QoS

QoS (Quality of Service) – . QoS , .

QoS

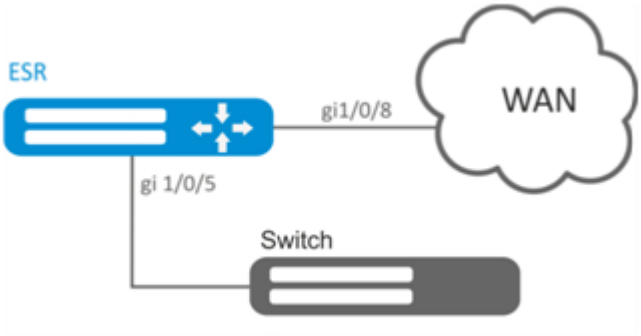
ESR ( ) (QoS)

1	QoS // . QoS, BasicQoS.	**esr(config-if-gi)# qos enable**	
2	802.1p DSCP ( ).	**esr(config)# qos trust <MODE>**	<MODE> – 802.1p DSCP, : • **dscp** – DSCP IP-. IP- . • **cos** – 802.1p 802.1q. . • **cos - dscp** – DSCP IP- 802.1p .
3	DSCP //, QOS ( ).	**esr(config)# qos map dscp-queue <DSCP> to <QUEUE>**	<DSCP> – IP- , [0..63]; <QUEUE> – , [1..8]. : • DSCP: (0-7), 1 • DSCP: (8-15), 2 • DSCP: (16-23), 3 • DSCP: (24-31), 4 • DSCP: (32-39), 5 • DSCP: (40-47), 6 • DSCP: (48-55), 7 • DSCP: (56-63), 8
4	802.1p . //, QOS ( ).	**esr(config)# qos map cos-queue <COS> to <QUEUE>**	<COS> – 802.1q , [0..7]; <QUEUE> – , [1..8]. : • CoS: (0), 1 • CoS: (1), 2 • CoS: (2), 3 • CoS: (3), 4 • CoS: (4), 5 • CoS: (5), 6 • CoS: (6), 7 • CoS: (7), 8
5	DSCP DSCP ( ). //, QoS.	**esr(config)# qos map dscp-queue <DSCP> to <DSCP>**	<DSCP> – IP- , [0..63].
6	DSCP DSCP-Mutation ( ).	**esr(config)# qos dscp mutation**	
7	, IP DSCP-.	**esr(config)# qos queue default <QUEUE>**	<QUEUE> – , [1..8].
8	. ( ).	**esr(config)# priority-queue out num-of-queues <VALUE>**	<VALUE> – , [0..8], : • **0** – WRR (WRR –); • **8** – «strictpriority» (strictpriority – ,). , 8-, . : 8

9	.	**esr(config)# qos wrr-queue <QUEUE> bandwidth <WEIGHT>**	<QUEUE> – , [1..8]; <WEIGHT> – , [1..255]. : 1
10	. BasicQoS- . QoS, ( ).	**esr(config-if-gi)# traffic-shape { <BANDWIDTH> [BURST] | queue <QUEUE><BANDWIDTH> [BURST] }**	<QUEUE> – , [1..8]; <BANDWIDTH> – /, [3000..10000000] TengigabitEthernet [64.. 1000000] ; <BURST> – , [4..16000]. 128 . :
11	( ).	**esr(config-if-gi)# rate-limit <BANDWIDTH> [BURST]**	<BANDWIDTH> – /, [3000..10000000] TengigabitEthernet [64.. 1000000] ; <BURST> – , [4..16000]. 128 . :

:

gigabitethernet 1/0/8: DSCP 22 , DSCP 14 , 60 / .



