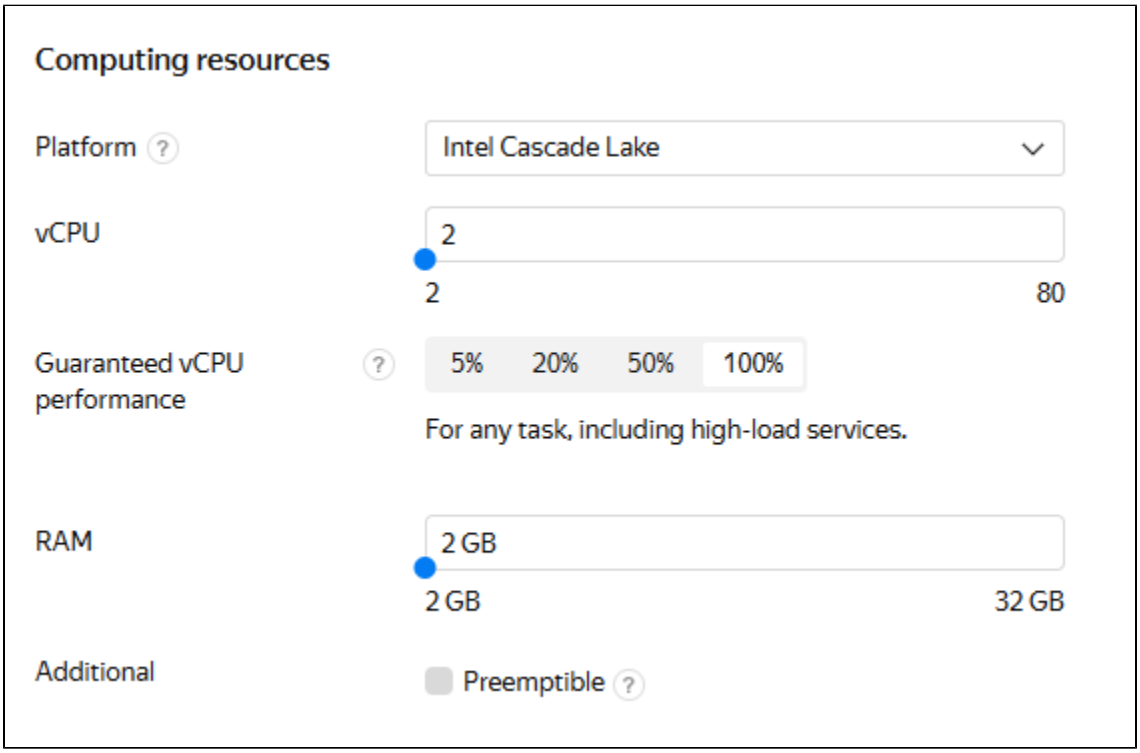
Create and launch VM instance

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

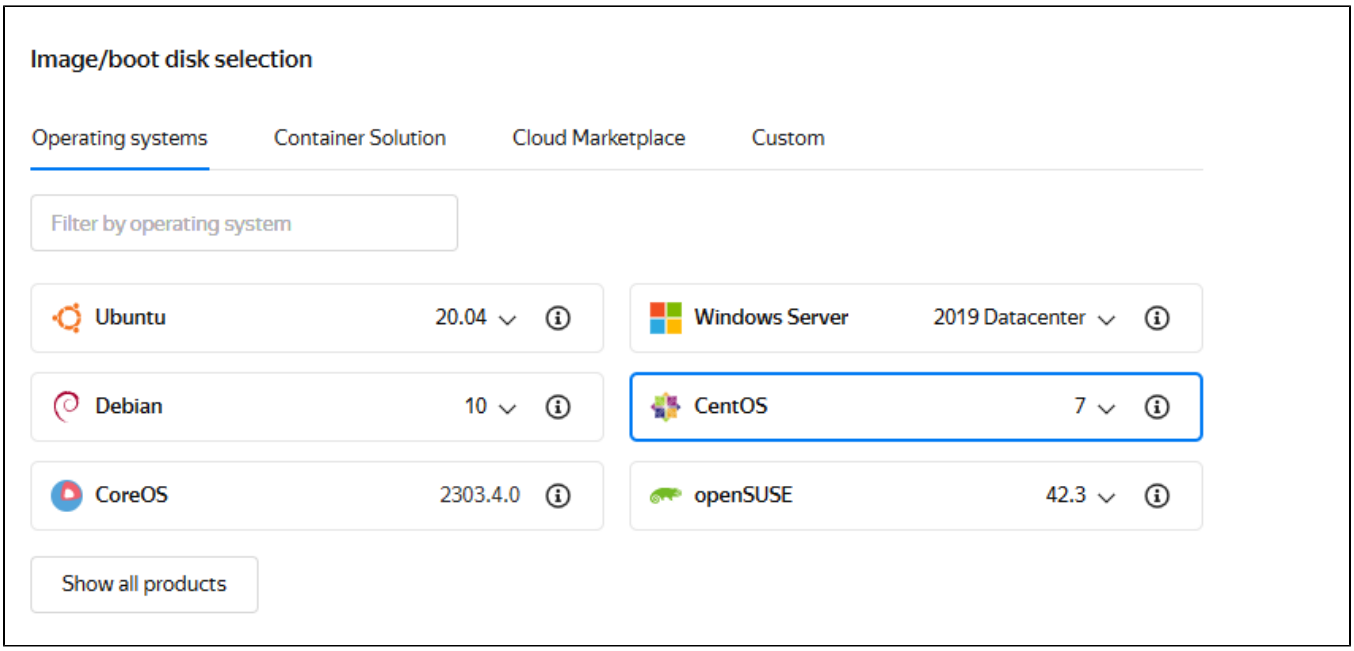


2. Enter server name, description and choose datacenter region

