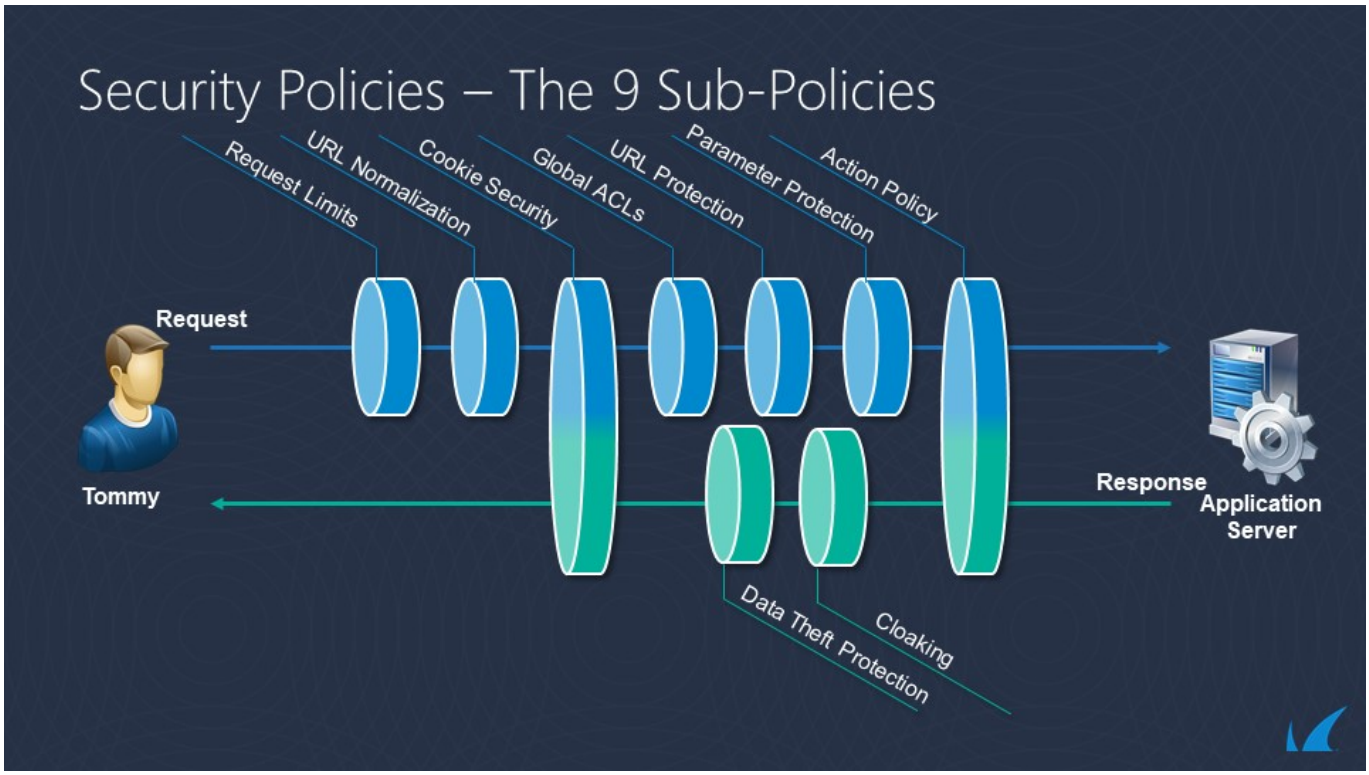
Evaluation Policy and Flow

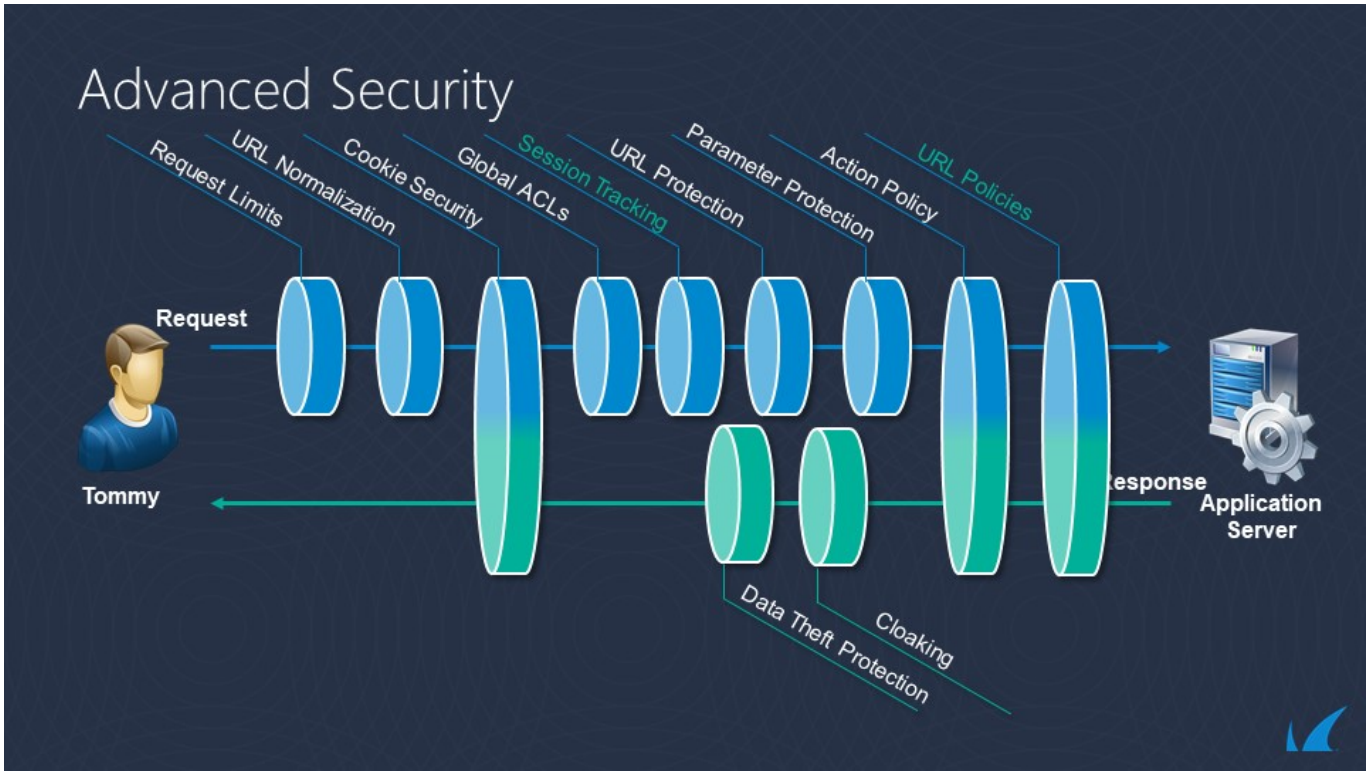
https://campus.barracuda.com/doc/9012050/

