



Creating Threat Prevention Connection Rules

The Threat Prevention feature runs a connection rules script each time a client tries to connect to ePrism. The script determines whether to accept or reject a connection based on its threat prevention history. The script is also responsible for moving IP addresses into appropriate dynamic lists, such as "infected" or "spammers".

The full script itself is not editable, but it is updated with the condition statements and actions that are defined for each Threat Prevention rule. These rules are configurable, and the system will check the script when new rules are applied to ensure there are no syntax or execution errors.

Basic Rule Structure

The basic structure of a connection rule is as follows:

- **Rule Condition** — A set of criteria that must be met for the rule to be triggered, such as "`stats1h.virus > 10`" (10 or greater virus-infected messages sent in the last hour). ePrism collects over 15 different types of data that can be used to create a rule condition.
- **Action** — Action to take when the rule condition is met, such as "Accept" or "Reject".
- **Reply code** — The reply code to send back to the sending server, such as temporarily reject (450) or permanently reject (550).
- **Add to Dynamic List** — Add the IP address to a configured dynamic list, if applicable. For example, a sender that triggers a spam rule can be placed in the "spammers" dynamic list.

Connection Rules Script

Select **Mail Delivery** → **Threat Prevention** on the menu, and then click the **Advanced** button to see the entire default connection rules script based on the configured pre-defined rules. The full default connection rules script is as follows:

```
// This script is generated based on the conditions configured above

// This script is called at each time a client tries to
// connect, to accept or reject based on Threat Prevention history.
// It must contain the evaluate_rules() method.
// It drives reject/accept, and also moves IP addresses into dynamic
// lists.
// If an IP is already in a dynamic list with action "reject",
// it will not get to this script at all.
evaluate_rules() {

    int is_internal, is_blacklist, is_mynetworks, is_peers;

    // determine what pre-defined IP lists this IP belongs to
    // internal: non routable IPs per RFCs
    is_internal = ip_on_list("internal");
    // blacklist: permanent blacklisted clients
    is_blacklist = ip_on_list("blacklist");
    // mynetworks: subnets we own or control
    is_mynetworks = ip_on_list("mynetworks");
    // peers: special sites, such as fellow ISPs
    is_peers = ip_on_list("peers");

    // Blacklisted clients
    if (is_blacklist) {
        action = "reject";
        reason = "client ip not accepted";
        reply_code = 550;
        rule_no = 10;
        change_to_group = "blacklisted";
        return;
    }
    // Directory harvesters
    if (stats30m.bad_recipients >= 50 && stats30m.good_recipients < 3 &&
(!is_internal && !is_mynetworks )) {
        action = "reject";
        reason = "too many unknown recipients";
        reply_code = 550;
        rule_no = 20;
        change_to_group = "harvesters";
        return;
    }
    // Big virus senders
    if (stats1h.virus > 10 && stats1h.perc_virus_to_messages > 50 &&
stats1h.perc_ham_to_messages < 25 && (!is_internal && !is_mynetworks ))
    {
        action = "reject";
        reason = "too many viruses seen from %IP%";
        reply_code = 550;
        rule_no = 30;
    }
}
```

This section initializes the pre-defined static lists.