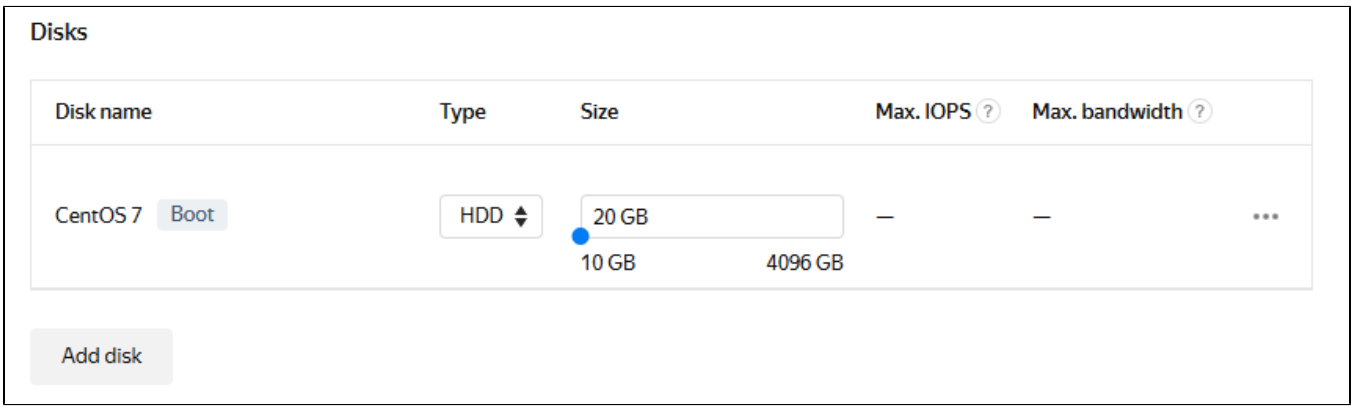
3. In "Computing resources" section choose processor type and count, memory size. Set the parameter "Guaranteed vCPU performance" to "100%"



