Manual:IP/Proxy

From MikroTik Wiki
< Manual:IP

Contents

Applies to RouterOS: v3, v4

Summary

Sub-menu: `/ip proxy`

Standards: `RFC 1945, RFC 2616`

MikroTik RouterOS performs proxying of HTTP and HTTP-proxy (for FTP and HTTP protocols) requests. Proxy server performs Internet object cache function by storing requested Internet objects, i.e., data available via HTTP and FTP protocols on a system positioned closer to the recipient in the form of speeding up customer browsing by delivering them requested file copies from proxy cache at local network speed. MikroTik RouterOS implements the following proxy server features:

- Regular HTTP proxy – customer (itself) specify what is proxy server for him
- Transparent proxy – customer does not know about the proxy being enabled and there isn't need any additional configuration for web browser of client.
- Access list by source, destination, URL and requested method (HTTP firewall)
- Cache access list to specify which objects to cache, and which not.
- Direct Access List – to specify which resources should be accessed directly, and which - through another proxy server
- Logging facility – allows to get and to store information about proxy operation
- Parent proxy support – allows to specify other proxy server, *('**if they don't have the requested object ask their parents, or to the original server.)***

A proxy server usually is placed at various points between users and the destination server (*also known as origin server*) on the Internet. *(see Figure 10.1)*.

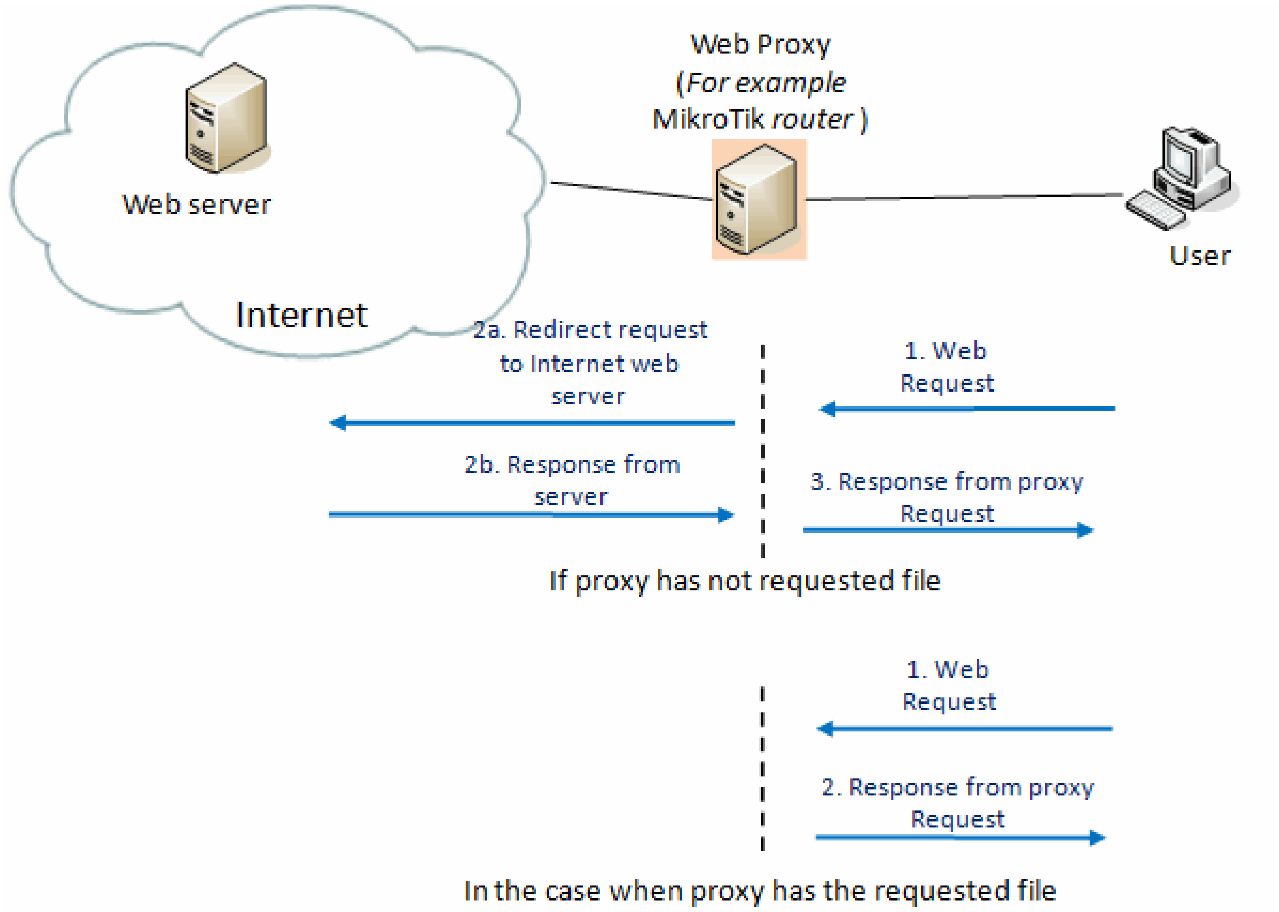
Web server
Internet
Web Proxy
(*For example*
MikroTik *router*)
User
2a. Redirect request to Internet web server
1. Web Request
2b. Response from server
3. Response from proxy Request
If proxy has not requested file
1. Web Request
2. Response from proxy Request
In the case when proxy has the requested file

