Configuring Weighted Random Early Detection

Last Updated: December 13, 2011

Feature History

Release	Modification
Cisco IOS	For information about feature support in Cisco IOS software, use Cisco Feature Navigator.

This module describes the tasks for configuring Weighted Random Early Detection (WRED), distributed WRED (DWRED), flow-based WRED, and DiffServ Compliant WRED on a router.

Note WRED is useful with adaptive traffic such as TCP/IP. With TCP, dropped packets indicate congestion, so the packet source will reduce its transmission rate. With other protocols, packet sources may not respond or may resend dropped packets at the same rate. Thus, dropping packets does not decrease congestion. WRED treats non-IP traffic as precedence 0, the lowest precedence. Therefore, non-IP traffic is more likely to be dropped than IP traffic. You cannot configure WRED on the same interface as Route Switch Processor (RSP)-based custom queueing (CQ), priority queueing (PQ), or weighted fair queueing (WFQ). However, you can configure both DWRED and DWFQ on the same interface.

Random Early Detection (RED) is a congestion avoidance mechanism that takes advantage of the congestion control mechanism of TCP. By randomly dropping packets prior to periods of high congestion, RED tells the packet source to decrease its transmission rate. WRED drops packets selectively based on IP precedence. Edge routers assign IP precedences to packets as they enter the network. (WRED is useful on any output interface where you expect to have congestion. However, WRED is usually used in the core routers of a network, rather than at the edge.) WRED uses these precedences to determine how it treats different types of traffic.

When a packet arrives, the following events occur:

1. The average queue size is calculated.
2. If the average is less than the minimum queue threshold, the arriving packet is queued.
3. If the average is between the minimum queue threshold for that type of traffic and the maximum threshold for the interface, the packet is either dropped or queued, depending on the packet drop probability for that type of traffic.
4. If the average queue size is greater than the maximum threshold, the packet is dropped.

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Weighted Random Early Detection Configuration Task List

Enabling WRED

Note To avoid counter issues do not configure WRED and queue-limit on the same interface at the same time.

Command	Purpose
`Router(config-if)#` **random-detect**	Enables WRED. If you configure this command on a Versatile Interface Processor (VIP) interface, DWRED is enabled.

Changing WRED Parameters

Note The default WRED parameter values are based on the best available data. We recommend that you do not change the parameters from their default values unless you have determined that your applications will benefit from the changed values.

Command	Purpose
`Router(config-if)#` **random-detect exponential-weighting-constant** *exponent*	Configures the weight factor used in calculating the average queue length.
`Router(config-if)#` **random-detect precedence** *precedence min-threshold max-threshold mark-prob-denominator*	Configures parameters for packets with a specific IP Precedence. The minimum threshold for IP Precedence 0 corresponds to half the maximum threshold for the interface. Repeat this command for each precedence. To configure RED, rather than WRED, use the same parameters for each precedence.

Monitoring WRED

Command	Purpose
`Router#` **show queue** *interface-type interface-number*	Displays the header information of the packets inside a queue. This command does not support DWRED.
`Router#` **show queueing interface** *interface-number* `[`**vc** `[[`*vpi* `/]` *vci*`]]`	Displays the WRED statistics of a specific virtual circuit (VC) on an interface.
`Router#` **show queueing random-detect**	Displays the queueing configuration for WRED.
`Router#` **show interfaces** `[`*type slot* `|` *port-adapter* `|` *port*`]`	Displays WRED configuration on an interface.

DWRED Configuration Task List

Configuring DWRED in a Traffic Policy

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Steps 3, 4, and 5 are optional. If you do not want to configure the exponential weight factor, specify the amount of bandwidth, or specify the number of queues to be reserved, you can skip these three steps and continue with step 6.
4. Router(config-pmap-c)# **random-detect exponential-weighting-constant** *exponent*
5. Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*
6. Router(config-pmap-c)# **fair-queue queue-limit** *queue-values*
7. Router(config-pmap-c)# **queue-limit** *number-of-packets*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the traffic policy to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a traffic class to be created and included in the traffic policy
Step 3	Steps 3, 4, and 5 are optional. If you do not want to configure the exponential weight factor, specify the amount of bandwidth, or specify the number of queues to be reserved, you can skip these three steps and continue with step 6.	
Step 4	Router(config-pmap-c)# **random-detect exponential-weighting-constant** *exponent*	Configures the exponential weight factor used in calculating the average queue length.
Step 5	Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*	Specifies the amount of bandwidth, in kbps, to be assigned to the traffic class.
Step 6	Router(config-pmap-c)# **fair-queue queue-limit** *queue-values*	Specifies the number of queues to be reserved for the traffic class.
Step 7	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the maximum number of packets that can be queued for the specified traffic class.