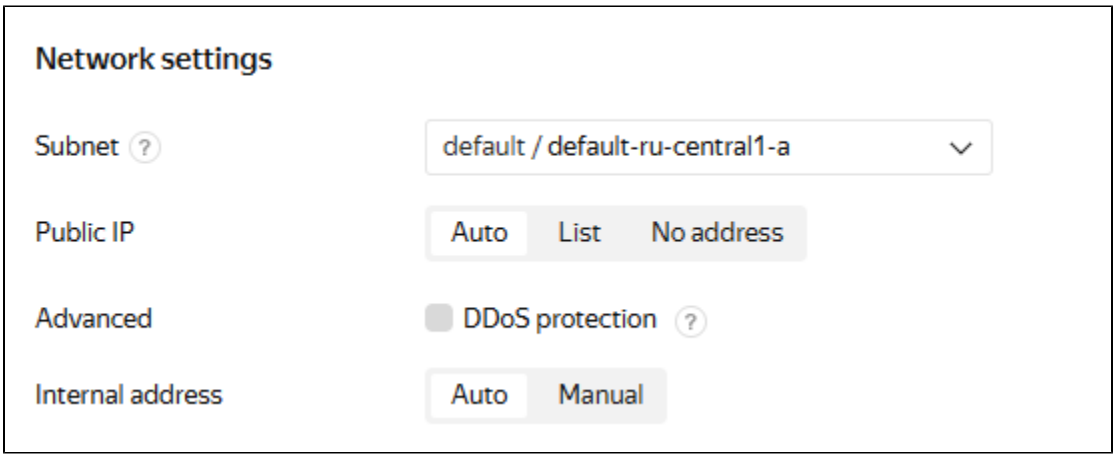
4. In "Image/boot disk selection" section choose Centos, version 7 (other operating systems listed here are allowed too)



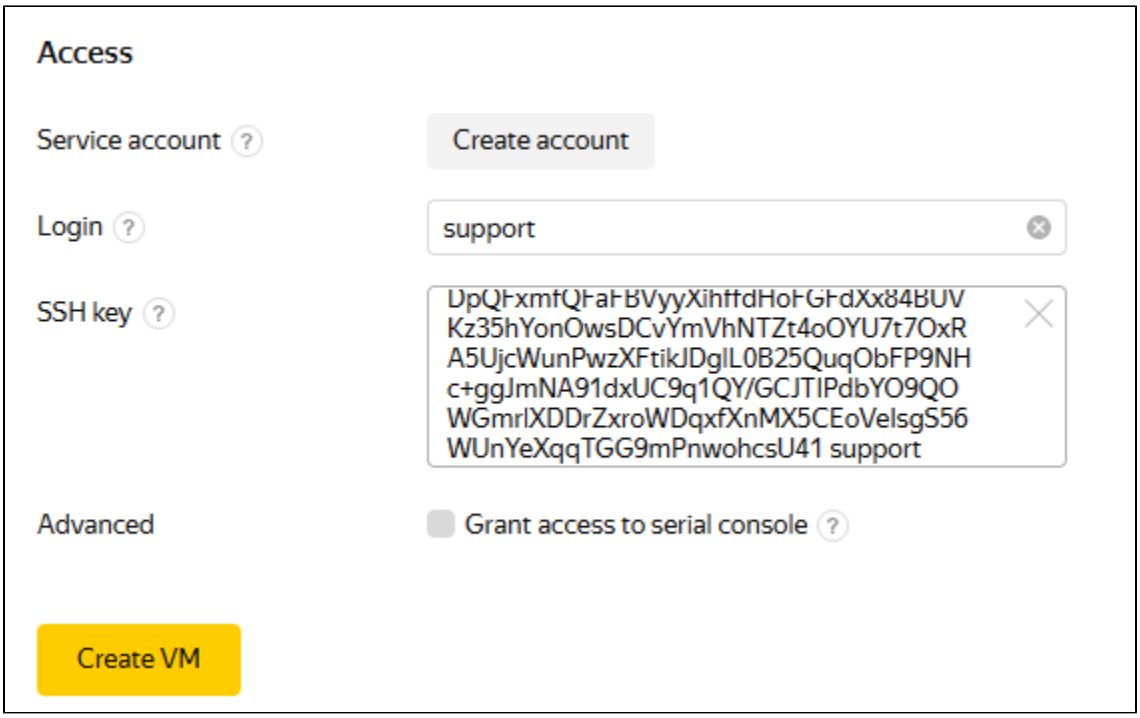
5. In "Disks" section choose disk type and size