Figure 10.1. Web proxy basic operation scheme

A *Web proxy (cache)* watches requests coming from client, saving copies of the responses for itself. Then, if there is another request for the same URL, it can use the response that it has, instead of asking the origin server for it again. If proxy has not requested file, it downloads that from the original server.

There can be many potential purpose of proxy server:

- To increase access speed to resources (it takes less time for the client to get the object).
- Works as HTTP firewall (deny access to undesirable web pages),

Allows to filter web content (by specific parameters, like source address, destination address and port, URL, HTTP request method) scan outbound content, e.g., for data leak protection.

Note: it may be useful to have Web proxy running even with no cache when you want to use it only as something like HTTP and FTP firewall (for example, denying access undesired web pages or deny specific type of files e.g. .mp3 files) or to redirect requests to external proxy (possibly, to a proxy with caching functions) transparently.

Proxy configuration example

In MikroTik RouterOS proxy configuration is performed in **/ip proxy** menu. See below how to enable the proxy on port 8080 and set up 195.10.10.1 as proxy source address:

```
[admin@MikroTik] ip proxy> set enabled=yes port=8080 src-address=195.10.10.1

[admin@MikroTik] ip proxy> print
                     enabled: yes
                 src-address: 195.10.10.1
                        port: 8080
                parent-proxy: 0.0.0.0:0
                 cache-drive: system
         cache-administrator: "admin@mikrotik.com"
         max-disk-cache-size: none
          max-ram-cache-size: 100000KiB
          cache-only-on-disk: yes
  maximal-client-connections: 1000
  maximal-server-connections: 1000
              max-fresh-time: 3d
```

When setting up regular proxy service, make sure it serves only your clients and prevent unauthorised access to it by creating firewall that allow only your clients to use proxy, otherwise it may be used as an open proxy.

Remember that regular proxy require also client's web browser configuration.

For example:

Explorer 8.x	**Firefox 3.x**	**Opera 10.x**
Select *Tools>Internet options.* Click the *Connections tab.* Select the necessary connection and choose *Settings* button.	Select *Tools>Options.* Click the *Advanced tab.* Open the *Network tab.* Click the *Connection/Settings*	Select *Tool>Preferences.* Open the *Advanced tab/Network.* Click the *Proxy servers.* *Enter proxy address and port.*

Configure proxy address and port.	Select *Manual proxy configuration'*	

Transparent proxy configuration example

RouterOS can also act as a Transparent Caching server, with no configuration required in the customer’s web browser. Transparent proxy does not modify requested URL or response. RouterOS will take all HTTP requests and redirect them to the local proxy service. This process will be entirely transparent to the user (users may not know anything about proxy server that is located between them and original server), and the only difference to them will be the increased browsing speed.

To enable the transparent mode, firewall rule in destination NAT has to be added, specifying which connections (to which ports) should be transparently redirected to the proxy. Check proxy settings above and redirect us users (192.168.1.0/24) to proxy server.

```
[admin@MikroTik] ip firewall nat> add chain=dstnat protocol=tcp src-address=192.168.1.0/24 \
dst-port=80 action=redirect to-ports=8080

[admin@MikroTik] ip firewall nat> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=dstnat protocol=tcp dst-port=80 action=redirect to-ports=8080
[admin@MikroTik] ip firewall nat>
```

The web proxy can be used as transparent and normal web proxy at the same time. In transparent mode it is possible to use it as standard web proxy, too. However, in this case, proxy users may have trouble to reach web pages which are accessed transparently.

