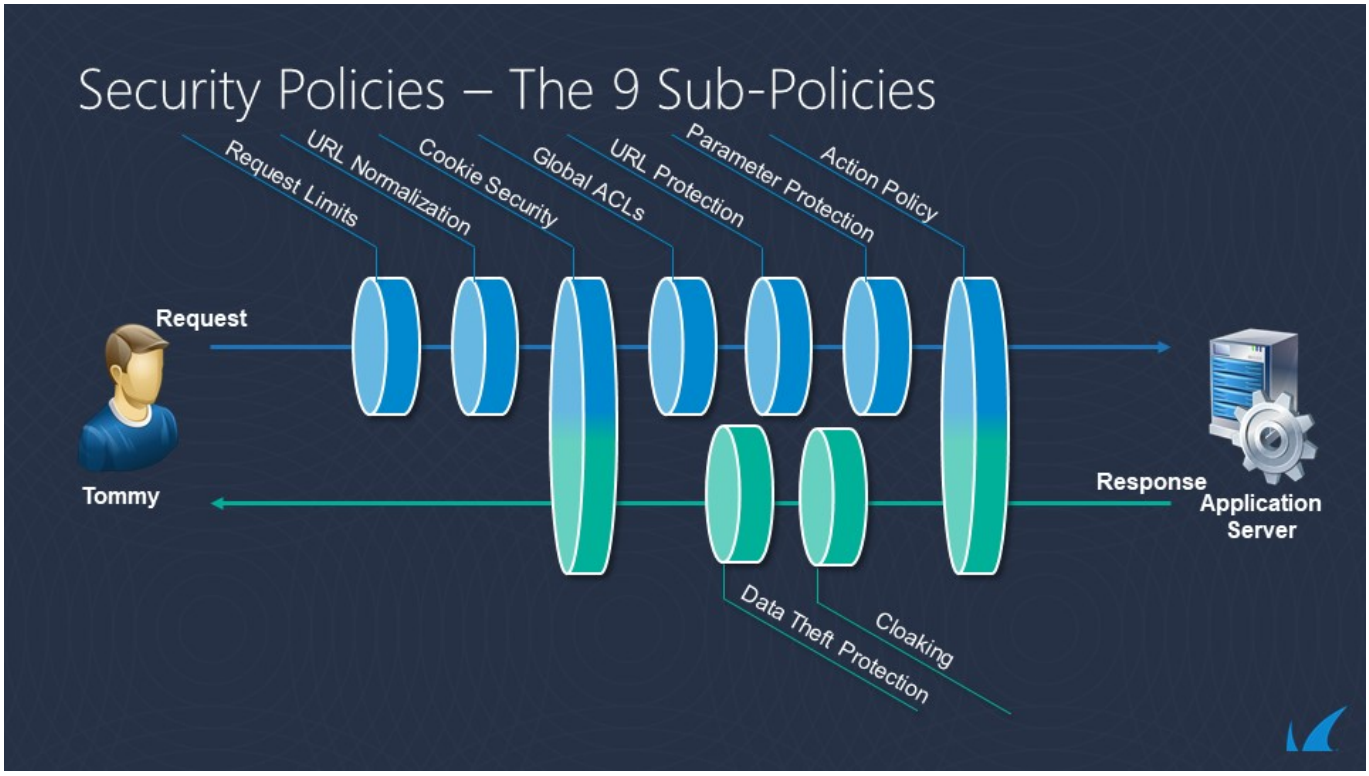