The connection rules begin here.

```
        change_to_group = "infected";
        return;
    }
    // DNSBL clients (on more than one list)
    if (block_list > 1 && (!is_internal && !is_mynetworks )) {
        action = "reject";
        reason = "not accepting mail from %IP%";
        reply_code = 450;
        rule_no = 40;
        change_to_group = "spammers";
        return;
    }
    // DNSBL clients
    if (block_list == 1 && stats30m.bad_mail > 10 && stats30m.ham < 2 &&
(!is_internal && !is_mynetworks )) {
        action = "reject";
        reason = "not accepting mail from %IP%";
        reply_code = 450;
        rule_no = 50;
        change_to_group = "spammers";
        return;
    }
    // Junk senders
    if (stats1h.bad_mail > 20 && stats1h.perc_ham_to_spam < 25 &&
stats5m.messages > 10 && (!is_internal && !is_mynetworks )) {
        action = "reject";
        reason = "exceeded quota";
        reply_code = 450;
        rule_no = 60;
        change_to_group = "tarpit";
        return;
    }
    // Internal DoS
    if (open_connections > 50 && is_internal) {
        action = "reject";
        reason = "too many open connections";
        reply_code = 450;
        rule_no = 70;
        return;
    }
    // External DoS
    if (open_connections > 20 && !is_internal) {
        action = "reject";
        reason = "too many open connections";
        reply_code = 450;
        rule_no = 80;
        return;
    }
    // Excessive senders
    if (!is_peers && !is_internal && stats1h.messages > 50000) {
        action = "reject";
        reason = "too many messages in the last hour";
        reply_code = 450;
        rule_no = 90;
        return;
    }
}
```

```
// default rule
action = "accept";
reason = "";
reply_code = 220;
change_to_group = "";
}
```

This is the default rule used when no connection rule match is found.

Default Connection Rules

The default connection rules are active when the Threat Prevention feature is enabled. These rules include checks for typical conditions such as blacklisted clients, virus and junk mail senders, and denial of service (DoS) attempts. The default rules are also helpful in learning how to put together condition statements for customized connection rules.

The screenshot shows the 'Threat Prevention Configuration' window. At the top, there is a checkbox for 'Enable Threat Prevention' which is checked. Below this is a section titled 'Connection Rules' containing a table with the following columns: Rule, Condition, List, Action, Reject Code, and Move. The table lists several default rules such as 'Blacklisted clients', 'Directory harvesters', 'Big virus senders', 'DNSBL clients', 'Junk senders', 'Internal DoS', 'External DoS', and 'Excessive senders'. At the bottom of the window, there are buttons for 'Apply', 'Add Rule', 'Advanced', 'Reset to Defaults', and 'Help'.

Rule	Condition	List	Action	Reject Code	Move
Blacklisted clients	is_blacklist	blacklisted	reject	550	▼
Directory harvesters	stats30m.bad_recipients >= 50 && stats30m.good_recipients < 3	harvesters	reject	550	▲▼
Big virus senders	stats1h.virus > 10 && stats1h.perc_virus_to_messages > 50 && stats1h.perc_ham_to_messages < 25	infected	reject	550	▲▼
DNSBL clients (on more than one list)	block_list > 1	spammers	reject	450	▲▼
DNSBL clients	block_list == 1 && stats30m.bad_mail > 10 && stats30m.ham < 2	spammers	reject	450	▲▼
Junk senders	stats1h.bad_mail > 20 && stats1h.perc_ham_to_spam < 25 && stats5m.messages > 10	tarpit	reject	450	▲▼
Internal DoS	open_connections > 50 && is_internal		reject	450	▲▼
External DoS	open_connections > 20 && !is_internal		reject	450	▲▼
Excessive senders	lis_peers && lis_internal && stats1h.messages > 50000		reject	450	▲

Note: Any of the default rules can be customized to change any aspect of the rule to better suit the needs of your organization.

Blacklisted clients

This rule checks if the client is already blacklisted by ePrism. The condition statement "is_blacklist" simply checks if the client is listed in the *blacklist* static IP address list. If the check is true, the client will be rejected and added to the *blacklisted* dynamic IP address list.