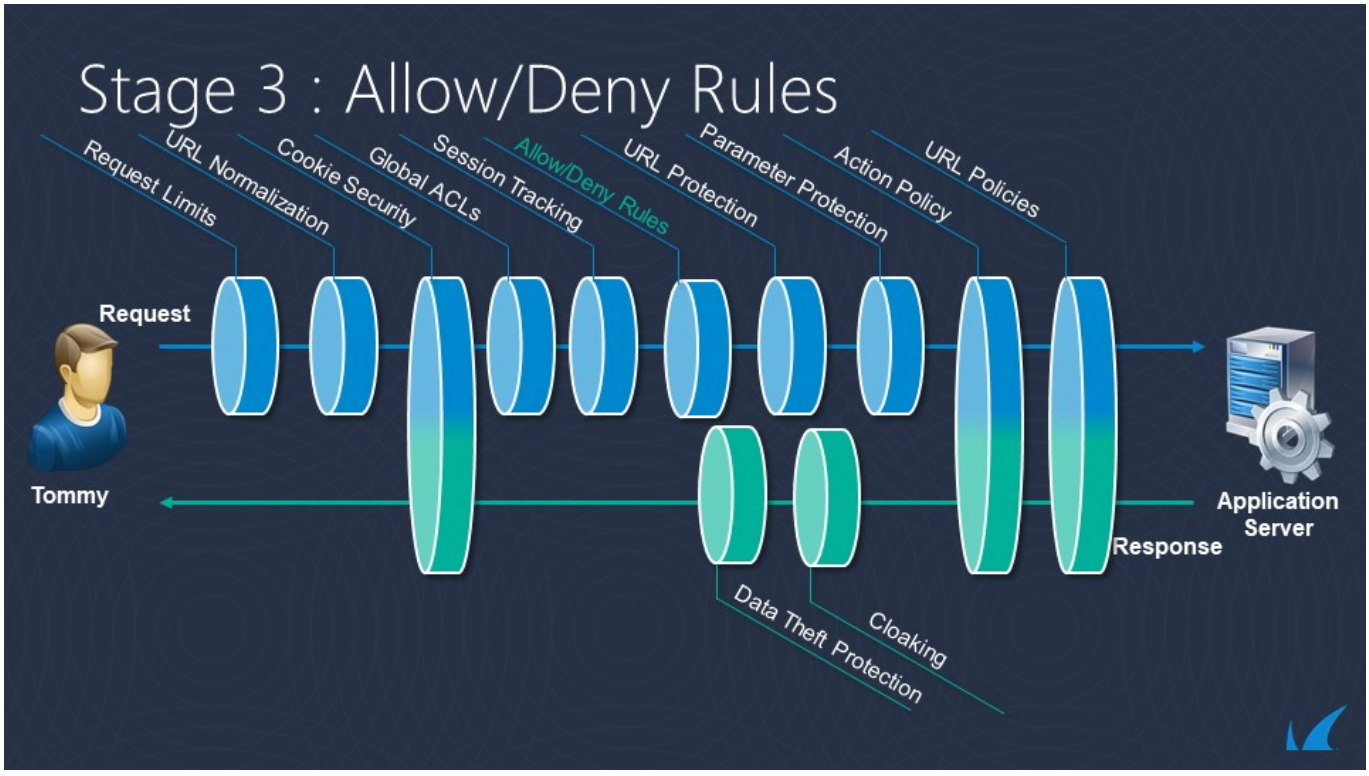
Evaluation Policies

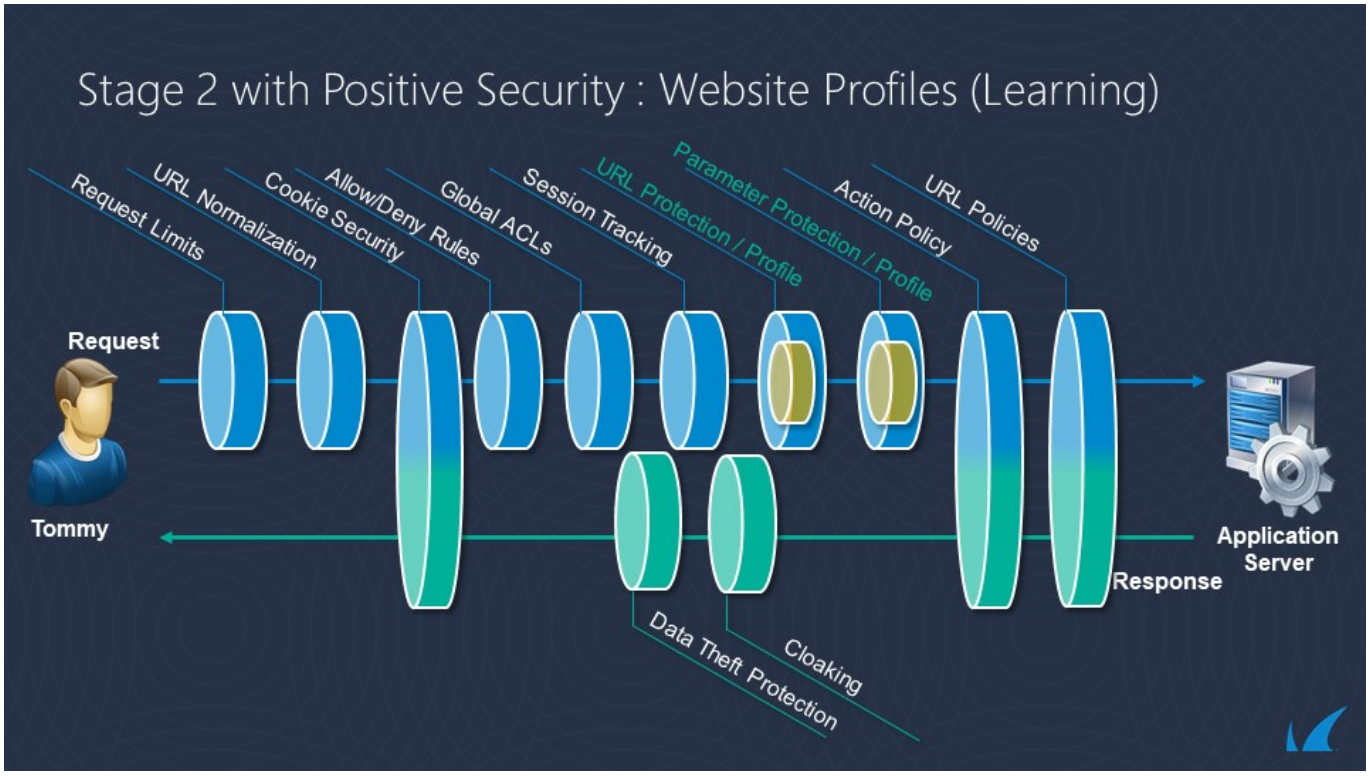
The Barracuda Web Application Firewall applies policies to evaluate requests and responses.

The complete evaluation flow for requests and responses is shown in the images below. A more detailed flow chart to show the decision tree is available under the image set.




Advanced Security
Request Limits
URL Normalization
Cookie Security
Global ACLs
Session Tracking
URL Protection
Parameter Protection
Action Policy
URL Policies
Request
Tommy
Application Server
Data Theft Protection
Cloaking


Stage 3 : Allow/Deny Rules
Request Limits
URL Normalization
Cookie Security
Global ACLs
Session Tracking
Allow/Deny Rules
URL Protection
Parameter Protection
Action Policy
URL Policies
Request
Tommy
Application Server
Response
Data Theft Protection
Cloaking