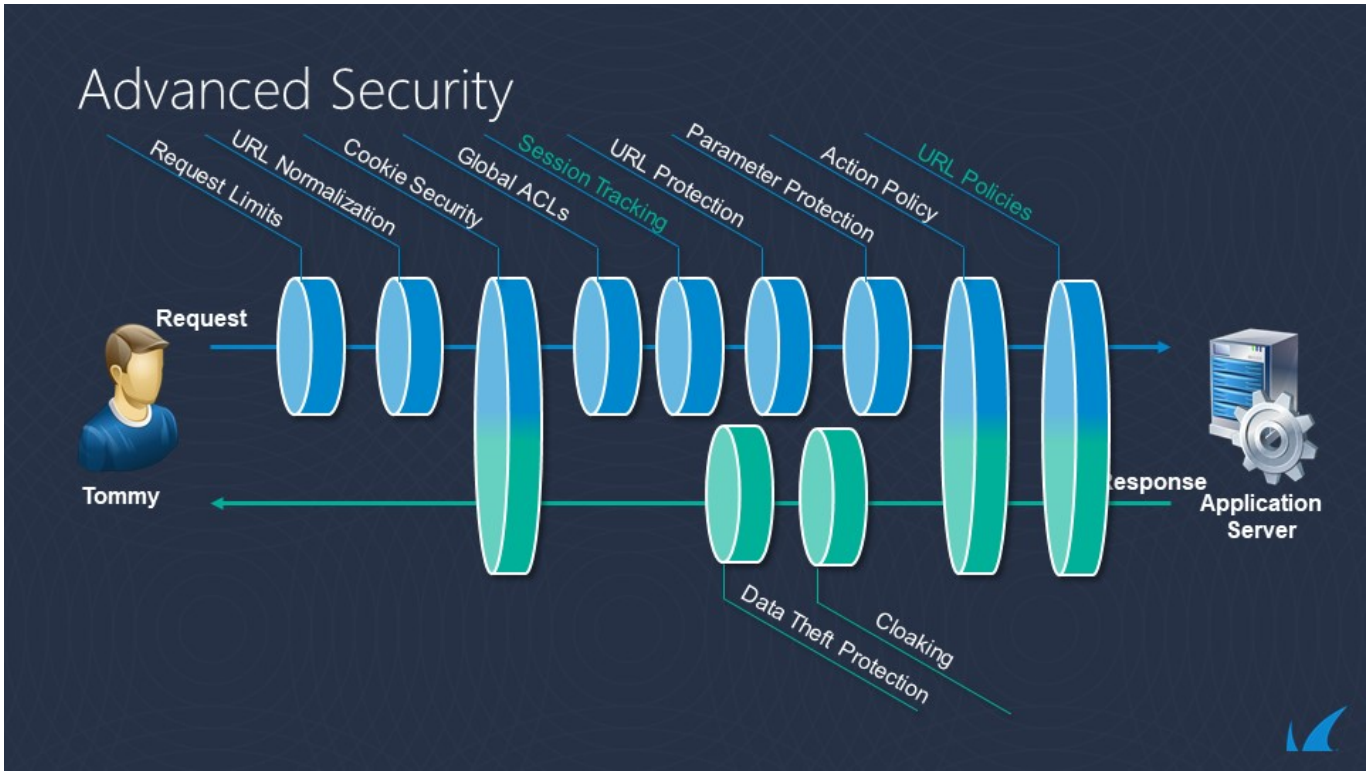
Evaluation Policy and Flow

