

What is the Barracuda Syslog?

The Barracuda Spam & Virus Firewall generates syslog messages as a means of logging both changes to the Web interface configuration and what happens to each message as it is processed. The syslog messages are stored in text file format on the Barracuda Spam & Virus Firewall and can be sent to a remote server configurable by the administrator. There are two syslog outputs you can monitor: the *Mail* syslog and the *Web* syslog.

The *Web* syslog contains information about user login activities and any configuration changes made to the Barracuda Spam & Virus Firewall Web interface. User activity data appears on the local facility with login information at the *info priority level*, and configuration changes appear at the *debug priority level* on the specified syslog server. See the **Syslog** section of the **ADVANCED > Troubleshooting** page for the facility to open a browser window and view the *Web* syslog output. Since *Web* syslog messages do not use any special formatting, *Web* syslog is not covered in this guide.

The *Mail syslog* logs what happens to each message as it is processed and is presented in a raw data format that includes reason codes relative to the message process. This guide will help you understand, parse, and utilize the mail syslog messages and reason codes generated by the Barracuda Spam & Virus Firewall, version 5.x.

Configuring the Barracuda Mail Syslog

To configure the *Mail* syslog, using the Barracuda Spam & Virus Firewall Web interface, navigate to the **ADVANCED > Advanced Networking** page and enter the IP address of the syslog server to which syslog data related to mail flow should be sent.

Syslog data is the same information as that used to build the Message Log in the Barracuda Spam & Virus Firewall and includes data such as the connecting IP Address, envelope 'From' address, envelope 'To' address, and the spam score for the messages transmitted. This syslog data appears on the mail facility at the debug priority level on the specified syslog server. As the Barracuda Spam & Virus Firewall uses the syslog messages internally for its own message logging, it is not possible to change the facility or the priority level. See the **Syslog** section of the **ADVANCED > Troubleshooting** page in the Barracuda Spam & Virus Firewall Web interface to open a window and view the Mail syslog output.

If you are running syslog on a UNIX machine, be sure to start the syslog daemon process with the “-r” option so that it can receive messages from sources other than itself. Windows users will have to install a separate program to utilize syslog since the Windows OS doesn't include syslog capability. Kiwi Syslog is a popular solution, but there are many others available to choose from, both free and commercial.

Syslog messages are sent via UDP to the standard syslog port of 514. If there are any firewalls between the Barracuda Spam & Virus Firewall and the server receiving the syslog messages, make sure that port 514 is open on the firewalls.

Barracuda Syslog Format

The Barracuda Spam & Virus Firewall sends syslog messages in the following format. Whenever an action is taken on a message, it is logged with the syslog. A message sent to multiple recipients will be logged separately for each recipient. Please be aware that the various syslog implementations may not display the messages in this exact format. However, the sections should still be present in the syslog lines as shown in the table below. The following represents the main part of the syslog line.:

```
Timestamp      Host Barracuda Process Client IP  Message ID  Start      End      Service  Info
Sep  8 17:38:48 dev1 inbound/pass1[27564]: XX.XX.XX.XX 1126226282-27564-2-0 1126226286 1126226328 RECV [ . . . ]
```

Syslog Section:	Timestamp
The time that the syslog message was logged. For reporting purposes, this section of the syslog line can be ignored. It is useful when analyzing the logs by hand, but is not needed for compiling reports.	
Syslog Section:	Host
Indicates the host that generated the syslog message. Useful if you have multiple Barracuda appliances and need to know which host sent the message.	
Syslog Section:	Barracuda Process
Indicates the process that the email message was in when the syslog message was generated. Possibilities are: inbound/pass1 ... inbound/pass2 ... scan ... outbound/smt	
Syslog Section:	Barracuda Message ID
The most important piece of the syslog entry. This ID is used to uniquely identify a message. The ID may occur in one of two formats (a different format is used for the inbound process and for the scan process). For example, this ID 1126226282-27564-2-0 is used for RECV transactions and it means the following: 1126226282 = UNIX timestamp 27564-2 = Internal Process ID 0 = Message number in SMTP session – this number indicates how many messages have been sent in that single SMTP session	
Syslog Section:	Start
The start time of the message in UNIX timestamp format, indicating when the sender began giving us the “From” information for the message.	
Syslog Section:	End
The end time of the message in UNIX timestamp format, indicating when the sending server terminated sending of the message.	
Syslog Section:	Service
The service that produced the message. The following services are available: <ul style="list-style-type: none"> • RECV: This service indicates a message was handled by the MTA and processing stopped. • SCAN: This service indicates the message was scanned and processing may have stopped or it may have been sent to the outbound processing for delivery. • SEND: This service indicates status of outbound delivery. It is the only message that may appear multiple times for a given message ID since delivery may initially have been deferred before succeeding later on. 	