Proxy based firewall – Access List

Access list is implemented in the same way as MikroTik firewall rules processed from the top to the bottom. First matching rule specifies decision of what to do with this connection. Connections can be matched by its source address, destination address, destination port, sub-string of requested URL (Uniform Resource Locator) or request method. If none of these parameters is specified, every connection will match this rule.

If connection is matched by a rule, action property of this rule specifies whether connection will be allowed or not (deny). If connection does not match any rule, it will be allowed.

In this example assume that we have configured transparent proxy server as given in example above.

Block particular Websites.

```
/ip proxy access add dst-host=www.facebook.com action=deny
```

It will block website http://www.facebook.com, we can always block the same for different networks by giving src-address.

```
/ip proxy access add src-address=192.168.1.0/24 dst-host=www.facebook.com action=deny
```

Users from network 192.168.1.0/24 will not be able to access website www.facebook.com (http://www.facebook.com).

You can block also websites that contain specific words in URL:

```
/ip proxy access add dst-host=:mail action=deny
```

This statement will block all websites which contain word “mail” in URL. Like www.mail.com (http://www.mail.com), www.hotmail.com (http://www.hotmail.com), mail.yahoo.com etc.

We can also stop downloading specific types of files like .flv, .avi, .mp4, .mp3, .exe, .dat, …etc.

```
/ip proxy access
add path=*.flv action=deny
add path=*.avi action=deny
add path=*.mp4 action=deny
add path=*.mp3 action=deny
add path=*.zip action=deny
add path=*.rar action=deny.
```

Here are available also different wildcard characters, to creating specific conditions and to match it by proxy access list.

Wildcard properties (dst-host and dst-path) match a complete string (i.e., they will not match "example.com" if they are set to "example"). Available wildcards are '*' (match any number of any characters) and '?' (match any one character).

Regular expressions are also accepted here, but if the property should be treated as a regular expression, it should start with a colon (':').

To show that no symbols are allowed before the given pattern, we use ^ symbol at the beginning of the pattern.

To specify that no symbols are allowed after the given pattern, we use $ symbol at the end of the pattern.

Enabling RAM or Store based caching.

In this example it will presumed that you already have proxy configured and working and you just want to enable caching. If command/parameter detailed description is required check reference section which is located right below example section.

- RAM based caching:
  - Good if you have device with considerable amount of RAM for caching. Enabling this on device with RAM 256MB or less will not give your network any benefit.
  - Way faster cache write/read than one that is stored on usb or sata connected mediums.

- Store based caching:
  - Larger proxy caches available simply due to medium capacity differences.

RAM proxy cache:

Important commands:

- max-cache-size=
- max-cache-object-size=
- cache-on-disk=

```
[admin@MikroTik] /ip proxy> set max-cache-size=unlimited max-cache-object-size=50000KiB cache-
...
[admin@MikroTik] /ip proxy> print
                 enabled: yes
             src-address: ::
                    port: 8080
               anonymous: no
            parent-proxy: 0.0.0.0
       parent-proxy-port: 0
     cache-administrator: webmaster
          max-cache-size: unlimited  <-------
   max-cache-object-size: 500000KiB  <-------
           cache-on-disk: no  <-------
  max-client-connections: 600
  max-server-connections: 600
          max-fresh-time: 3d
   serialize-connections: no
       always-from-cache: no
          cache-hit-dscp: 4
```

```
              cache-path: proxy-cache
```

Store proxy cache:

Important commands:

- max-cache-size=
- max-cache-object-size=
- cache-on-disk=
- cache-path=

```
[admin@MikroTik] > ip proxy set cache-on-disk=yes cache-path=/usb1/proxy/cache

[admin@MikroTik] > ip proxy print
                 enabled: yes
             src-address: ::
                    port: 8080
               anonymous: no
            parent-proxy: 0.0.0.0
       parent-proxy-port: 0
     cache-administrator: webmaster
          max-cache-size: unlimited  <-------
   max-cache-object-size: 50000KiB  <-------
           cache-on-disk: yes  <-------
  max-client-connections: 600
  max-server-connections: 600
          max-fresh-time: 3d
   serialize-connections: no
       always-from-cache: no
          cache-hit-dscp: 4
              cache-path: usb1/proxy/cache  <-------

[admin@MikroTik] > file print
 # NAME                                                                TYPE
 0 skins                                                               directory
 5 usb1/proxy                                                          directory
 6 usb1/proxy/cache                                                    web-proxy store   <-------
 7 usb1/lost+found                                                     directory
```

Notes:

- This example shows how to configure caching for version starting from v6.20 as stores are now located in file menu as directories.