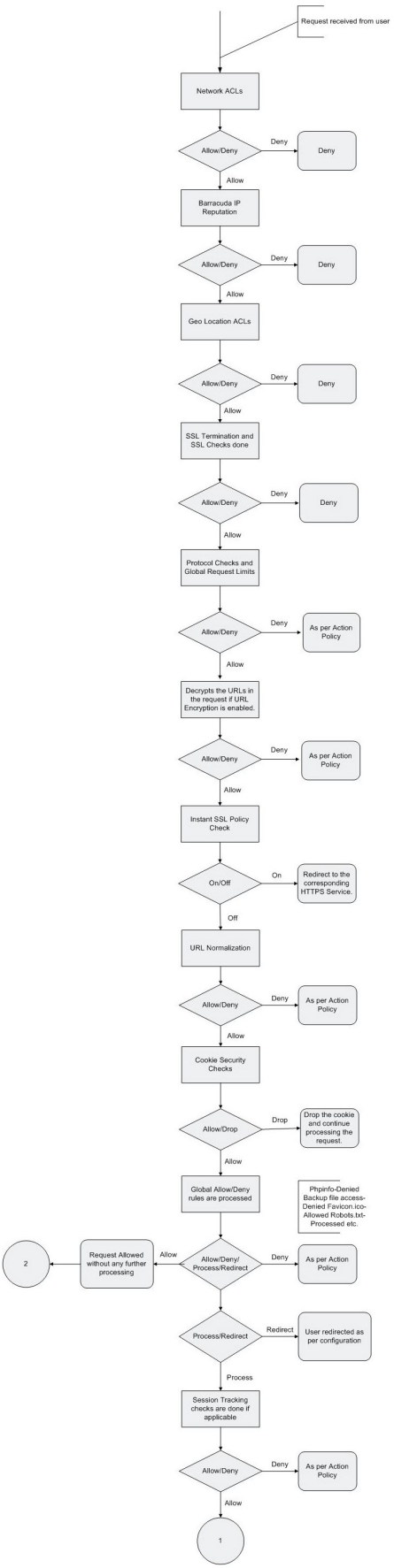
Request Policy Order

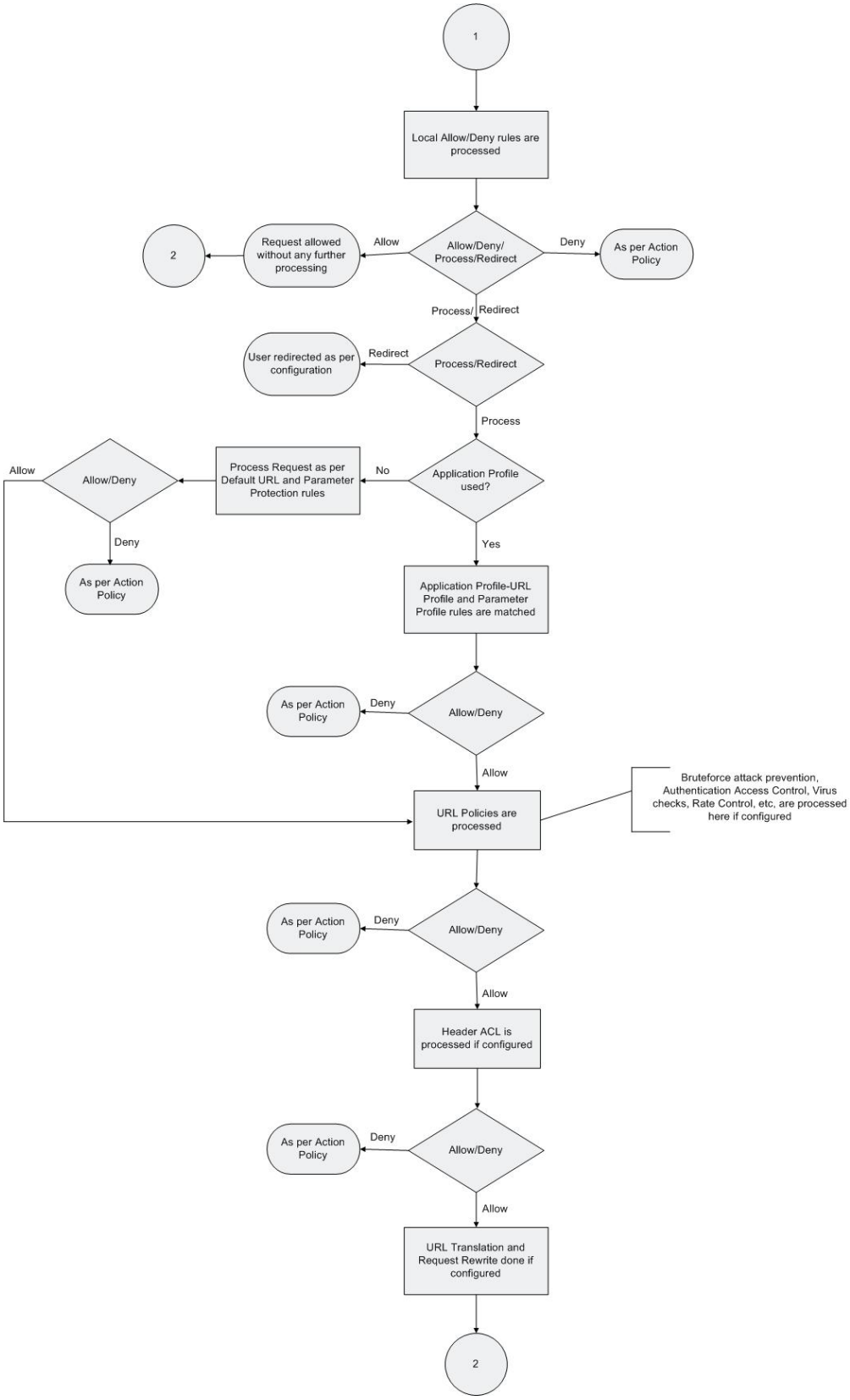
Policies are applied to web application in the following order:

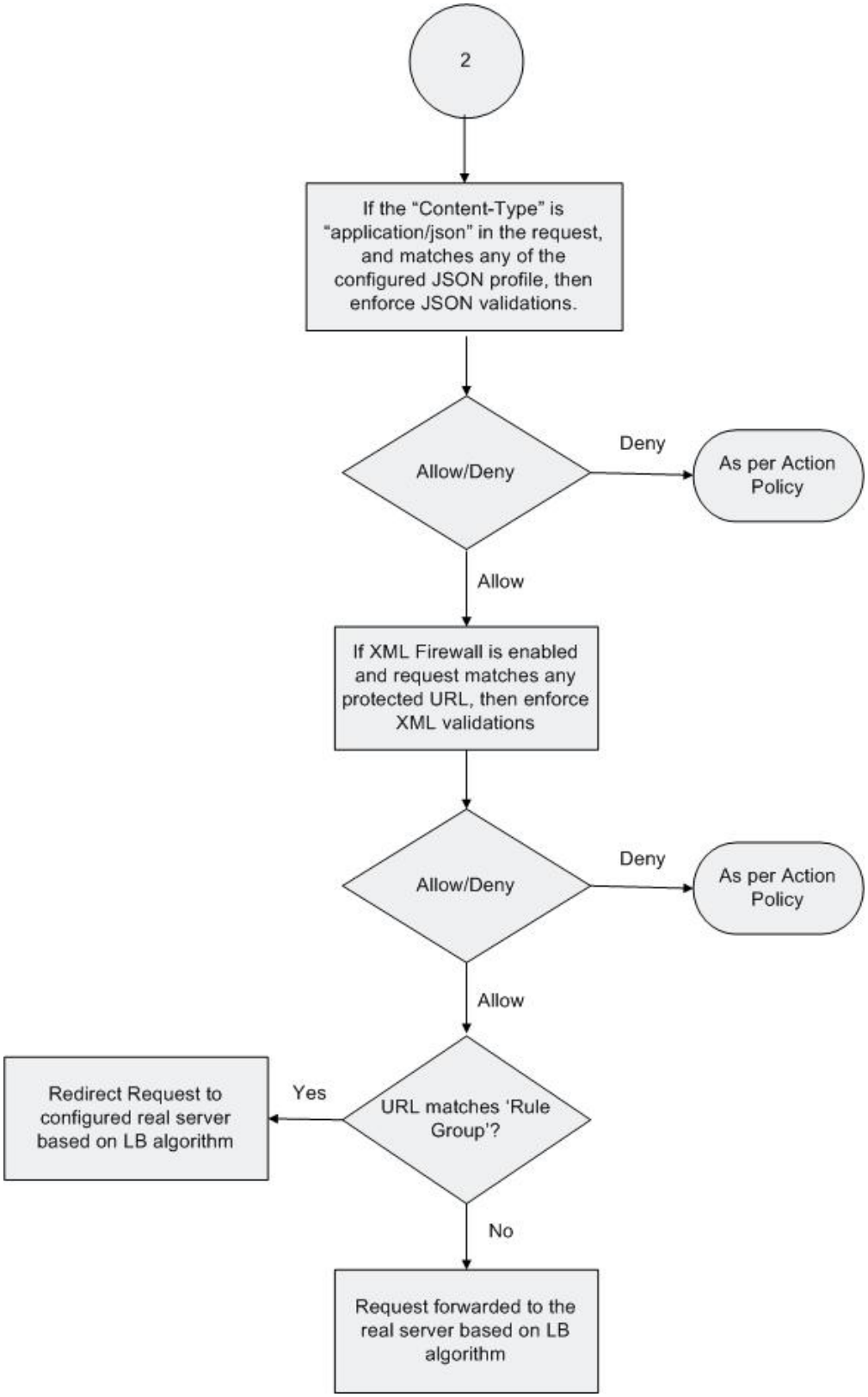
1. Network ACLs [P1]
2. Barracuda IP Reputation [P2]
3. Geo Location ACLs [P3]
4. SSL Termination and SSL Checks [P4]
5. Protocol Checks and Global Request Limits [P5]
6. URL Encryption Rule [P6]
7. Instant SSL redirect policy [P7]
8. URL normalization [P8]
9. Rule match [P9]
10. Cookie security [P10]
11. Global Allow Deny Rules [P11]
12. Session Tracking [P12]
13. Local Allow Deny Rules [P13]
14. Process Profile or Default URL protection and Parameter protection [P14]
15. Advanced Security [P15]
16. Authentication and Access Control [P16]
17. Caching [P17]
18. Web address translation (WAT) [P18]
19. JSON Security [P19]
20. XML Firewall [P20]