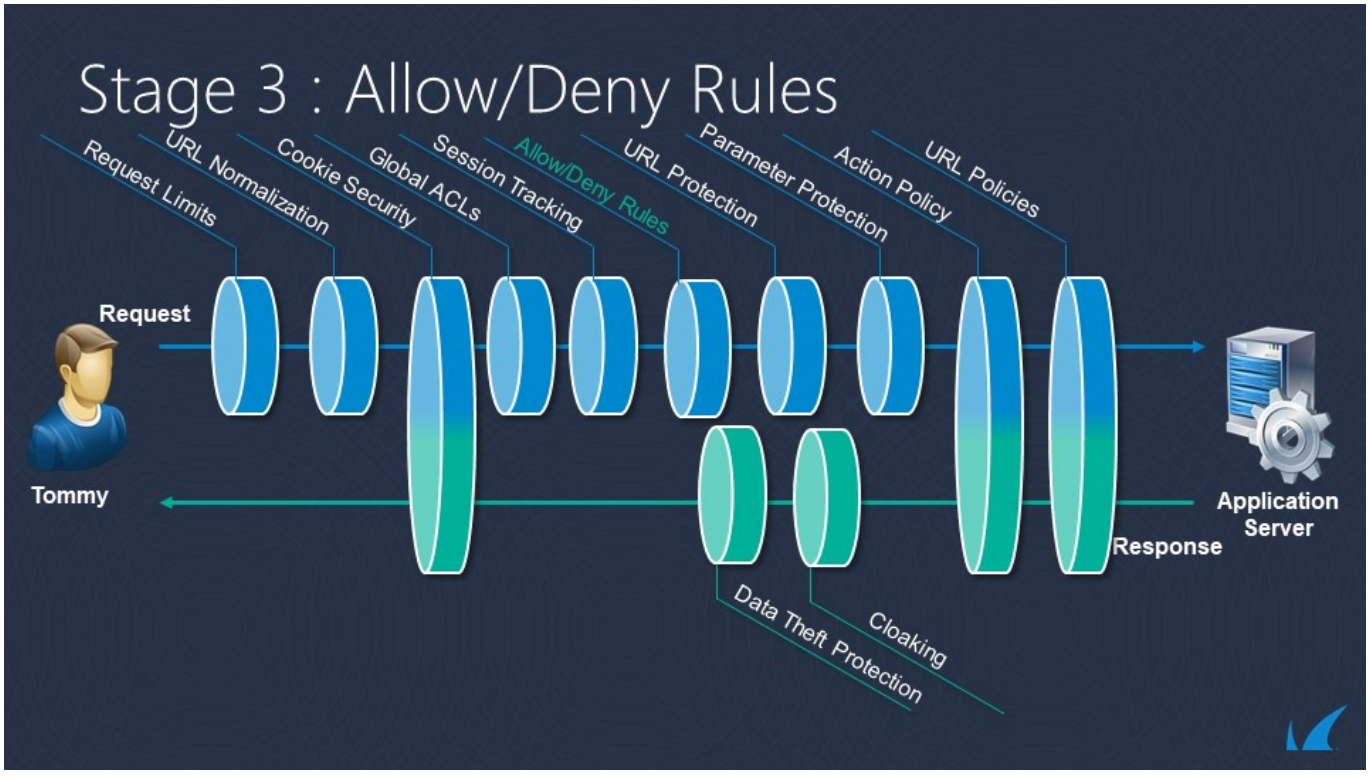
https://campus.barracuda.com/doc/9012050/