Configuring DWRED to Use IP Precedence Values in a Traffic Policy

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **random-detect exponential-weighting-constant** *exponent*
4. Router(config-pmap-c)# **random-detect precedence** *precedence min-threshold max-threshold mark-prob-denominato r*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the traffic policy to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a traffic class to associate with the traffic policy
Step 3	Router(config-pmap-c)# **random-detect exponential-weighting-constant** *exponent*	Configures the exponential weight factor used in calculating the average queue length.
Step 4	Router(config-pmap-c)# **random-detect precedence** *precedence min-threshold max-threshold mark-prob-denominato r*	Configures the parameters for packets with a specific IP Precedence. The minimum threshold for IP Precedence 0 corresponds to half the maximum threshold for the interface. Repeat this command for each precedence.

Monitoring and Maintaining DWRED

Command	Purpose
`Router#` **show policy-map**	Displays all configured traffic policies.
`Router#` **show policy-map** *policy-map-name*	Displays the user-specified traffic policy.
`Router#` **show policy-map interface**	Displays statistics and configurations of all input and output policies attached to an interface.
`Router#` **show policy-map interface** *interface-spec*	Displays configuration and statistics of the input and output policies attached to a particular interface.
`Router#` **show policy-map interface** *interface-spec input*	Displays configuration and statistics of the input policy attached to an interface.

Command	Purpose
`Router#` **show policy-map interface** *interface-spec output*	Displays configuration statistics of the output policy attached to an interface.
`Router#` **show policy-map interface** [*interface-spec* [*input* | **output**] [*class class-name*]]]	Displays the configuration and statistics for the class name configured in the policy.

Flow-Based WRED Configuration Task List

Configuring Flow-Based WRED

SUMMARY STEPS

1. Router(config-if)# **random-detect flow**
2. Router(config-if)# **random-detect flow average-depth-factor** *scaling-factor*
3. Router(config-if)# **random-detect flow count** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **random-detect flow**	Enables flow-based WRED.
Step 2	Router(config-if)# **random-detect flow average-depth-factor** *scaling-factor*	Sets the flow threshold multiplier for flow-based WRED.
Step 3	Router(config-if)# **random-detect flow count** *number*	Sets the maximum flow count for flow-based WRED.

DiffServ Compliant WRED Configuration Task List

Configuring WRED to Use the Differentiated Services Code Point Value

WRED at the Interface Level

SUMMARY STEPS

1. Router(config-if)# **random-detect** *dscp-based*
2. Router(config-if)# **random-detect dscp** *dscpvalue min-threshold max-threshold*[*mark-probability-denominator*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **random-detect** *dscp-based*	Indicates that WRED is to use the DSCP value when it calculates the drop probability for the packet.
Step 2	Router(config-if)# **random-detect dscp** *dscpvalue min-threshold max-threshold*[*mark-probability-denominator*]	Specifies the minimum and maximum thresholds, and, optionally, the mark-probability denominator for the specified DSCP value.

WRED at the per-VC Level

SUMMARY STEPS

1. Router(config)# **random-detect-group** *group-name dscp-based*
2. Router(cfg-red-grp)# **dscp** *dscpvalue min-threshold max-threshold*[*mark-probability-denominator*]
3. Router(config-atm-vc)# **random-detect**[**attach** *group-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **random-detect-group** *group-name dscp-based*	Indicates that WRED is to use the DSCP value when it calculates the drop probability for the packet.
Step 2	Router(cfg-red-grp)# **dscp** *dscpvalue min-threshold max-threshold*[*mark-probability-denominator*]	Specifies the DSCP value, the minimum and maximum packet thresholds and, optionally, the mark-probability denominator for the DSCP value.
Step 3	Router(config-atm-vc)# **random-detect**[**attach** *group-name*]	Enables per-VC WRED or per-VC VIP-DWRED.