21. Load Balancing [P21]

The following flowchart explains the request policy order:


Request received from user
Network ACLs
Allow/Deny
Deny
Deny
Allow
Barracuda IP Reputation
Allow/Deny
Deny
Deny
Allow
Geo Location ACLs
Allow/Deny
Deny
Deny
Allow
SSL Termination and SSL Checks done
Allow/Deny
Deny
Deny
Allow
Protocol Checks and Global Request Limits
Allow/Deny
Deny
As per Action Policy
Allow
Decrypts the URLs in the request if URL Encryption is enabled.
Allow/Deny
Deny
As per Action Policy
Allow
Instant SSL Policy Check
On/Off
On
Redirect to the corresponding HTTPS Service.
Off
URL Normalization
Allow/Deny
Deny
As per Action Policy
Allow
Cookie Security Checks
Allow/Drop
Drop
Drop the cookie and continue processing the request.
Allow
Global Allow/Deny rules are processed
Phpinfo-Denied Backup file access-Denied Favicon.ico-Allowed Robots.txt-Processed etc.
2
Request Allowed without any further processing
Allow
Allow/Deny/ Process/Redirect
Deny
As per Action Policy
Process/Redirect
Redirect
User redirected as per configuration
Process
Session Tracking checks are done if applicable
Allow/Deny
Deny
As per Action Policy
Allow
1


1
Local Allow/Deny rules are processed
Allow/Deny/ Process/Redirect
Allow
Deny
Request allowed without any further processing
2
As per Action Policy
Process/
Redirect
Process/Redirect
Redirect
User redirected as per configuration
Process
Application Profile used?
No
Process Request as per Default URL and Parameter Protection rules
Allow/Deny
Allow
Deny
As per Action Policy
Yes
Application Profile-URL Profile and Parameter Profile rules are matched
Allow/Deny
Deny
As per Action Policy
Allow
URL Policies are processed
Bruteforce attack prevention, Authentication Access Control, Virus checks, Rate Control, etc, are processed here if configured
Allow/Deny
Deny
As per Action Policy
Allow
Header ACL is processed if configured
Allow/Deny
Deny
As per Action Policy
Allow
URL Translation and Request Rewrite done if configured
2