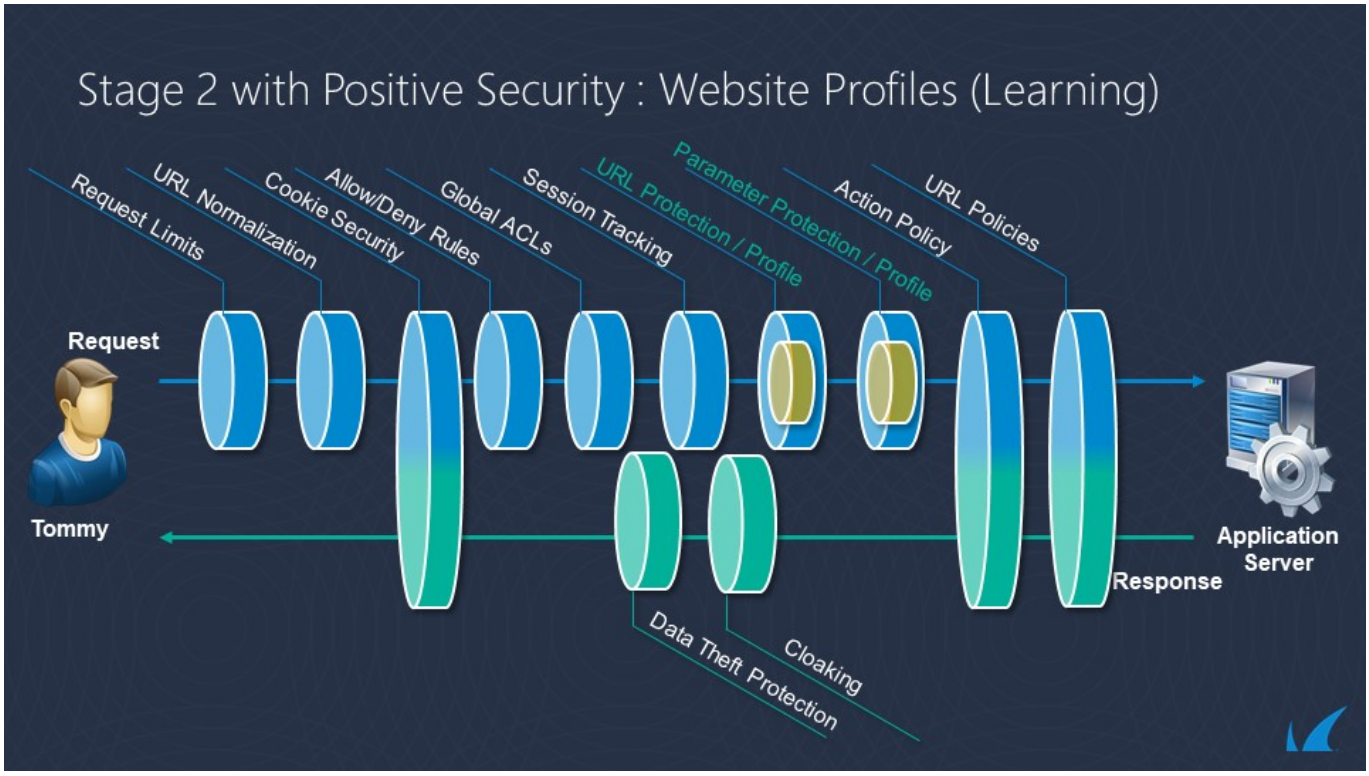
The Barracuda Web Application Firewall applies policies to evaluate requests and responses.