Syslog Section:	Info
-----------------	------

This section contains the actual information about what happened to a given message. It is dependent on the service that sent the information, and the following formats are used:

RECV: Sender Recipient Action Reason ReasonExtra

SCAN: Encrypted Sender Recipient Score Action Reason ReasonExtra "SUBJ:"Subject

SEND: Encrypted Action QueueID Response

The possible fields have the following meanings:

- **Sender:** The address of the sender if available, and '-' if not available.
- **Recipient:** The address of the recipient if available and, '-' if not available.
- **Action:** The action code indicating what action was taken for the message. For the "SEND" service these action codes have different meanings.
- **Reason:** The reason code indicating the reason for the taken action.
- **ReasonExtra:** Extra information about a given reason (e.g. the RBL or the body filter that matched in the message).

- **Encrypted:** Indicates whether or not the message was received or sent encrypted.
- **Score:** The score given to the message if the scoring mechanism was run.
- **Subject:** The subject of the message if it was available.
- **QueueID:** The queue ID of the message on the Barracuda as delivery is being attempted.
- **Response:** The response given back by the mail server if available.

Barracuda Action Codes

Barracuda Action Codes (RECV and SCAN services)		Barracuda Action Codes (SEND service)	
ID	Meaning	ID	Meaning
0	Allowed Message	1	Delivered Message
1	Aborted Message	2	Rejected Message
2	Blocked Message	3	Deferred Message
3	Quarantined Message	4	Expired Message
4	Tagged Message		
5	Deferred Message		
6	Per-User Quarantined Message		
7	Whitelisted Message		
8	Encrypted Message		
9	Redirected Message		

Barracuda Reason Codes

Barracuda Reason Codes (RECV and SCAN services)			
ID	Meaning	ID	Meaning
1	Virus	46	Too Many Recipients
2	Banned Attachment	47	Need RCPT Command
3	RBL Match	48	DATA Syntax Error
4	Rate Control	49	Internal Error
5	Too Many Message In Session	50	Too Many Hops
6	Timeout Exceeded	51	Mail Protocol Error
7	No Such Domain	55	Invalid Parameter Syntax
8	No Such User	56	STARTTLS Syntax Error
9	Subject Filter Match	57	TLS Already Active
11	Client IP	58	Too Many Errors
12	Recipient Address	59	Need STARTTLS First
13	No Valid Recipients	60	Spam Fingerprint Found
14	Domain Not Found	61	Barracuda Reputation
15	Sender Address	62	Barracuda Blocklist
17	Need Fully Qualified Recipient	63	DomainKeys
18	Need Fully Qualified Sender	64	Recipient Verification Unavailable
19	Unsupported Command	65	Realtime Intent
20	MAIL FROM Syntax Error	66	Client Reverse DNS
21	Bad Address Syntax	67	Email Registry
22	RCPT TO Syntax Error	68	Invalid Bounce
23	Send EHLO/HELO First	69	Intent - Adult
24	Need MAIL Command	70	Intent - Political
25	Nested MAIL Command	71	Multi-Level Intent
27	EHLO/HELO Syntax Error	72	Attachment Limit Exceeded
30	Mail Protocol Violation	73	System Busy
31	Score	74	BRTS Intent
34	Header Filter Match	75	Per-Domain Recipient
35	Sender Block/Accept	76	Per-Domain Sender
36	Recipient Block/Accept	77	Per Domain Client IP
37	Body Filter Match	78	Sender Spoofed
38	Message Size Bypass	79	Attachment Content
39	Intention Analysis Match	80	Outlook Add-in
40	SPF/Caller-ID	82	Barracuda IP/Domain Reputation
41	Client Host Rejected	83	Authentication Failure
44	Authentication Not Enabled		
45	Allowed Message Size Exceeded		

For Programmers: Parsing the Barracuda Syslog

Syslog messages generated by the Barracuda Spam & Virus Firewall can be parsed for reporting purposes or for building of a custom message log. It is easiest to think of each syslog line in terms of the main components, and the INFO portion can then be parsed based on that service.