Directory harvesters

This rule checks if the client has been involved with directory harvesting activities intended to discover valid e-mail addresses from ePrism. The following condition statement is used to identify if a client is considered a directory harvester:

```
stats30m.bad_recipients >= 50 && stats30m.good_recipients < 3 &&
(!is_internal && !is_mynetworks )
```

This statement indicates:

- If the number of invalid recipients from the client in the last 30 minutes is greater than or equal to 50
- and the number of good recipients from the client in the last 30 minutes is less than 3
- and the client does not exist in the *internal* or *mynetworks* static lists (to whitelist the client)
- then the connecting system is rejected and entered into the *harvesters* dynamic IP address list

Big virus senders

This rule checks if the client has recently sent a large number of viruses. The following condition statement is used to identify if a client is considered a source of viruses:

```
stats1h.virus > 10 && stats1h.perc_virus_to_messages > 50 &&  
stats1h.perc_ham_to_messages < 25 && (!is_internal &&  
!is_mynetworks )
```

This statement indicates:

- If the number of viruses received from this client in the last hour is greater than 10
- and the percentage of virus infected messages received from this client in the last hour is greater than 50
- and the percentage of clean messages received from this client in the last hour is less than 25
- and the client does not exist in the *internal* or *mynetworks* static lists (to whitelist the client)
- then the connecting system is rejected and entered into the *infected* dynamic IP address list

DNSBL clients (on more than one list)

This rule checks if the client has been listed on more than one DNS Block List of blacklisted clients. If the client is on more than one DNSBL, it is a known open-relay that may send out a large number of spam messages. The following condition statement is used to identify if a client is on more than one DNSBL:

```
block_list > 1 && (!is_internal && !is_mynetworks )
```

This statement indicates:

- If the client exists on more than one DNSBL
- and the client does not exist in the *internal* or *mynetworks* static lists (to whitelist the client)
- then the connecting system is temporarily rejected and entered into the *spammers* dynamic list

DNSBL clients

This rule checks if the client exists on only one DNS Block List. In this case, there is the possibility that the client is on this DNSBL by mistake, and ePrism makes additional checks to examine its recent history of mail messages. The following condition statement is used to identify if a client is on one DNSBL and sends a large number of spam messages:

```
block_list == 1 && stats30m.bad_mail > 10 && stats30m.ham < 2 &&  
(!is_internal && !is_mynetworks )
```

This statement indicates:

- If the client exists on only one DNSBL
- and the number of spam and junk messages received from this client in the last 30 minutes is greater than 10
- and the number of clean messages received from this client in the last 30 minutes is less than 2
- and the client does not exist in the *internal* or *mynetworks* static lists (to whitelist the client)
- then the connecting system is temporarily rejected and entered into the *spammers* dynamic IP address list

Junk senders

This rule checks if the client sends out a large amount of spam or junk mail in proportion to the number of legitimate messages. The following condition statement is used to identify if a client is sending a large amount of spam or junk messages compared to legitimate messages:

```
stats1h.bad_mail > 20 && stats1h.perc_ham_to_spam < 25 &&  
stats5m.messages > 10 && (!is_internal && !is_mynetworks)
```

This statement indicates:

- If the number of spam and junk messages received from this client in the last hour is greater than 20
- and the percentage of clean messages compared to spam received from this client in the last hour is less than 25
- and the number of messages sent from this client in the last five minutes is greater than 10
- and the client does not exist in the *internal* or *mynetworks* static lists (to whitelist the client)
- then the connecting system is temporarily rejected and entered into the *tarpit* dynamic IP address list

Internal DoS

This rule checks if the client is on an internal network and is using a lot of open connections that may result in a denial of service. The following condition statement is used to identify if an internal client is creating a large amount of open connections:

```
open_connections > 50 && is_internal
```

This statement indicates:

- If the number of open connections from this client is greater than 50
- and the client is listed in the *internal* static address list
- then the connecting system is temporarily rejected

External DoS

This rule checks if an external client is using a lot of open connections that may result in a denial of service. The following condition statement is used to identify if an external client is creating a large amount of open connections:

```
open_connections > 20 && !is_internal
```

This statement indicates:

- If the number of open connections from this client is greater than 20
- and the client is not listed in the *internal* static address list
- then the connecting system is temporarily rejected

Excessive senders

This rule checks if a client is sending too many messages that could result in a denial of service. The following condition statement is used to identify if a client is sending an abnormal amount of messages:

```
!is_peers && !is_internal && stats1h.messages > 50000
```

This statement indicates:

- If the client is not listed in the *peers* and *internal* static address lists (to whitelist the client)
- and the number of messages sent from this client in the last hour is greater than 50000
- then the connecting system is temporarily rejected

Creating Connection Rules

To create customized connection rules for the Threat Prevention feature, select **Mail Delivery** → **Threat Prevention** on the menu, and then click the **Add Rule** button.