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



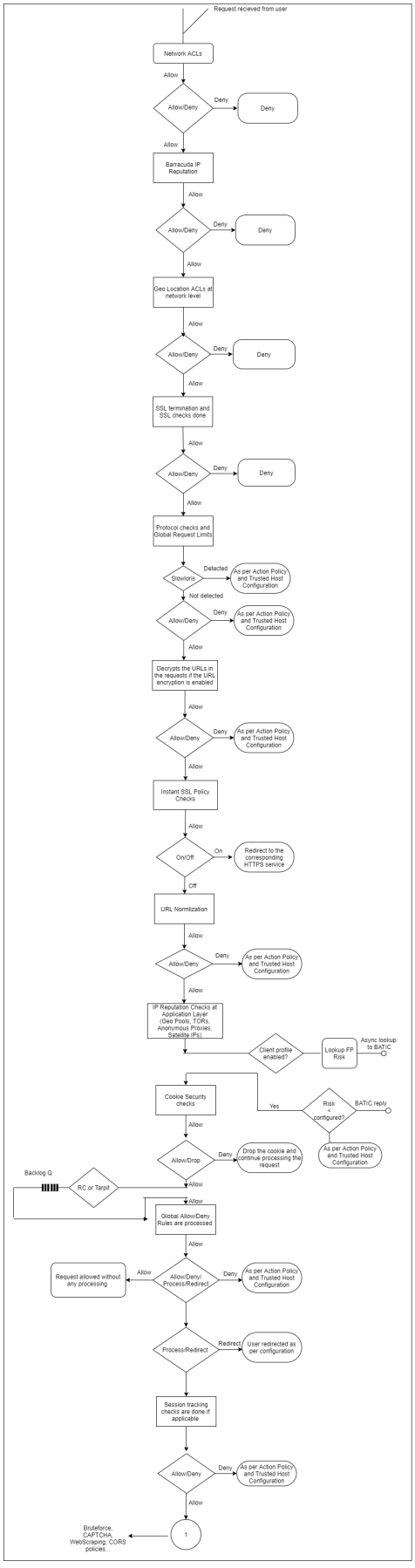
Request Policy Order

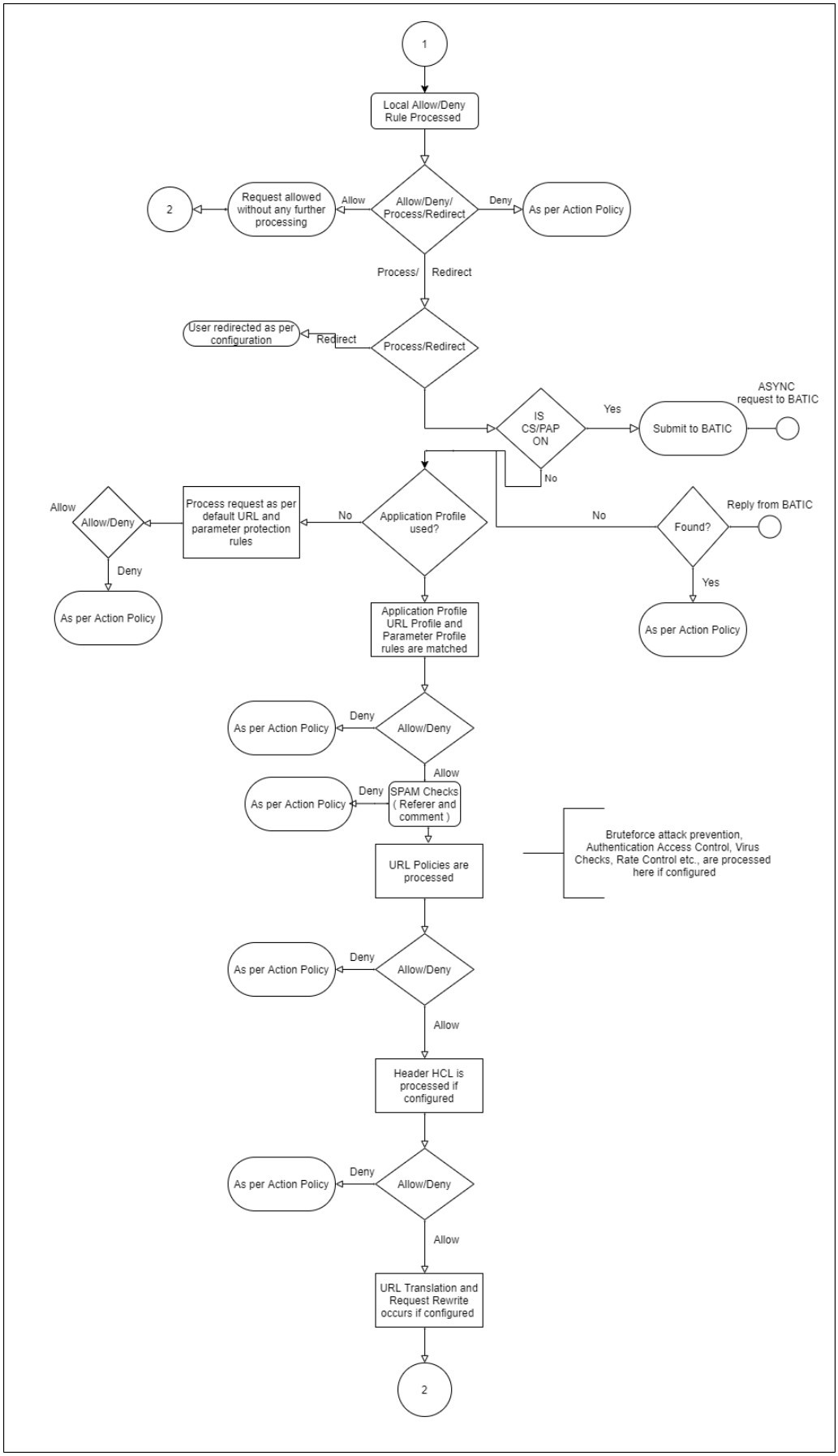
Policies are applied to web application in the following order:

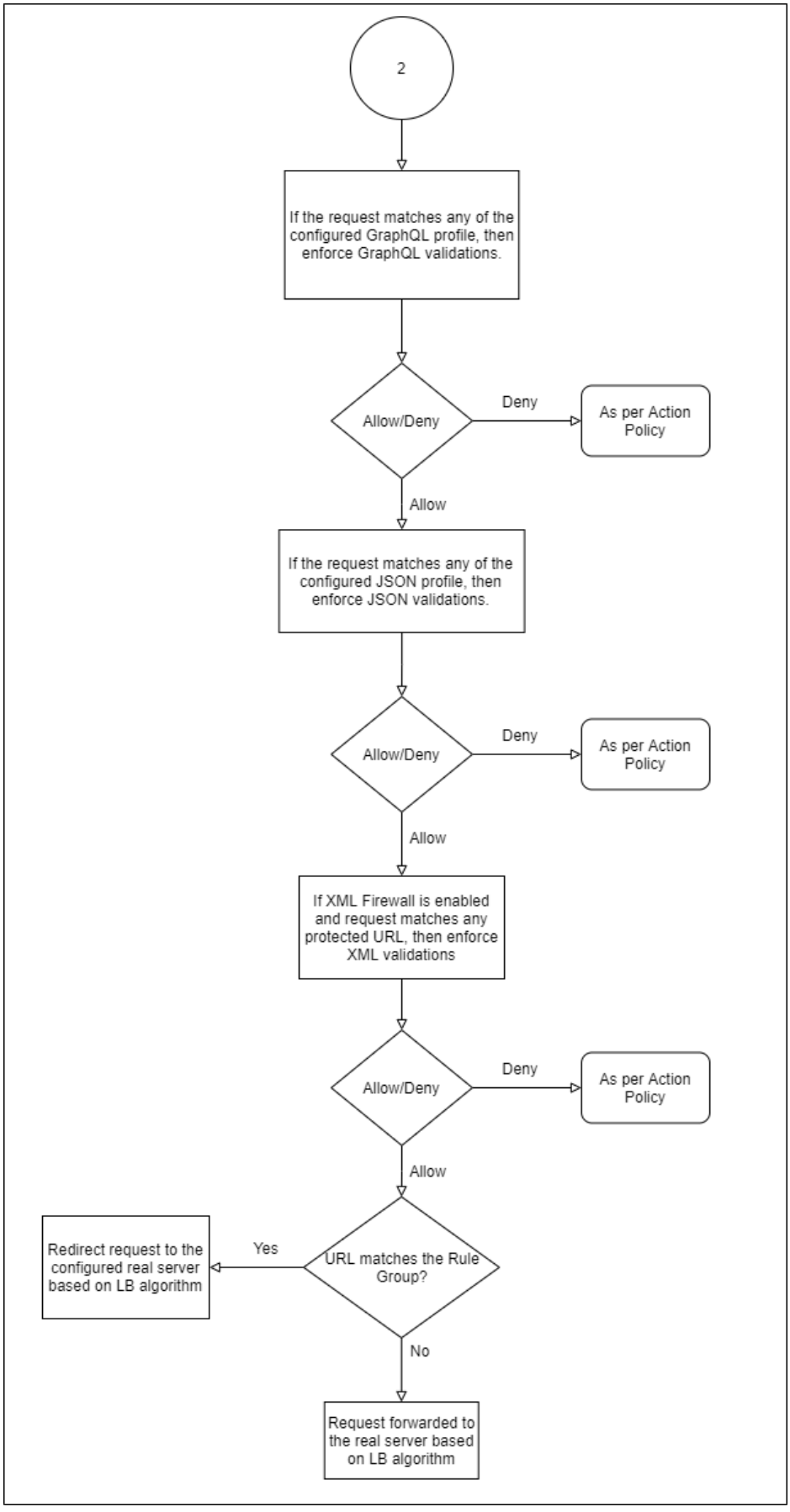
1. Network ACLs [P1]
2. Barracuda IP Reputation [P2]
3. Geo Location ACLs [P3]
4. SSL Termination and SSL Checks [P4]
5. Protocol Checks and Global Request Limits [P5]
6. URL Encryption Rule [P6]
7. Instant SSL redirect policy [P7]
8. URL normalization [P8]
9. Rule match [P9]
10. IP Reputation Checks at the Application Layer [P10]
11. Cookie security [P11]
12. Tarpit and rate controlling [P12]
13. Global Allow Deny Rules [P13]
14. Session Tracking [P14]
15. Bruteforce, CAPTCHA/reCAPTCHA, Web Scraping and CORS [P15]
16. Local Allow Deny Rules [P16]
17. Credential Stuffing/Credential Spraying and Privileged Account Protection [P17]
18. Process Profile or Default URL protection and Parameter protection [P18]
19. Comment and Referrer header spam [P19]
20. Advanced Security [P20]

21. Authentication and Access Control [P21]
22. Caching [P22]
23. Web address translation (WAT) [P23]
24. GraphQL Security [P24]
25. JSON Security [P25]
26. XML Firewall [P26]
27. Load Balancing [P27]

The following flowchart explains the request policy order:


Request recieved from user
Network ACLs
Allow
Allow/Deny
Deny
Deny
Allow
Barracuda IP Reputation
Allow
Allow/Deny
Deny
Deny
Allow
Geo Location ACLs at network level
Allow
Allow/Deny
Deny
Deny
Allow
SSL termination and SSL checks done
Allow
Allow/Deny
Deny
Deny
Allow
Protocol checks and Global Request Limits
Slowloris
Detected
As per Action Policy and Trusted Host Configuration
Not detected
Allow/Deny
Deny
As per Action Policy and Trusted Host Configuration
Allow
Decrypts the URLs in the requests if the URL encryption is enabled
Allow
Allow/Deny
Deny
As per Action Policy and Trusted Host Configuration
Allow
Instant SSL Policy Checks
Allow
On/Off
On
Redirect to the corresponding HTTPS service
Off
URL Normlization
Allow
Allow/Deny
Deny
As per Action Policy and Trusted Host Configuration
Allow
IP Reputation Checks at Application Layer (Geo Pools, TORs, Anonymous Proxies, Satellite IPs)
Client profile enabled?
Lookup FP Risk
Async lookup to BATIC
Cookie Security checks
Yes
Risk configured?
BATIC reply
As per Action Policy and Trusted Host Configuration
Allow
Allow/Drop
Deny
Drop the cookie and continue processing the request
Backlog Q
RC or Tarpit
Allow
Allow
Global Allow/Deny Rules are processed
Allow
Request allowed without any processing
Allow
Allow/Deny/ Process/Redirect
Deny
As per Action Policy and Trusted Host Configuration
Process/Redirect
Redirect
User redirected as per configuration
Session tracking checks are done if applicable
Allow/Deny
Deny
As per Action Policy and Trusted Host Configuration
Allow
1
Bruteforce, CAPTCHA, WebScraping, CORS policies...


1
Local Allow/Deny Rule Processed
Allow/Deny/ Process/Redirect
Allow
Request allowed without any further processing
2
Deny
As per Action Policy
Process/
Redirect
Process/Redirect
Redirect
User redirected as per configuration
IS CS/PAP ON
Yes
Submit to BATIC
ASYNC request to BATIC
No
Application Profile used?
No
Found?
Reply from BATIC
Yes
As per Action Policy
No
Process request as per default URL and parameter protection rules
Allow/Deny
Allow
Deny
As per Action Policy
Application Profile URL Profile and Parameter Profile rules are matched
Allow/Deny
Deny
As per Action Policy
Allow
SPAM Checks (Referer and comment)
Deny
As per Action Policy
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 HCL is processed if configured
Allow/Deny
Deny
As per Action Policy
Allow
URL Translation and Request Rewrite occurs if configured
2