:

, , 1:

```
esr(config)# priority-queue out num-of-queues 1
```

DSCP 22 :

```
esr(config)# qos map dscp-queue 22 to 8
```

DSCP 14 :

```
esr(config)# qos map dscp-queue 14 to 7
```

QoS LAN:

```
esr(config)# interface gigabitethernet 1/0/5
esr(config-if-gi)# qos enable
esr(config-if-gi)# exit
```

QoS WAN :

```
esr(config)# interface gigabitethernet 1/0/8
esr(config-if-gi)# qos enable
```

60 / :

```
esr(config-if)# traffic-shape queue 7 60000
esr(config-if)# exit
```

QoS :

```
esr# show qos statistics gigabitethernet 1/0/8
```

QoS

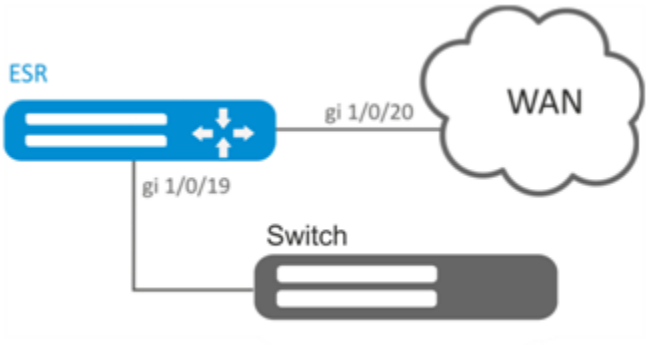
ESR , .

1	, QoS.		. (ACL).
2	QoS .	**esr(config)# class-map <NAME>**	<NAME> – , 31 .
3	QoS ( ).	**esr(config-class-map)# description <description>**	<description> – 255 .
4	, (ACL).	**esr(config-class-map)# match access-group <NAME>**	<NAME> – , 31 .
5	DSCP, IP-, (IP Precedence CoS) ( ).	**esr(config-class-map)# set dscp <DSCP>**	<DSCP> – DSCP, [0..63].
6	IP Precedence, IP-, (DSCP CoS) ( ).	**esr(config-class-map)# set ip-precedence <IPP>**	<IPP> – IP Precedence, [0..7].
7	802.1p , , (DSCP IP Precedence) ( ).	**esr(config-class-map)# set os <COS>**	<COS> – 802.1p , [0..7].
8	QoS .	**esr(config)# policy-map <NAME>** **esr(config-policy-map)#**	<NAME> – , 31 .
9	QoS ( ).	**esr(config-policy-map)# description <description>**	<description> – 255 .
10	.	**esr(config-policy-map)# shape average { <BANDWIDTH> | percent <BANDWIDTH_PERCENT> } [BURST]**	<BANDWIDTH> – /, [64..10000000]; <BANDWIDTH_PERCENT> – %, ( ): • **shape average** ; • **traffic-shape** , bridge, ; • **speed** . [1..100]. <BURST> – , [128..16000]. 128 .
11	, , ( ).	**esr(config-policy-map)# shape auto-distribution**	
12	QoS- .	**esr(config-policy-map)# class <NAME>** **esr(config-class-policy-map)#**	<NAME> – , 31 . «class-default» , .
13	QoS QoS QoS.	**esr(config-class-policy-map)# service-policy <NAME>**	<NAME> – , 31 . .
14	( ).	**esr(config-class-policy-map)# shape average { <BANDWIDTH> | percent <BANDWIDTH_PERCENT> } [BURST]**	<BANDWIDTH> – /, [64..10000000]; <BANDWIDTH_PERCENT> – %, ( ): ▪ **shape average** ; ▪ **traffic-shape** , bridge, ; ▪ **speed** . [1..100]. <BURST> – , [4..16000]. 128 .