Check if cache is working:

```
[admin@MikroTik] > ip proxy monitor
                 status: running
                 uptime: 2w20h28m25s
     client-connections: 15
     server-connections: 7
               requests: 79772
                   hits: 30513
             cache-used: 481KiB
         total-ram-used: 1207KiB
```

```
received-from-servers: 4042536KiB
      sent-to-clients: 4399757KiB
 hits-sent-to-clients: 176934KiB
```

Reference

List of all available parameters and commands per menu.

General

Sub-menu: `/ip proxy`

Property	Description
`always-from-cache` (*yes | no*; Default: **no**)	
`cache-administrator` (*string*; Default: **webmaster**)	Administrator's e-mail displayed on proxy error page
`cache-hit-dscp` (*integer: 0..63*; Default: **4**)	
`cache-on-disk` (*yes | no*; Default: **no**)	
`max-cache-size` (*none | unlimited | integer: 0..4294967295*; Default: **none**)	Specifies the maximal cache size, measured in kilobytes
`max-client-connections` (*integer: Dynamic* ; Default: **600**)	Maximal number of

	connections accepted from clients (any further connections will be rejected)
`max-fresh-time` (*time*; Default: **3d**)	Maximal time to store a cached object. The validity period of an object is is usually defined by the object itself, but in case it is set too high, you can override the maximal value
`max-server-connections` (*integer: Dynamic* ; Default: **600**)	Maximal number of connections made to servers (any

	further connections from clients will be put on hold until some server connections will terminate)
`parent-proxy` (*Ip4 | ip6*; Default: **0.0.0.0**)	IP address and port of another HTTP proxy to redirect all requests to. If set to **0.0.0.0** parent proxy is not used.
`parent-proxy-port` (*integer: 0..65535*; Default: **0**)	Port that parent proxy is listening on.
`port` (*integer: 0..65535*; Default: **8080**)	TCP port the proxy server will be listening on. This

	port have to be specified on all clients that want to use the server as HTTP proxy. Transparent (with zero configuration for clients) proxy setup can be made by redirecting HTTP requests to this port in IP firewall using destination NAT feature
`serialize-connections` (*yes* | *no*; Default: **no**)	
`src-address` (*Ip4* | *Ip6*; Default: **0.0.0.0**)	Proxy will use specified address when connecting to parent proxy or web

	site. If set to **0.0.0.0** then appropriate IP address will be taken from routing table.

Menu Specific commands

Access List

Sub-menu: `/ip proxy access`

Access list is configured like a regular firewall rules. Rules are processed from the top to the bottom. First matching rule specifies decision of what to do with this connection. There is a total of 6 classifiers that specify matching constraints. If none of these classifiers is specified, the particular rule will match every connection.

If connection is matched by a rule, action property of this rule specifies whether connection will be allowed or not. If the particular connection does not match any rule, it will be allowed.

Property	Description
`action` (*allow | deny*; Default: **allow**)	

	Specifies whether to pass or deny matched packets
`dst-address` (*Ip4[-Ip4 | /0..32] | Ip6/0..128*; Default:)	Destination address of the target server.
`dst-host` (*string*; Default:)	IP address or DNS name used to make connection the target server (this is the string user wrote in browser before specifying port and path to a particular web page
`dst-port` (*integer[-integer[,integer[,...]]]: 0..65535*; Default:)	List or range of ports the packet is destined to
`local-port` (*integer: 0..65535*; Default:)	Specifies the port of

	the web proxy via which the packet was received. This value should match one of the ports web proxy is listening on.
`method` (*any | connect | delete | get | head | options | post | put | trace*; Default:)	HTTP method used in the request (see HTTP Methods section in the end of this document)
`path` (*string*; Default:)	Name of the requested page within the target server (i.e. the name of a particular web page or document

	without the name of the server it resides on)
`redirect-to` (*string*; Default:)	In case access is denied by this rule, the user shall be redirected to the URL specified here
`src-address` (*Ip4[-Ip4 | /0..32] | Ip6/0..128*; Default:)	Source address of the connection originator.

Read only properties:

Property	Description
`hits` (*integer*)	Count of requests that were matched by this rule

Wildcard properties (dst-host and dst-path) match a complete string (i.e., they will not match "example.com" if they are set to "example"). Available wildcards are '*' (match any number of any characters) and '?' (match any one character). Regular expressions are also accepted here, but if the property should be treated as a regular expression, it should start with a colon (':').