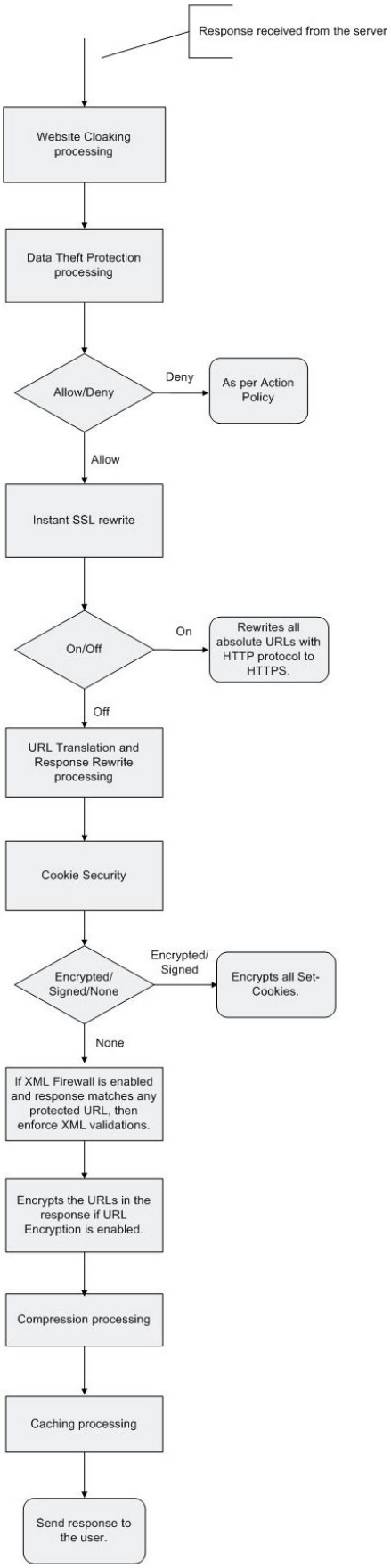

2
If the "Content-Type" is "application/json" in the request, and matches any of the configured JSON profile, then enforce JSON validations.
Allow/Deny
Deny
As per Action Policy
Allow
If XML Firewall is enabled and request matches any protected URL, then enforce XML validations
Allow/Deny
Deny
As per Action Policy
Allow
URL matches 'Rule Group'?
Yes
Redirect Request to configured real server based on LB algorithm
No
Request forwarded to the real server based on LB algorithm

Response Policy Order

Policies are applied to web application responses in the following order:

1. Cloaking policy [P21]
2. Data Theft Protection policy [P22]
3. Instant SSL rewrite [P23]
4. Web address translation (WAT) [P24]
5. Cookie Security [P25]
6. XML Firewall [P26]
7. URL Encryption policy [P27]
8. Compression policy [P28]
9. Caching [P29]

The following flowchart explains the response policy order:


Response received from the server
Website Cloaking processing
Data Theft Protection processing
Allow/Deny
Deny
As per Action Policy
Allow
Instant SSL rewrite
On/Off
On
Rewrites all absolute URLs with HTTP protocol to HTTPS.
Off
URL Translation and Response Rewrite processing
Cookie Security
Encrypted/ Signed/None
Encrypted/ Signed
Encrypts all Set-Cookies.
None
If XML Firewall is enabled and response matches any protected URL, then enforce XML validations.
Encrypts the URLs in the response if URL Encryption is enabled.
Compression processing
Caching processing
Send response to the user.

Execution Flow

The following sections describe how the Barracuda Web Application Firewall processes and evaluates web application requests and responses. (Policies are referenced by the associated number from the preceding lists P1 through P22.)

HTTP Request

The Barracuda Web Application Firewall applies the following policies, listed in order, to a request before forwarding it to the back-end server:

1. When a request is received, the Barracuda Web Application Firewall first performs Network Layer checks such as Network ACLs [P1], Barracuda Block list [P2] (i.e. checks if the IP address has been identified as a potential spam, malware or bot originator.), then the Geo Location ACLs [P3].
2. If the request passes network layer checks, the Barracuda Web Application Firewall then performs SSL checks [P4], and then limit checks [P5] on various components of the HTTP request including URL length, header length, number of headers, and request length. A request exceeding any of these limits is dropped. These checks are performed as request headers are received via one or more TCP packets at the application layer.
3. If URL Encryption is enabled for the service, the Barracuda Web Application Firewall decrypts the URLs in the request [P6].
4. If Instant SSL redirect policy is enabled for the application [P7], the request is redirected to the corresponding HTTPS application, with no further policies applied to it.
5. After all headers are received, the URL and domain of the request are normalized [P8] into a temporary local buffer (see Configuring URL Normalization). The normalized URL and domain strings are used by all remaining policies.
6. Next, the best matching rule group [P9] is found by comparison to the entire request header.
7. Encrypted cookies are then decrypted by applying cookie security policies [P10] (see How to Secure HTTP Cookies). At this point, any cookies inserted by modules such as authentication are also decrypted. If cookie decryption fails, the cookie is removed from the request. This policy never denies a request.
8. Global allow-deny rules [P11] are now enforced, taking the corresponding action of any matching rule (allow, deny with log, deny without log or redirect). If session tracking [P12] is enabled, the session is updated. Then, the local allow-deny rules [P13], if any, are enforced.
9. If profiles [P14] are enabled, perform profile validations using any matching profile, and continue learning [P14] if configured in the profile and enabled for this request. If profiles are not enabled or no profile matches, perform default URL protection [P14] and parameter protection [P14].
10. If a URL policy [P15] matches the request, perform the configured validations including virus scan, data theft protection, brute force prevention, and/or rate control.
11. If access control is enabled for the request [P16] and it has no authentication cookie, the request is dropped. If access is allowed for the user according to the authentication cookie, the

request proceeds to the next policy. A request goes through the authentication process if the request is for the authentication landing or login page. If authentication is successful, the Barracuda Web Application Firewall inserts an authorization cookie. None of the remaining policies apply to authorization requests.

12. If caching policy [P17] is enabled (see Configuring Caching and Compression) and if the request can be retrieved from the cache, the Barracuda Web Application Firewall serves the object from its cache store. If the object is found in the cache, none of the other policies apply to the request. No response policies are applied to a response from the cache. If the object is not found in cache, the request proceeds to the next step.
13. WAT policies [P18] are applied to the request (see Content Rewriting, Content Based Rules) including request rewrite and URL translation policies. Some portions of the request are rewritten due to this policy. Rewriting might also redirect the request, in which case none of the remaining policies are applied to the request.
14. If the request Content-Type is **application/json**, and the request matches any of the configured JSON profiles associated with the service, then JSON validations are enforced.[P19]
15. If XML Firewall [P20] is enabled (see Web Services and XML Firewall Protection), the request Content-Type is **XML**, and any matching protected URL is configured, then the enabled XML protection validations are enforced. The validations can include XML document checks, WS-I basic profile assertions and SOAP validations.
16. Load balancing policy [P21] (see Configuring Load Balancing for a Service) directs the request to the appropriate back-end server. The Barracuda Web Application Firewall may add a connection header at this step.