The following Perl code illustrates a simple parsing of the log lines. It takes a line and places the resulting message information into a hash – pushing that hash onto a global array of messages when it completes.

```

sub parse_log_line
{
    # Grab the line we were given and create a new message hash for our message
    my($line) = @_;
    my %message = ();

    # These are the components we may have parsed out of the message based on the service
    my ($ip, $id, $start_time, $end_time, $name, $info, $domain);
    my ($enc, $sender, $recip, $score, $action, $reason, $reason_extra, $subject);

    # Grab the main components from the line (IP, MSG_ID, START_TIME, END_TIME, SERVICE, INFO)
    #
    # NOTE: If this is for the SEND log line then the IP, as well as the START/END times are
    # bogus values of 127.0.0.1 and 0/0 respectively
    if( $line =~ /\|:\s+([\s]+) ([^\s]+) (\d+) (\d+) (RCV|SCAN|SEND) (.*)$/ )
    {
        # Grab the main pieces of the log entry and the process specific info
        ($ip, $id, $start_time, $end_time, $name, $info) = ($1, $2, $3, $4, $5, $6);

        # Set the connecting IP, message-id, start-time, and end-time if this wasn't
        # for the SEND service
        if( $name !~ /SEND/ )
        {
            $message{client} = $ip;
            $message{id} = $id;
            $message{start_time} = $start_time;
            $message{end_time} = $end_time;
        }

        # Break out the process specific pieces from the info portion
        if( $name =~ /RCV/ )
        {
            # Break the MTA info up into sender/recip/action/reason/reason_extra
            if( $info =~ /([\s]+\s)([\s]+\s)([\d]+\s)([\d]+\s)(.*)$/ )
            {
                ($sender, $recip, $action, $reason, $reason_extra) = ($1, $2, $3, $4, $5);

                # Store the readable time of this message based on when it was started by
                # converting the unix time to its components and then sprintf'ing into readable form
                my ($sec,$min,$hour,$mday,$mon,$year,$yday,$yday,$isdst) = localtime($start_time);
                $message{time} = sprintf("%02d/%02d/%02d %02d:%02d:%02d", $mon+1, $mday, $year-100, $hour, $min, $sec);

                # Store the sender if we had one
                if( $sender ne '-' )
                {
                    $message{from} = $sender;
                }

                # Store the recipient if we had one
                if( $recip ne '-' )
                {
                    $message{mailto} = $recip;
                }

                # Set our action/reason codes
                $message{action_id} = $action;
                $message{reason_id} = $reason;

                # Pull in the reason_extra field. This should never be anything other
                # than ASCII since the mta doesn't have any multi-byte functionality
                # ... thus we don't need to eval it.
                if( $reason_extra ne '-' )
                {
                    $message{reason_extra} = " ($reason_extra)";
                }
            }
        }
    }
    elsif( $name =~ /SCAN/ )
    {
        # Break the scanner info up into encrypted/sender/recip/score/action/reason/reason_extra/subject
        if( $info =~ /([\s]+\s)([\s]+\s)([\s]+\s)([\s]+\s)([\d]+\s)([\d]+\s)([\d]+\s)([\d]+\s)(.*)\sSUBJ:(.*)$/ )
        {
            ($enc, $sender, $recip, $score, $action, $reason, $reason_extra, $subject) =
                ($1, $2, $3, $4, $5, $6, $7, $8);

            # Store the readable time of this message based on when it was started by
            # converting the unix time to its components and then sprintf'ing into readable form
            my ($sec,$min,$hour,$mday,$mon,$year,$yday,$yday,$isdst) = localtime($start_time);
            $message{time} = sprintf("%02d/%02d/%02d %02d:%02d:%02d", $mon+1, $mday, $year-100, $hour, $min, $sec);

            # Store the sender if we had one
            if( $sender ne '-' )
            {
                $message{from} = $sender;
            }
        }
    }
}

```



```
# Store the recipient if we had one and build the msg_file path
if( $recipient ne '-' )
{
    $message{mailto} = $recipient;
}

# Set the subject line
if( $subject )
{
    eval
    {
        # Note: if this is encoded you may want to decode it here and that
        # is why this section is in an eval - since nothing guarantees the
        # sender encoded the subject properly.
        $message{subject} = decode( $subject );
    };
}

# Set the score if we had one
if( $score ne '-' )
{
    $message{spam_score} = $score;
}

# Set our action/reason codes
$message{action_id} = $action;
$message{reason_id} = $reason;

# Pull in the reason_extra field. This has the extra info the filter that matched
# and other things that might be multi-byte so it should probably be eval'd
eval
{
    if( $reason_extra ne '-' )
    {
        $message{reason_extra} = decode( $reason_extra );
    }
}

}
elseif( $name =~ /SEND/ )
{
    # Break the Outbound MTA info up into encrypted/action/queue_id/response
    if( $info =~ /([^\s]+\s(\d+)\s([^\s]+\s(.*)$/ )
    {
        my ( $enc, $action, $queue_id, $reason ) = ( $1, $2, $3, $4 );

        # Do whatever you would like with the delivery transactions - just keep in
        # mind that a single message may have multiple outbound entries because of
        # being deferred by the downstream server.
    }
}

# Put a ref to this message onto our array of messages so we can use it later
push( @message_list, \%message );

# Send back whatever info you would like to the caller here. In this case
# we are sending back the end time as an example that could handle tracking
# last seen message time or something similar
return( $end_time );
}

# No message info to send back
return undef;
}
```

For questions after reading this document, please call Barracuda Networks at 1-408-342-5400, or, if you are in the United States, (888) Anti-Spam, or (888) 268-4772. You can also send an email to support@barracuda.com.