15	. , ( ).	**esr(config-class-policy-map)# shape peak { <BANDWIDTH> | percent <BANDWIDTH_PERCENT> } [BURST]**	<BANDWIDTH> – priority class /, , [64..10000000]; <BANDWIDTH_PERCENT> – priority class %, , ( ): • **shape average** ; • **traffic-shape** , bridge, ; • **speed** . [1..100]. <BURST> – , [4..16000]. 128 .
16	( ).	**esr(config-class-policy-map)# mode <MODE>**	<MODE> – : • **fifo** – FIFO (First In, First Out); • **gred** – GRED (Generalized RED); • **red** – RED (Random Early Detection); • **sfq** – SFQ (SFQ). : **FIFO**.
17	WRR- ( ).	**esr(config-class-policy-map)# priority class <PRIORITY>**	<PRIORITY> – WRR-, [1..8]. .
18	StrictPriority ( ).	**esr(config-class-policy-map)# priority level <PRIORITY>**	<PRIORITY> – StrictPriority-, [1..8]. . : WRR, .
19	( ).	**esr(config-class-policy-map)# fair-queue <QUEUE-LIMIT>**	<QUEUE-LIMIT> – , [16..4096]. : 16.
20	( ).	**esr(config-class-policy-map)# queue-limit <QUEUE-LIMIT>**	<QUEUE-LIMIT> – , [2..4096]. : 127.
21	RED (Random Early Detection) ( ).	**esr(config-class-policy-map)# random-detect <LIMIT> <MAX> <MIN> <PROBABILITY>**	<LIMIT> – , [1..1000000]; <MAX> – , [1..1000000]; <MIN> – , [1..1000000]; <PROBABILITY> – , [0..100]. : • <MAX>> 2 * <MIN> • <LIMIT>> 3 * <MAX>
22	GRED (Generalized Random Early Detection) ( ).	**esr(config-class-policy-map)# random-detect precedence <PRECEDENCE><LIMIT><MAX><MIN><PROBABILITY>**	<PRECEDENCE> – IP Precendence [0..7]; <LIMIT> – , [1..1000000]; <MAX> – , [1..1000000]; <MIN> – , [1..1000000]; <PROBABILITY> – , [0..100]. : • <MAX>> 2 * <MIN> • <LIMIT>> 3 * <MAX>
23	tcp- ( ).	**esr(config-class-policy-map)# compression header ip tcp**	
24	QoS // .	**esr(config-if-gi)# qos enable**	
25	QoS // (input) (output) .	**esr(config-if-gi)# service-policy { input | output } <NAME>**	<NAME> – QoS-, 31 .

:

(10.0.11.0/24, 10.0.12.0/24), DSCP (38 42) (40 / 60 /), 250 /, SFQ.



:

, :

```
esr(config)# ip access-list extended fl1
esr(config-acl)# rule 1
esr(config-acl-rule)# action permit
esr(config-acl-rule)# match protocol any
esr(config-acl-rule)# match source-address 10.0.11.0 255.255.255.0
esr(config-acl-rule)# match destination-address any
esr(config-acl-rule)# enable
esr(config-acl-rule)# exit
esr(config-acl)# exit
esr(config)# ip access-list extended fl2
esr(config-acl)# rule 1
esr(config-acl-rule)# action permit
esr(config-acl-rule)# match protocol any
esr(config-acl-rule)# match source-address 10.0.12.0 255.255.255.0
esr(config-acl-rule)# match destination-address any
esr(config-acl-rule)# enable
esr(config-acl-rule)# exit
esr(config-acl)# exit
```

fl1 fl2, , :

```
esr(config)# class-map fl1
esr(config-class-map)# set dscp 38
esr(config-class-map)# match access-group fl1
esr(config-class-map)# exit
esr(config)# class-map fl2
esr(config-class-map)# set dscp 42
esr(config-class-map)# match access-group fl2
esr(config-class-map)# exit
```

:

```
esr(config)# policy-map fl
esr(config-policy-map)# shape average 250000
```

, :

```
esr(config-policy-map)# class fl1
esr(config-class-policy-map)# shape average 40000
esr(config-class-policy-map)# exit
esr(config-policy-map)# class fl2
esr(config-class-policy-map)# shape average 60000
esr(config-class-policy-map)# exit
```

shape average percent.

SFQ:

```
esr(config-policy-map)# class class-default
esr(config-class-policy-map)# mode sfq
esr(config-class-policy-map)# fair-queue 800
esr(config-class-policy-map)# exit
esr(config-policy-map)# exit
```

QoS ， gi 1/0/19 gi1/0/20 SFQ ：

```
esr(config)# interface gigabitethernet 1/0/19
esr(config-if-gi)# qos enable
esr(config-if-gi)# service-policy input fl
esr(config-if-gi)# exit
esr(config)# interface gigabitethernet 1/0/20
esr(config-if-gi)# qos enable
esr(config-if-gi)# service-policy output fl
esr(config-if-gi)# exit
```

：

```
esr# do show qos policy statistics gigabitethernet 1/0/20
```