HTTP Response

The following policies are applied to the back-end server response:

1. If Cloaking policy [P21] is enabled, HTTP headers and return codes are masked before sending the response to the client.
2. If Data Theft Protection policy [P22] is enabled, sensitive data elements are masked before sending the response to the client.
3. If the Instant SSL rewrite policy is enabled [P23], some of the absolute URIs (hyperlinks with domain fields) found by the parsing engine might be rewritten to use the HTTPS protocol, depending on the rewrite domain list in the Instant SSL policy.
4. WAT policies are applied [P24]:
    1. Response rewrite policy rewrites the response headers (see Content Rewriting). The policy can add or delete a header conditionally based on other response headers and request URL and domain fields.
    2. URL translation policy rewrites the content (see Content Based Rules). If any hyperlink reference in the HTML content recognized by the HTML parsing engine matches a URL translation “inside rule,” the link is rewritten by applying the corresponding “outside rule.” If a page is translated, the response is either encoded using HTTP/1.1 Transfer Chunk Encoding scheme, or the underlying TCP connection is closed if the front end used the HTTP/1.0 protocol.
5. If Cookie Security (Tamper Proof Mode) [P25] is set to *Encrypted/Signed*, the Barracuda Web Application Firewall encrypts the set cookie in the response.

6. If XML Firewall [P26] is enabled and the response matches any protected URL, then XML validations are enforced.
7. If URL Encryption [P27] is enabled for the service, the Barracuda Web Application Firewall encrypts the URLs before sending the response to the client.
8. If the Compression policy [P28] is *On* for the service/URL, the response page is compressed (see Configuring Caching and Compression).
9. If the Caching policy [P29] is *On* for the request URL, and if the response headers indicate the object can be cached, a copy of the page is stored in the cache.

Local Allow/Deny Rules

URL ACL rules (step 7 in "HTTP Request") are applied in the following order. If the ACL mode is set to *Active*, request processing is terminated when a violation of any policy in the ACL is detected. If the ACL mode is set to *Passive*, the violation is logged and request processing continues. (The matching algorithm for URL ACLs and rule groups is the same.)

1. URL ACLs: The incoming request is compared to the list of URL ACLs to find the best match. If no match is found or the action parameter is set to *deny*, the request is dropped. The request is also compared to the limits and normalization policies, and dropped for any violation of these policies. The comparison is done with the < *domain*, *URL,* and *header* > values, in that order of precedence.
2. Header ACLs: Each of the headers in the request is compared to the list of header ACLs. If a match is found, the header value is validated by checking for denied metacharacters, denied keywords, and for valid length (compared to configured maximum length) for violations.

URL Policies

URL policies (step 8 in "HTTP Request") are applied in the following order.

1. Virus Scan: The incoming client requests are scanned for viruses. Requests containing virus signatures are denied.
2. Rate Control: The Rate Control Pool is defined to limit client requests to the Barracuda Web Application Firewall. You can add a rate control pool and set a request maximum for that pool. Rate Controls protect applications from being pushed beyond their performance limits, preventing application layer Denial of Service (DoS) attacks.
3. Bruteforce Prevention: Bruteforce prevention protects web applications and websites from bruteforce attacks, which try every possible code, combination, or password to find the right one.

Rule Matching

A rule consists of three patterns: host, URL, and extended match rules. The policies of the "best matching" rule are applied to a request. Host and URL rules are used to match Host and URL fields, respectively. Extended match rules can be compared to any combination of HTTP headers and/or

query string parameters in a request. A “*” rule (to be read as a rule consisting of URL “*”) matches any value for that parameter.

The best matching rule is the one with the longest matching host and URL and is the first matching rule in hierarchical order. If more than one extended match rule is configured with the same host and URL keys, the extended match rules are searched based on extended match sequence.

Rule Match Algorithm

Host and URL rules (used for both URL Policies and Rule Groups) are treated as <prefix, suffix> pairs by the rule-match engine:

- A Prefix rule key is the part of the rule preceding the asterisk (*). The asterisk is treated as a wildcard, meaning any value.
- A Suffix rule key is the part of the rule following the asterisk.

If a rule does not have an asterisk, its suffix rule key is NULL.

The following algorithm is used by the Rule Match engine:

1. Find best matching Host Prefix Rule Key. The best match is defined as the longest rule matching the HTTP request Host header, left to right. The number of characters matched is the length of the Prefix Rule Key. If no Prefix Rule matches, the Rule Match engine terminates with failure and the request is dropped.
2. Find the best matching Host Suffix Rule Key. Best match is defined as the longest Suffix Rule Key matching the HTTP request Host header right to left. The number of characters matched is the length of Suffix Rule Key. If a matching rule is found, the current < Prefix, Suffix > pair matches the Host Rule Key, so go to Step 3. If no Host suffix Rule matches, discard this Prefix Rule Key and go to Step 1 to find the next matching Host Prefix Rule.
3. Find the best matching URL Prefix Rule Key. If none found, discard this Host Rule Key and go to step 1 to find the next matching Host Rule Key. Best matching URL Prefix Key is defined as the longest URL Prefix Rule matching the HTTP Request-URI header, left to right. The number of characters matched is the length of URL Prefix Rule Key.
4. Find the best matching URL Suffix Rule Key. Best match is defined as the longest Suffix Rule Key matching the HTTP Request-URI header, right to left. If found, the current < Prefix, Suffix > pair is the matching URL Rule Key. We found the best matching Host Rule Key and URL Rule Key and continue at Step 5. If no match is found, discard this Prefix URL Rule and return to Step 3 to find the next matching URL Prefix Rule.
5. Find the first matching Extended Match Rule. The Extended Match rules are sorted based on the extended match sequence. If a matching rule is found, the Rule match engine terminates with success and the HTTP Request follows the policies defined on this rule. If no match is found, discard this Extended Match Rule and go back to Step 4 to find the next matching URL Suffix Rule Key.