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

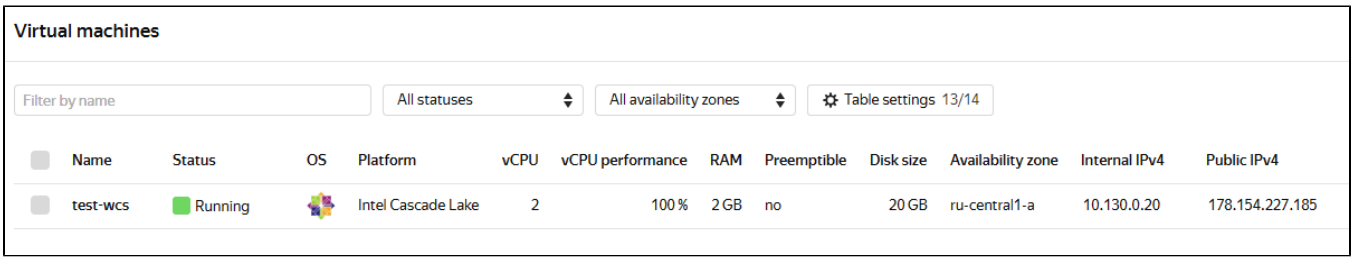


7. In "Access" set user name and public SSH access key

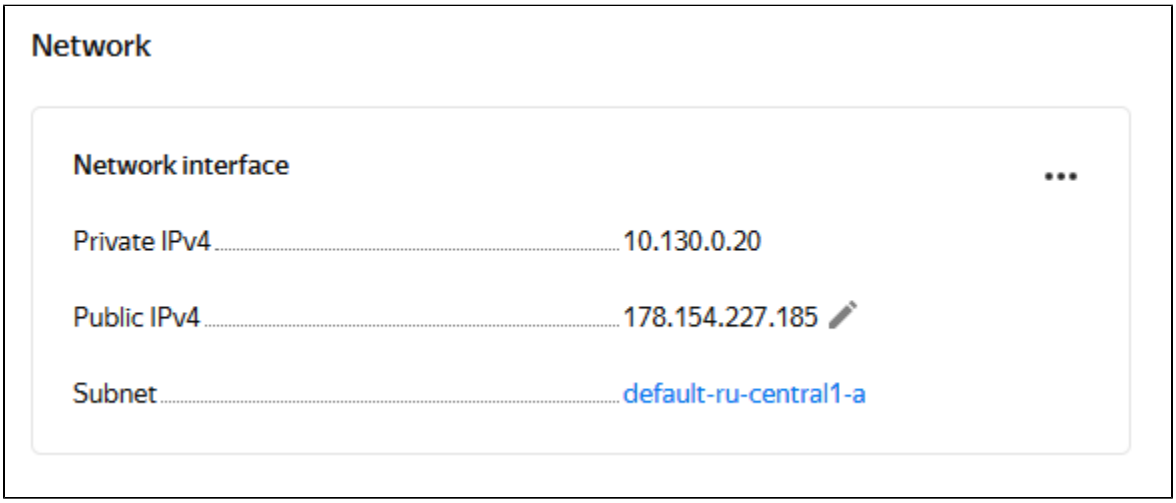


then click "Create VM"

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

```
$ ssh -i /g/.ssh/id_rsa_yandex support@178.154.227.185
The authenticity of host '178.154.227.185 (178.154.227.185)' can't be established.
ECDSA key fingerprint is SHA256:69SQ1JWPNe3+F7fHHx1K70gmN/hIohHce9NNsrWbVA0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '178.154.227.185' (ECDSA) to the list of known hosts.
[support@test-wcs ~]$ uname
Linux
[support@test-wcs ~]$ uname -a
Linux test-wcs.ru-central1.internal 3.10.0-1127.el7.x86_64 #1 SMP Tue Mar 31 23:36:51 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
[support@test-wcs ~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                2
On-line CPU(s) list:   0,1
Thread(s) per core:    2
Core(s) per socket:    1
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel Xeon Processor (Cascadelake)
Stepping:              6
CPU MHz:               2095.068
BogoMIPS:              4190.13
Hypervisor vendor:     KVM
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              4096K
L3 cache:              16384K
NUMA node0 CPU(s):     0,1
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht sys
x16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch in
dx smap clflushopt clwb avx512cd avx512bw avx512vl xsaveopt xsavec xgetbv1 arat pku ospke avx512_vnni md_clear spec_ctrl intel_
[support@test-wcs ~]$ free
              total        used        free      shared  buff/cache   available
Mem:        1881860       99072     1662108         556      120680     1647320
Swap:             0           0           0
[support@test-wcs ~]$
```