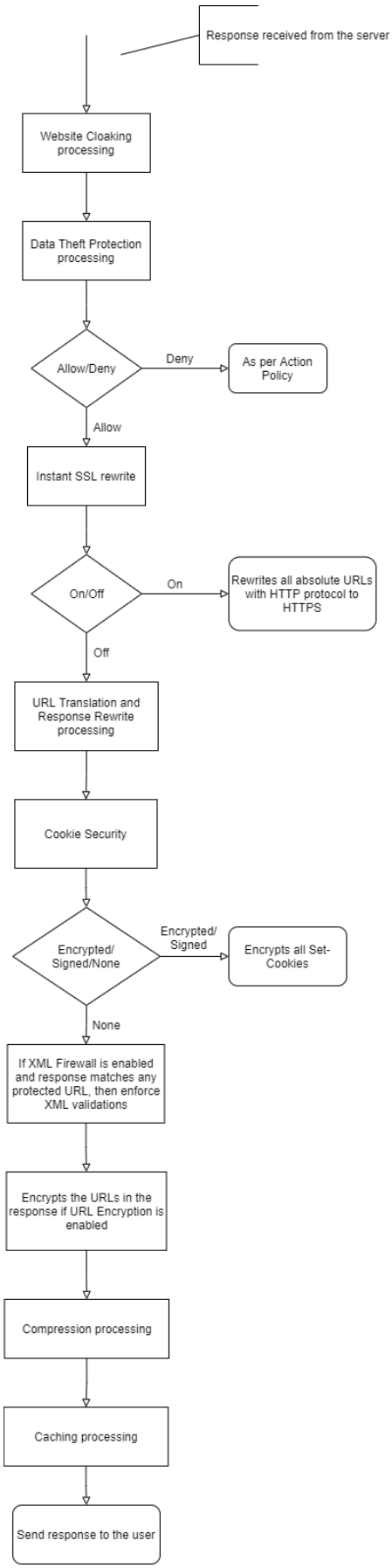

2
If the request matches any of the configured GraphQL profile, then enforce GraphQL validations.
Allow/Deny
Deny
As per Action Policy
Allow
If the request matches any of the configured JSON profile, then enforce JSON validations.
Allow/Deny
Deny
As per Action Policy
Allow
If XML Firewall is enabled and request matches any protected URL, then enforce XML validations
Allow/Deny
Deny
As per Action Policy
Allow
URL matches the Rule Group?
Yes
Redirect request to the 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 [P28]
2. Data Theft Protection policy [P29]
3. Instant SSL rewrite [P30]
4. Web address translation (WAT) [P31]
5. Cookie Security [P32]
6. XML Firewall [P33]
7. URL Encryption policy [P34]
8. Compression policy [P35]
9. Caching [P36]

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 P36.)

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. If IP Reputation and Geo IP rules are enabled, the request is inspected for matches with the IP reputation policies at the application layer [P10] (based on Client IP headers).
8. Encrypted cookies are then decrypted by applying cookie security policies [P11] (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.
9. The request is then checked against the configured tarpit policy and the rate controlling policy to check if any connection limits are exceeding [P12].
10. Global allow-deny rules [P13] are now enforced, taking the corresponding action of any matching rule (allow, deny with log, deny without log, or redirect). If session tracking [P14] is enabled, the session is updated. The request payload is then evaluated for any bruteforce or web scraping violations. If violations are found, the captcha/reCAPTCHA challenge is given per the action policy follow-up action. CORS security headers are inserted in the HTTP responses if CORS Security is enabled [P15]. Then, the local allow-deny rules [P16], if any, are enforced. Account Protection features like Credential Stuffing/Spraying check if credentials used to log into the application are compromised. The client is profiled for login actions to determine any account takeover attempts if the Privileged Account Protection feature is enabled [P17].

11. If profiles [P18] are enabled, perform profile validations using any matching profile, and continue learning [P18] if configured in the profile and enabled for this request. If profiles are not enabled or no profile matches, perform default URL protection [P18] and parameter protection [P18].
12. If "Comment spam" or "Referrer spam" is configured, the payload is checked for any spam content in comment fields or the Referrer header value [P19].
13. If a URL policy [P20] matches the request, perform the configured validations including virus scan, data theft protection, brute force prevention, and/or rate control.
14. If access control is enabled for the request [P21] 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.
15. If caching policy [P22] 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.
16. WAT policies [P23] 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.
17. If the request matches any of the configured GraphQL profiles associated with the service, GraphQL Security validations are enforced [P24]. See GraphQL Security.
18. If the request matches any of the configured JSON profiles associated with the service, JSON Security validations are enforced [P25]. See REST API and JSON Security.
19. If XML Firewall [P26] 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.
20. Load balancing policy [P27] (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 [P28] is enabled, HTTP headers and return codes are masked before sending the response to the client.
2. If Data Theft Protection policy [P29] is enabled, sensitive data elements are masked before sending the response to the client.
3. If the Instant SSL rewrite policy is enabled [P30], 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 [P31]:
    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) [P32] is set to *Encrypted/Signed*, the Barracuda Web Application Firewall encrypts the set cookie in the response.
6. If XML Firewall [P33] is enabled and the response matches any protected URL, then XML validations are enforced.
7. If URL Encryption [P34] 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 [P35] is *On* for the service/URL, the response page is compressed (see Configuring Caching and Compression).
9. If the Caching policy [P36] 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.

Related Articles:

- Security Policies
- Allow/Deny Rules for Headers and URLs

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. EvalFlowdiagram1.png
6. Eval_Flowdiagram2.png
7. Eval_Flowdiagram3.png
8. Eval-Response-flowdiagram.png