WRED at the Class Level

SUMMARY STEPS

1. Router(config-if)# **class-map** *class-map-name*
2. Router(config-cmap)# **match** *match criterion*
3. Router(config-if)# **policy-map** *policy-map*
4. Router(config-pmap)# **class** *class-map-name*
5. Router(config-pmap-c)# **bandwidth** {*bandwidth-kbps* | **percent** *percent*}
6. Router(config-pmap-c)# **random-detect dscp-based**
7. Router(config-pmap-c)# **random-detect dscp** *dscpvalue min-threshold max-threshold*[*mark-probability-denominator*]
8. Router(config-if)# **service-policy output** *policy-map*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **class-map** *class-map-name*	Creates a class map to be used for matching packets to a specified class.
Step 2	Router(config-cmap)# **match** *match criterion*	Configures the match criteria for a class map.
Step 3	Router(config-if)# **policy-map** *policy-map*	Creates or modifies a policy map that can be attached to one or more interfaces to specify a traffic policy.
Step 4	Router(config-pmap)# **class** *class-map-name*	Specifies the QoS actions for the default class.
Step 5	Router(config-pmap-c)# **bandwidth** {*bandwidth-kbps* | **percent** *percent*}	Specifies or modifies the bandwidth allocated for a class belonging to a policy map.
Step 6	Router(config-pmap-c)# **random-detect dscp-based**	Indicates that WRED is to use the DSCP value when it calculates the drop probability for the packet.
Step 7	Router(config-pmap-c)# **random-detect dscp** *dscpvalue min-threshold max-threshold*[*mark-probability-denominator*]	Specifies the minimum and maximum packet thresholds and, optionally, the mark-probability denominator for the DSCP value.
Step 8	Router(config-if)# **service-policy output** *policy-map*	Attaches a policy map to an output interface or VC to be used as the traffic policy for that interface or VC.

Verifying the DSCP Value Configuration

Command	Purpose
`Router#` **show queueing interface**	Displays the queueing statistics of an interface or VC.
`Router#` **show policy-map interface**	Displays the configuration of classes configured for traffic policies on the specified interface or permanent virtual circuit (PVC).

WRED Configuration Examples

Example WRED Configuration

The following example enables WRED with default parameter values:

```
interface Serial5/0
 description to qos1-75a
 ip address 200.200.14.250 255.255.255.252
 random-detect
```

Use the **show interfaces** command output to verify the configuration. Notice that the "Queueing strategy" report lists "random early detection (RED)."

```
Router# show interfaces serial 5/0
Serial5/0 is up, line protocol is up
  Hardware is M4T
  Description: to qos1-75a
  Internet address is 200.200.14.250/30
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 237/255
  Encapsulation HDLC, crc 16, loopback not set
  Keepalive not set
  Last input 00:00:15, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:05:08
  Input queue: 0/75/0 (size/max/drops); Total output drops: 1036
  Queueing strategy: random early detection(RED)
  5 minutes input rate 0 bits/sec, 2 packets/sec
  5 minutes output rate 119000 bits/sec, 126 packets/sec
     594 packets input, 37115 bytes, 0 no buffer
     Received 5 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     37525 packets output, 4428684 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Use the **show queue** command output to view the current contents of the interface queue. Notice that there is only a single queue into which packets from all IP precedences are placed after dropping has taken place. The output has been truncated to show only three of the five packets.

```
Router# show queue serial 5/0

Output queue for Serial5/0 is 5/0
Packet 1, linktype: ip, length: 118, flags: 0x288
  source: 190.1.3.4, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 128 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
Packet 2, linktype: ip, length: 118, flags: 0x288
  source: 190.1.3.5, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 160 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
Packet 3, linktype: ip, length: 118, flags: 0x280
  source: 190.1.3.6, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 192 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
```

Use the **show queueing** command output to view the current settings for each of the precedences. Also notice that the default minimum thresholds are spaced evenly between half and the entire maximum threshold. Thresholds are specified in terms of packet count.

```
Router# show queueing
Current random-detect configuration:
     Serial5/0
       Queueing strategy:random early detection (WRED)
       Exp-weight-constant:9 (1/512)
       Mean queue depth:28

       Class   Random       Tail    Minimum    Maximum     Mark
                 drop       drop  threshold  threshold  probability
         0        330          0         20         40     1/10
         1        267          0         22         40     1/10
         2        217          0         24         40     1/10
         3        156          0         26         40     1/10
         4         61          0         28         40     1/10
         5          6          0         31         40     1/10
         6          0          0         33         40     1/10
         7          0          0         35         40     1/10
        rsvp        0          0         37         40     1/10
```

Example Parameter-Setting DWRED

The following example specifies the same parameters for each IP precedence. Thus, all IP precedences receive the same treatment. Start by enabling DWRED.

```
interface FastEthernet1/0/0
 ip address 200.200.14.250 255.255.255.252
 random-detect
```

Next, enter the **show queueing random-detect** command to determine reasonable values to use for the precedence-specific parameters.