A successful search terminates at step 5. An unsuccessful search terminates at step 1.

Hierarchical Match

The hierarchical match first compares the host header in the request to all host matches configured, and takes the best match. The ACL list is reduced to this set. Next, the URL path in the request is compared to all URLs in the reduced set of ACLs. This is again a best match. If multiple ACLs match, then each extended match rule is evaluated in ascending order of extended match sequence. The first extended match rule that matches will be the ACL match.

For example, consider the requests given below and see how it is matched with the ACLs in the following table:

ACLs	Host Match	URL Match	Extended Match	Extended Match Sequence
1	www.abc.com	/sales1/*	Header User-Agent co IE5.0 1	1
2	www.abc.com	/sales1/*	Header User-Agent co Mozilla 2	2
3	www.abc.com	/sales1/*	*	3
4	www.abc.com	/sales2/*	Header User-Agent co wget 0	0
5	*.abc.com	/sales2/*	*	0
6	*.abc.com	/sales3/*	*	0
7	*	/sales1/*	*	0
8	*	*	*	0

1. http://www.abc.com/sales1/index.html, from IE5.0
    1. Host header is www.abc.com, and therefore ACLs 1 to 4 match this request. Also ACLs 4 to 8 match, but 1 to 4 are better matches, since they are more specific.
    2. URL Path is /sales1/index.html, and therefore ACLs 1 to 3 match.
    3. Evaluate extended match rule for ACL 1 first, since extended match sequence = 1.
    4. As a result, ACL 1 is matched.
2. http://www.abc.com/sales2/index.html, from IE5.0
    1. Host header is www.abc.com, and therefore ACLs 1 to 4 match. Also ACLs 4 to 8 match, but this is the best match, since it is most specific.
    2. URL Path is /sales2/index.html, and therefore only ACL 4 matches.
    3. Evaluate extended match rule for ACL 4. ACL 4 does not match, since User-Agent is expected to be wget, which is not true.
    4. Thus, no matches in ACLs 1 to 4.
    5. The next best matching ACLs on the host header are ACLs 5 to 6.
    6. URL Path matches only ACL 5.
    7. Evaluate extended match rule for ACL 5. * matches anything.
    8. As a result, ACL 5 is matched.

3. http://www.abc.com/sales3/index.html
    1. Host header is www.abc.com, and therefore ACLs 1 to 4 match. Also ACLs 4 to 8 match, but this is the best match, since it is most specific.
    2. URL Path is /sales3/index.html, which does not match any of the URLs in ACLs 1 to 4.
    3. The next best matching ACLs in the host header are ACLs 5 to 6.
    4. URL path matches only ACL 6.
    5. Header rule matches ACL 6.
    6. As a result, ACL 6 is matched.
4. http://mirror.abc.com/sales4/index.html
    1. Host header is mirror.abc.com, and therefore ACLs 5 and 6 match.
    2. URL Path does not match any of the URLs in ACLs 5 and 6.
    3. The next best matching ACLs in the host header are ACLs 7 and 8.
    4. URL path matches only ACL 8.
    5. Header rule matches ACL 8.
    6. As a result, ACL 8 is matched.

Sequential Match

Sequential match completely ignores the host header and URL path. Each extended match rule is evaluated in sequential order based on the extended match sequence. The first rule that matches is the ACL match.

To explain how the rule-match engine selects the best match, consider the following Rule Match table:

ACLs	Host Match	URL Match	Extended Match Rule	Extended Match Sequence
1	*	*	Header Host eq www.abc.com && Header User-Agent co IE5.0 && URI req /sales1/*	1
2	*	*	Header Host eq www.abc.com && Header User-Agent co Mozilla && URI req /sales1/*	2
3	*	*	* * Header Host eq www.abc.com && URI req /sales1/*	3
4	*	*	* * Header Host eq www.abc.com && Header User-Agent co wget && URI req /sales2/*	4
5	*	*	* * Header Host req .abc.com && URI req /sales2/*	5
6	*	*	* * Header Host req *.abc.com && URI req /sales3/*	6
7	*	*	* URI req /sales1/*	7
8	*	*	*	8

1. sales2http://www.abc.com/sales1/index.html, from IE5.0
    1. ACLs 1 to 8 match host and URL keys.
    2. Evaluate extended match rule for ACL 1 first, since extended match sequence = 1.
    3. As a result, ACL 1 is matched.
2. http://www.abc.com/sales2/index.html, from IE5.0
    1. ACLs 1 to 8 match host and URL keys.
    2. Evaluate extended match rule for ACL 1 to 8 in order.
    3. Extended match rule for ACL 5 matches.
    4. As a result, ACL 5 is matched.
3. http://www.abc.com/sales3/index.html
    1. ACLs 1 to 8 match host and URL keys.
    2. Evaluate header rules for ACL 1 to 8 in order.
    3. Header rule for ACL 6 matches.
    4. As a result, ACL 6 is matched.
4. http://mirror.abc.com/sales4/index.html
    1. ACLs 1 to 8 match host and URL keys.
    2. Evaluate header rules for ACL 1 to 8 in order.
    3. Header rule matches ACL 8.
    4. As a result, ACL 8 is matched.

Figures

1. 01 -Stage 1 - Security Policies.jpg
2. 02 - Stage 2 Advanced Security.jpg
3. 03 - Stage 3 Allow Deny Rules.jpg
4. 04 - Stage 2 with Positive Security.jpg
5. FlowChart1_May_27_2014.jpg
6. FlowChart_3.jpg
7. FlowChart_4_April_7_2015.jpg
8. Response_Flow_Updated.jpg