The screenshot shows a dialog box titled "Add New Connection Rule". It contains the following fields and values:

- Description: Sample Rule
- Condition: stats1h.bad_mail > 20 && (!is_internal && !is_mynetworks)
- Action: Reject mail
- Reject Code: 450 (temporary)
- Reject Message: Too many bad messages from client %IP%
- Add to List: tarpit - Tar Pit

At the bottom of the dialog are three buttons: Apply, Cancel, and Help.

The following options can be configured:

- **Description** — Enter a descriptive summary of the rule.
- **Condition** — Enter a condition statement to execute, such as:

```
stats1h.bad_mail > 20 && (!is_internal && !is_mynetworks)
```

This statement checks if the client has sent more than 20 virus-infected or spam messages in the last hour, and is not on the *internal* or *mynetworks* IP address lists.

Note: See the next section "**Building Condition Statements**" for detailed information on creating these statements.

- **Action** — Action to take if the condition is "True". Options are *Accept Mail* or *Reject Mail*.
- **Reject Code** — Reply code to send to the connecting client. For *Reject*, this is 450 (temporary) or 550 (permanent). For *Accept*, the reply code is set to 220.
- **Reject Message** — A customized reject message to send to the connecting client. The `%IP%` variable can be used to indicate the IP address of the client.
- **Add to List** — Select a Dynamic Address List to add the client IP address to if the condition is true. These lists can be viewed and configured via **Mail Delivery** → **Threat Prevention** → **Dynamic Lists**.

Building Condition Statements

The Threat Prevention rules are based on condition statements that are used to create various criteria for the connecting clients and their historical behaviour.

The following tables describe the variables, parameters, and Boolean operators available to create Threat Prevention rules.

General Statistics

The following are general statistics that can be used when creating connection rules. They include items such as the IP address of the connecting client and how many open connections a client is using.

Statistic	Description
ip_address	The IP address of the connecting client.
current_group	The name of the current Dynamic list the client IP addresses is in, if any.
open_connections	The current number of open connections to this IP address.
block_list	If DNS Block lists are enabled, this indicates the number of lists the IP address matched.
rule_no	Indicates the connection rule number for ordering purposes.

For example, as part of your condition statement to prevent denial of service attacks, check that the client does not have a large amount of open connections:

```
open_connections > 50
```

IP Lists

The following parameters indicate if the client IP address is listed in any of the pre-defined Static IP lists (defined via **Mail Delivery** → **Threat Prevention** → **Static Lists** on the menu.)

This allows you to check if the client IP address is whitelisted because it is identified as an internal system, a network under your control, or a peer address. The client can also be blacklisted if it appears in the local blacklist.

Static IP List	Description
is_internal	Checks if the client IP address is listed in the <i>internal</i> address list.
is_mynetworks	Checks if the client IP address is listed in the <i>mynetworks</i> address list.
is_peers	Checks if the client IP address is listed in the <i>peers</i> address list.
is_blacklist	Checks if the client IP address is listed in the <i>blacklisted</i> address list.

For example, to check if the connecting client is in the *blacklist* static IP list, use the following condition statement:

```
is_blacklist
```

If the client is already listed in the *blacklist* IP list, the condition is true and the configured action executed.

These lists can also be used to ensure clients are whitelisted because they are considered internal or under an organization's control. For example, to check for a large amount of open connections, and to ensure this client is not an internal client, use the following statement:

```
open_connections > 50 && !is_internal
```

This statement checks clients who have more than 50 open connections and do not belong to the *internal* static IP list.