```
Router# show queueing random-detect
Current random-detect configuration:
  FastEthernet2/0/0
    Queueing strategy:fifo
    Packet drop strategy:VIP-based random early detection (DWRED)
    Exp-weight-constant:9 (1/512)
    Mean queue depth:0
    Queue size:0       Maximum available buffers:6308
    Output packets:5  WRED drops:0  No buffer:0
    Class   Random       Tail    Minimum    Maximum     Mark       Output
              drop       drop  threshold  threshold  probability  Packets
      0          0          0        109        218      1/10            5
      1          0          0        122        218      1/10            0
      2          0          0        135        218      1/10            0
      3          0          0        148        218      1/10            0
      4          0          0        161        218      1/10            0
      5          0          0        174        218      1/10            0
      6          0          0        187        218      1/10            0
      7          0          0        200        218      1/10            0
```

Complete the configuration by assigning the same parameter values to each precedence. Use the values obtained from the **show queueing random-detect** command output to choose reasonable parameter values.

```
interface FastEthernet1/0/0
 random-detect precedence 0 100 218 10
 random-detect precedence 1 100 218 10
 random-detect precedence 2 100 218 10
 random-detect precedence 3 100 218 10
 random-detect precedence 4 100 218 10
 random-detect precedence 5 100 218 10
 random-detect precedence 6 100 218 10
 random-detect precedence 7 100 218 10
```

Example Parameter-Setting WRED

The following example enables WRED on the interface and specifies parameters for the different IP precedences:

```
interface Hssi0/0/0
 description 45Mbps to R1
 ip address 10.200.14.250 255.255.255.252
 random-detect
 random-detect precedence 0 32 256 100
 random-detect precedence 1 64 256 100
 random-detect precedence 2 96 256 100
 random-detect precedence 3 120 256 100
 random-detect precedence 4 140 256 100
 random-detect precedence 5 170 256 100
 random-detect precedence 6 290 256 100
 random-detect precedence 7 210 256 100
 random-detect precedence rsvp 230 256 100
```

DWRED Configuration Examples

Example DWRED on an Interface

The following example configures DWRED on an interface with a weight factor of 10:

```
Router(config)# interface hssi0/0/0
Router(config-if)# description 45mbps to R1
Router(config-if)# ip address 192.168.14.250 255.255.255.252
Router(config-if)# random-detect
Router(config-if)# random-detect exponential-weighting-constant 10
```

Example Modular QoS CLI

The following example enables DWRED using the Legacy CLI (non-Modular QoS Command-Line Interface) feature on the interface and specifies parameters for the different IP precedences:

```
interface Hssi0/0/0
 description 45Mbps to R1
 ip address 200.200.14.250 255.255.255.252
 random-detect
 random-detect precedence 0 32 256 100
 random-detect precedence 1 64 256 100
 random-detect precedence 2 96 256 100
 random-detect precedence 3 120 256 100
 random-detect precedence 4 140 256 100
 random-detect precedence 5 170 256 100
 random-detect precedence 6 290 256 100
 random-detect precedence 7 210 256 100
 random-detect precedence rsvp 230 256 100
```

The following example uses the Modular QoS CLI to configure a traffic policy called policy10. For congestion avoidance, WRED packet drop is used, not tail drop. IP Precedence is reset for levels 0 through 5.

```
policy-map policy10
 class acl10
 bandwidth 2000
 random-detect exponential-weighting-constant 10
 random-detect precedence 0 32 256 100
 random-detect precedence 1 64 256 100
 random-detect precedence 2 96 256 100
 random-detect precedence 3 120 256 100
 random-detect precedence 4 140 256 100
 random-detect precedence 5 170 256 100
```

Example Configuring DWRED in Traffic Policy

The following example configures policy for a traffic class named int10 to configure the exponential weight factor as 12. This is the weight factor used for the average queue size calculation for the queue for traffic class int10. WRED packet drop is used for congestion avoidance for traffic class int10, not tail drop.

```
policy-map policy12
 class int10
 bandwidth 2000
 random-detect exponential-weighting-constant 12
```

Flow-Based WRED Configuration Example

The following example enables WRED on the serial interface 1 and configures flow-based WRED. The **random-detect** interface configuration command is used to enable WRED. Once WRED is enabled, the **random-detect flow** command is used to enable flow-based WRED.

After flow-based WRED is enabled, the **random-detect flow average-depth-factor** command is used to set the scaling factor to 8 and the **random-detect flow count** command is used to set the flow count to 16. The scaling factor is used to scale the number of buffers available per flow and to determine the number of packets allowed in the output queue for each active flow.