Small hints in using regular expressions:

- \\ symbol sequence is used to enter \ character in console
- \. pattern means . only (in regular expressions single dot in pattern means any symbol)
- to show that no symbols are allowed before the given pattern, we use ^ symbol at the beginning of the pattern
- to specify that no symbols are allowed after the given pattern, we use $ symbol at the end of the pattern
- to enter [or] symbols, you should escape them with backslash \.

It is strongly recommended to deny all IP addresses except those behind the router as the proxy still may be used to access your internal-use-only (intranet) web servers. Also, consult examples in Firewall Manual on how to protect your router.

Direct Access

Sub-menu: `/ip proxy direct`

If **parent-proxy** property is specified, it is possible to tell proxy server whether to try to pass the request to the parent proxy or to resolve it connecting to the requested server directly. Direct Access List is managed just like Proxy Access List described in the previous chapter except the action argument.

Unlike the access list, the direct proxy access list has default action equal to deny. It takes place when no rules are specified or a particular request did not match any rule.

Property	Description
`action` (*allow | deny*; Default: **allow**)	Specifies the action to perform on matched packets: ▪ `allow` - always resolve matched requests directly bypassing the parent router ▪ `deny` - resolve matched requests through the parent proxy. If no one is specified this has the same effect as **allow**.
`dst-address` (*Ip4[-Ip4 | /0..32] | Ip6/0..128*; Default:)	Destination address of the target server.
`dst-host` (*string*; Default:)	IP address or DNS name used to make connection the

	target server (this is the string user wrote in browser before specifying port and path to a particular web page
`dst-port` (*integer[-integer[,integer[,...]]]: 0..65535*; Default:)	List or range of ports used by connection to target server.
`local-port` (*integer: 0..65535*; Default:)	Specifies the port of the web proxy via which the packet was received. This value should match one of the ports web proxy is listening on.
`method` (*any | connect | delete | get |*	

head | options | post | put | trace; Default:)	HTTP method used in the request (see HTTP Methods section in the end of this document)
`path` (*string*; Default:)	Name of the requested page within the target server (i.e. the name of a particular web page or document without the name of the server it resides on)
`src-address` (*Ip4[-Ip4 | /0..32] | Ip6/0..128*; Default:)	Source address of the connection originator.

Read only properties:

Property	Description
hits (*integer*)	Count of requests that were matched by this rule

Cache Management

Sub-menu: `/ip proxy cache`

Cache access list specifies, which requests (domains, servers, pages) have to be cached locally by web proxy, and which not. This list is implemented exactly the same way as web proxy access list. Default action is to cache object (if no matching rule is found).

Property	Description
action (*allow | deny*; Default: **allow**)	Specifies the action to perform on matched packets: - allow - cache objects from matched request - deny - do not cache objects from matched request
dst-address (*Ip4[-Ip4 | /0..32] |*	

Ip6/0..128; Default:)	Destination address of the target server
`dst-host` (*string*; Default:)	IP address or DNS name used to make connection the target server (this is the string user wrote in browser before specifying port and path to a particular web page
`dst-port` (*integer[-integer[,integer[,...]]]: 0..65535*; Default:)	List or range of ports the packet is destined to.
`local-port` (*integer: 0..65535*; Default:)	Specifies the port of the web proxy via which the packet was received. This

	value should match one of the ports web proxy is listening on.
`method` (*any | connect | delete | get | head | options | post | put | trace*; Default:)	HTTP method used in the request (see HTTP Methods section in the end of this document)
`path` (*string*; Default:)	Name of the requested page within the target server (i.e. the name of a particular web page or document without the name of the server it resides on)
`src-address` (*Ip4[-Ip4 | /0..32] |*	

Ip6/0..128; Default:)	Source address of the connection originator

Read only properties:

Property	Description
`hits` (*integer*)	Count of requests that were matched by this rule

Menu Specific commands

Connections

Sub-menu: `/ip proxy connections`

This menu conntains the list of current connections the proxy is serving.