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

```
#!/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/GPL/openjdk-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

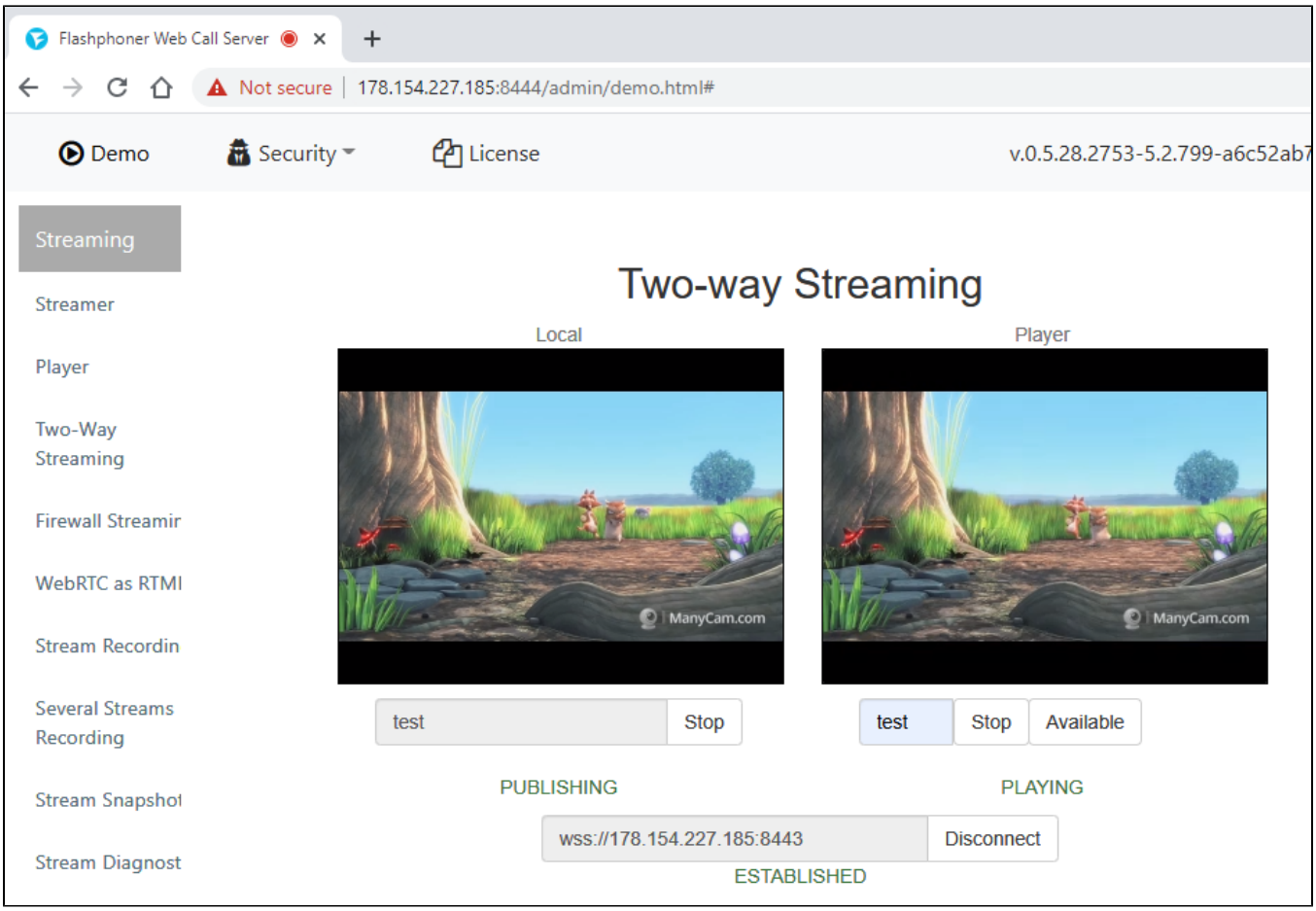
```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./activation.sh
```

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 build5.2.921.

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 build5.2.921WCS sets admin password toinstanceId 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
```