```
configure terminal
interface Serial1
 random-detect
 random-detect flow
 random-detect flow average-depth-factor 8
 random-detect flow count 16
end
```

The following part of the example shows a sample configuration file after the previous flow-based WRED commands are issued:

```
Router# more system:running-config
Building configuration...
Current configuration:
!
version 12.0
service timestamps debug datetime msec localtime
service timestamps log uptime
no service password-encryption
service tcp-small-servers
!
no logging console
enable password lab
!
```

```
clock timezone PST -8
clock summer-time PDT recurring
ip subnet-zero
no ip domain-lookup
!
interface Ethernet0
 no ip address
 no ip directed-broadcast
 no ip mroute-cache
 shutdown
!
interface Serial0
 no ip address
 no ip directed-broadcast
 no ip mroute-cache
 no keepalive
 shutdown
!
interface Serial1
 ip address 190.1.2.1 255.255.255.0
 no ip directed-broadcast
 load-interval 30
 no keepalive
 random-detect
 random-detect flow
 random-detect flow count 16
 random-detect flow average-depth-factor 8
!
router igrp 8
 network 190.1.0.0
!
ip classless
no ip http server
!
line con 0
 transport input none
line 1 16
 transport input all
line aux 0
 transport input all
line vty 0 4
 password lab
 login
!
end
```

DiffServ Compliant WRED Configuration Examples

Example WRED Configured to Use the DSCP Value

The following example configures WRED to use the DSCP value 8. The minimum threshold for the DSCP value 8 is 24 and the maximum threshold is 40. This configuration was performed at the interface level.

```
Router(config-if)# interface seo/0
Router(config-if)# random-detect dscp-based
Router(config-if)# random-detect dscp 8 24 40
```

The following example enables WRED to use the DSCP value 9. The minimum threshold for the DSCP value 9 is 20 and the maximum threshold is 50. This configuration can be attached to other VCs, as required.

```
Router(config)# random-detect-group sanjose dscp-based
```

```
Router(cfg-red-grp)# dscp 9 20 50
Router(config-subif-vc)# random-detect attach sanjose
```

The following example enables WRED to use the DSCP value 8 for the class c1. The minimum threshold for the DSCP value 8 is 24 and the maximum threshold is 40. The last line attaches the traffic policy to the output interface or VC p1.

```
Router(config-if)# class-map c1
Router(config-cmap)# match access-group 101
Router(config-if)# policy-map p1
Router(config-pmap)# class c1
Router(config-pmap-c)# bandwidth 48
Router(config-pmap-c)# random-detect dscp-based
Router(config-pmap-c)# random-detect dscp 8 24 40
Router(config-if)# service-policy output p1
```

Example DSCP Value Configuration Verification

When WRED has been configured to use the DSCP value when it calculates the drop probability of a packet, all entries of the DSCP table are initialized with the appropriate default values. The example in the following section are samples of the **show policy interface** command for WRED at the class level.

This example displays packet statistics along with the entries of the DSCP table, confirming that WRED has been enabled to use the DSCP value when it calculates the drop probability for a packet.

```
Router# show policy interface Serial6/3

 Serial6/3
  Service-policy output: test
    Class-map: c1 (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol ip
        0 packets, 0 bytes
        5 minute rate 0 bps
      Weighted Fair Queueing
        Output Queue: Conversation 265
        Bandwidth 20 (%)
        Bandwidth 308 (kbps)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
         exponential weight: 9
         mean queue depth: 0
dscp     Transmitted       Random drop      Tail drop    Minimum Maximum  Mark
          pkts/bytes        pkts/bytes       pkts/bytes    thresh  thresh  prob
af11          0/0               0/0              0/0           32      40  1/10
af12          0/0               0/0              0/0           28      40  1/10
af13          0/0               0/0              0/0           24      40  1/10
af21          0/0               0/0              0/0           32      40  1/10
af22          0/0               0/0              0/0           28      40  1/10
af23          0/0               0/0              0/0           24      40  1/10
af31          0/0               0/0              0/0           32      40  1/10
af32          0/0               0/0              0/0           28      40  1/10
af33          0/0               0/0              0/0           24      40  1/10
af41          0/0               0/0              0/0           32      40  1/10
af42          0/0               0/0              0/0           28      40  1/10
af43          0/0               0/0              0/0           24      40  1/10
cs1           0/0               0/0              0/0           22      40  1/10
cs2           0/0               0/0              0/0           24      40  1/10
cs3           0/0               0/0              0/0           26      40  1/10
cs4           0/0               0/0              0/0           28      40  1/10
cs5           0/0               0/0              0/0           30      40  1/10
cs6           0/0               0/0              0/0           32      40  1/10
cs7           0/0               0/0              0/0           34      40  1/10
ef            0/0               0/0              0/0           36      40  1/10
rsvp          0/0               0/0              0/0           36      40  1/10
default       0/0               0/0              0/0           20      40  1/10
```