E-mail Statistics

The following e-mail statistics can be used to build condition statements in the connection rules based on the types of messages received. These statistics identify the number of messages based on their classification, such as virus-infected, malformed, spam, and clean. Several statistics also indicate the percentage of one type of message to another, such as the percentage of spam messages to total messages received.

E-mail Statistic	Description
messages	Total number of messages from successful connections.
virus	Number of virus-infected messages.
malformed	Number of malformed messages.
spam	Number of spam messages (Intercept Certainly Spam or Probably Spam, PBMF spam, and Brightmail).
ham	Number of messages that were clean (not spam, virus, or malformed).
connection_attempts	Number of attempted connection attempts.
bad_mail	Number of viruses, malformed, and spam messages.
bad_recipients	Number of unknown recipients (or 0 if the "Reject on unknown recipient" feature is disabled).
good_recipients	Number of legitimate recipients.
perc_ham_to_messages	Percentage of clean messages to the total amount of messages.
perc_virus_to_messages	Percentage of virus-infected messages to the total amount of messages.
perc_spam_to_messages	Percentage of spam messages to the total amount of messages.
perc_malformed_to_messages	Percentage of malformed messages to the total amount of messages.
perc_bad_to_messages	Percentage of bad messages (virus, malformed, and spam) to the total amount of messages.
perc_ham_to_spam	Percentage of clean messages to the total amount of spam messages.

These e-mail statistics must be used in combination with a specific time period. This allows you to check for the number of certain types of e-mail messages, such as "spam" messages, in a certain time period such as 24 hours.

The following table describes various time periods that can be used in conjunction with the e-mail statistics variables.

Time Period	Description
stats1m	Statistics for the last minute
stats5m	Statistics for the last 5 minutes
stats15m	Statistics for the last 15 minutes
stats30m	Statistics for the last 30 minutes
stats1h	Statistics for the last hour
stats24h	Statistics for the last 24 hours (1 day)

Specify the time period and the e-mail statistics parameter separated by a "." (period).

For example, to check how many spam messages were received in the last 24 hours, use the following:

```
stats24h.spam
```

To check the percentage of the number of spam messages compared to the total amount of messages in the last hour, use the following:

```
stats1h.perc_spam_to_messages
```

Boolean Operators and Syntax

The following are the Boolean operators that can be used when building condition statements.

To combine operators, use the following syntax to ensure the order: (a && (b || c)). This indicates the result of "a" AND ("b" OR "c").

Boolean Operator	Description
&&	and
!	not
	or
>	Greater than
<	Less than
==	Equal to
>=	Greater than or equal to
<=	Less than or equal to

For example, to ensure a host is not listed in the *internal* and *mynetworks* static lists (to whitelist the system for Threat Prevention) use the following:

```
!is_internal && !is_mynetworks
```

The following example shows how to use multiple Boolean operators to combine condition statements:

```
stats30m.bad_recipients >= 50 && stats30m.good_recipients < 3
```

This example checks the number of good and bad recipients in the last 30 minutes. If the bad recipients are greater than or equal to 50, and the good recipients are less than 3, then the condition is true.

Connection Rules Script Error Checking

When you are finished with the changes and additions to the connection rules, click the **Apply** button.

Add New Connection Rule

Description: i

Condition: i

Action: i

Reject Code: i

Reject Message: i

Add to List: i

The results of the script test will be shown, including any syntax errors if they occur. If the script does not contain any errors, the following will be displayed:

```
rules test results:
tdr_server.spl 90978 1.0 ready
0 accept 220 unknown 0 unknown
tdr_server.spl exit
Rules check ok
Rules are now active.
```

If an error occurs, a message similar to the following will be displayed:

```
Error in rules server script

A system error has occurred.
```

Examine the rule you just applied and check the condition statement to ensure that it conforms to the proper syntax and that any variables or parameters are entered correctly.