Read only properties:

Property	Description
`client` ()	

`dst-address` (*Ip4 | Ip6*)	IPv4/Ipv6 destination address of the connection
`protocol` (*string*)	Protocol name
`rx-bytes` (*integer*)	The amount of bytes received by the client
`server` ()	
`src-address` (*Ip4 | Ip6*)	Ipv4/ipv6 address of the connection originator
`state` (*closing | connecting | converting | hotspot | idle | resolving | rx-header | tx-body | tx-eof | tx-header | waiting*)	Connection state: ▪ `closing` - the data transfer is finished, and the connection is being finalized ▪ `connecting` - establishing toe connection ▪ `converting` - replacing header and footer fields in response or request paket ▪ `hotspot` - check if hotspot authentication allows to continue (for hotspot proxy) ▪ `idle` - staying idle ▪ `resolving` - resolving server's DNS name ▪ `rx-header` - receiving HTTP header ▪ `tx-body` - transmitting HTTP body to the client ▪ `tx-eof` - writing chunk-end (when converting to chunked response) ▪ `tx-header` - transmitting HTTP header to the client ▪ `waiting` - waiting for transmission form a

	peer
tx-bytes (*integer*)	The amount of bytes sent by the client

Cache Inserts

Sub-menu: `/ip proxy inserts`

This menu shows statistics on objects stored in cache (cache inserts).

Read only properties:

Property	Description
denied (*integer*)	Number of inserts denied by the caching list.
errors (*integer*)	Number of disk or other system-related errors
no-memory (*integer*)	

	Number of objects not stored because there was not enough memory
successes (*integer*)	Number of successfull cache inserts.
too-large (*integer*)	Number of objects too large to store

Cache Lookups

Sub-menu: `/ip proxy lookup`

This menu shows statistics on objects read from cache (cache lookups).

Read only properties:

Property	Description
denied (*integer*)	Number of requests denied by the access list.

`expired` (*integer*)	Number of requests found in cache, but expired, and, thus, requested from an external server
`no-expiration-info` (*integer*)	Conditional request received for a page that does not have the information to compare the request with
`non-cacheable` (*integer*)	Number of requests requested from the external servers unconditionally (as their caching is denied by the cache access list)
`not-found` (*integer*)	

	Number of requests not found in the cache, and, thus, requested from an external server (or parent proxy if configured accordingly)
`successes` (*integer*)	Number of requests found in the cache.

Cache Contents

Sub-menu: `/ip proxy cache-contents`

This menu shows cached contents.

Read only properties:

Property	Description
`file-size` (*integer*)	Cached object size
`last-accessed` (*time*)	
`last-accessed-time` (*time*)	

`last-modified` (*time*)	
`last-modified-time` (*time*)	
`uri` (*string*)	

HTTP Methods

Options

This method is a request of information about the communication options available on the chain between the client and the server identified by the **Request-URI**. The method allows the client to determine the options and (or) the requirements associated with a resource without initiating any resource retrieval

GET

This method retrieves whatever information identified by the Request-URI. If the Request-URI refers to a data processing process than the response to the GET method should contain data produced by the process, not the source code of the process procedure(-s), unless the source is the result of the process.

The GET method can become a conditional GET if the request message includes an If-Modified-Since, If-Unmodified-Since, If-Match, If-None-Match, or If-Range header field. The conditional GET method is used to reduce the network traffic specifying that the transfer of the entity should occur only under circumstances described by conditional header field(-s).

The GET method can become a partial GET if the request message includes a Range header field. The partial GET method intends to reduce unnecessary network usage by requesting only parts of entities without transferring data already held by client.

The response to a GET request is cacheable if and only if it meets the requirements for HTTP caching.

HEAD

This method shares all features of GET method except that the server must not return a message-body in the response. This retrieves the metainformation of the entity implied by the request which leads to a wide usage of it for testing hypertext links for validity, accessibility, and recent modification.

The response to a HEAD request may be cacheable in the way that the information contained in the response may be used to update previously cached entity identified by that Request-URI.

POST

This method requests that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI.

The actual action performed by the POST method is determined by the origin server and usually is Request-URI dependent.

Responses to POST method are not cacheable, unless the response includes appropriate Cache-Control or Expires header fields.

PUT

This method requests that the enclosed entity be stored under the supplied Request-URI. If another entity exists under specified Request-URI, the enclosed entity should be considered as updated (newer) version of that residing on the origin server. If the Request-URI is not pointing to an existing resource, the origin server should create a resource with that URI.

If the request passes through a cache and the Request-URI identifies one or more currently cached entities, those entries should be treated as stale. Responses to this method are not cacheable.

TRACE

This method invokes a remote, application-layer loop-back of the request message. The final recipient of the request should reflect the message received back to the client as the entity-body of a 200 (OK) response. The final recipient is either the origin server or the first proxy or gateway to receive a Max-Forwards value of 0 in the request. A TRACE request must not include an entity.

Responses to this method MUST NOT be cached.

[Top | Back to Content **]**