

v4

# Penetration Testing Student

## Web Applications

Section 01 | Module 03

```
1 class ScientistResult
2   attr_reader :observations
3   attr_reader :control
4   attr_reader :candidates
5   attr_reader :matched
6   attr_reader :experiment
7   attr_reader :observations
8   attr_reader :control
9   attr_reader :candidates
10  attr_reader :matched
11  attr_reader :experiment
12  attr_reader :observations
13  attr_reader :control
14  attr_reader :candidates
15  attr_reader :matched
16  attr_reader :experiment
17  attr_reader :observations
18  attr_reader :control
19  attr_reader :candidates
20  attr_reader :matched
21  attr_reader :experiment
22  attr_reader :observations
23  attr_reader :control
24  attr_reader :candidates
25  attr_reader :matched
26  attr_reader :experiment
27  attr_reader :observations
28  attr_reader :control
29  attr_reader :candidates
30  attr_reader :matched
31  attr_reader :experiment
32  attr_reader :observations
33  attr_reader :control
34  attr_reader :candidates
35  attr_reader :matched
36  attr_reader :experiment
37  attr_reader :observations
38  attr_reader :control
39  attr_reader :candidates
40  attr_reader :matched
41  attr_reader :experiment
42  attr_reader :observations
43  attr_reader :control
44  attr_reader :candidates
45  attr_reader :matched
46  attr_reader :experiment
47  attr_reader :observations
48  attr_reader :control
49  attr_reader :candidates
50  attr_reader :matched
51  attr_reader :experiment
52  attr_reader :observations
53  attr_reader :control
54  attr_reader :candidates
55  attr_reader :matched
56  attr_reader :experiment
57  attr_reader :observations
58  attr_reader :control
59  attr_reader :candidates
60  attr_reader :matched
61  attr_reader :experiment
62  attr_reader :observations
63  attr_reader :control
64  attr_reader :candidates
65  attr_reader :matched
66  attr_reader :experiment
67  attr_reader :observations
68  attr_reader :control
69  attr_reader :candidates
70  attr_reader :matched
71  attr_reader :experiment
72  attr_reader :observations
73  attr_reader :control
74  attr_reader :candidates
75  attr_reader :matched
76  attr_reader :experiment
77  attr_reader :observations
78  attr_reader :control
79  attr_reader :candidates
80  attr_reader :matched
81  attr_reader :experiment
82  attr_reader :observations
83  attr_reader :control
84  attr_reader :candidates
85  attr_reader :matched
86  attr_reader :experiment
87  attr_reader :observations
88  attr_reader :control
89  attr_reader :candidates
90  attr_reader :matched
91  attr_reader :experiment
92  attr_reader :observations
93  attr_reader :control
94  attr_reader :candidates
95  attr_reader :matched
96  attr_reader :experiment
97  attr_reader :observations
98  attr_reader :control
99  attr_reader :candidates
100 attr_reader :matched
```

© Caendra Inc. 2019  
All Rights Reserved

# Table of Contents

## Module 3 | Web Applications

---

3.1 Introduction

3.2 HTTP Protocol Basics

3.3 HTTP Cookies

3.4 HTTP Sessions

3.5 Same Origin Policy

3.6 Burp Suite



# Learning Objectives

By the end of this module, you should have a better understanding of:

- HTTP Protocol
- Burp Suite basics



# Introduction



## 3.1 Introduction

**Web applications** are applications running on **web servers** and accessible via web browsers.

Many people interact with web applications every day, as currently, almost every website on the Internet includes some kind of **intelligence** in its web pages.

The intelligence could either be on the client side or the server side.



## 3.1 Introduction

The web app world is extremely **heterogeneous**. Every web application is different from others because developers have many ways to accomplish the same task.

As paraphrased from Stan Lee, “*with great flexibility comes great power of messing things up*”.

In other words, having flexibility in web app development also means having flexibility in creating insecure code.

```
21 # default context is the experiment's context
22
23 # Note: context is the experiment's context
24 # observations is an array of Observations, in this case
25 # control is the control observation
26
27 def initialize(experiment, observations = [], control = null)
28   @experiment = experiment
29   @observations = observations
30   @control = control
31   freeze
32 end
33
34 # Return the experiment's context
35 def context
36   @experiment.context
37 end
38
39 # Return the name of the experiment
40 def experiment_name
41   @experiment.name
42 end
43
44 # Return the result of the experiment
45 def result
46   @result
47 end
```



# 3.1 Introduction

To understand web application security, you need to know some web application fundamental aspects:

HTTP  
Protocol  
Basics

Cookies

Sessions

Same Origin  
Policy





# HTTP Protocol Basics





## 3.2 HTTP Protocol Basics

### How does this support my pentesting career?

- The ability to exploit web applications and find vulnerabilities in web servers and services
- Web applications technology is used market-wide also by desktop or mobile applications

## 3.2 HTTP Protocol Basics

**Hypertext Transfer Protocol (HTTP)** is the most used application protocol on the Internet. It is the client-server protocol used to transfer web pages and web application data.

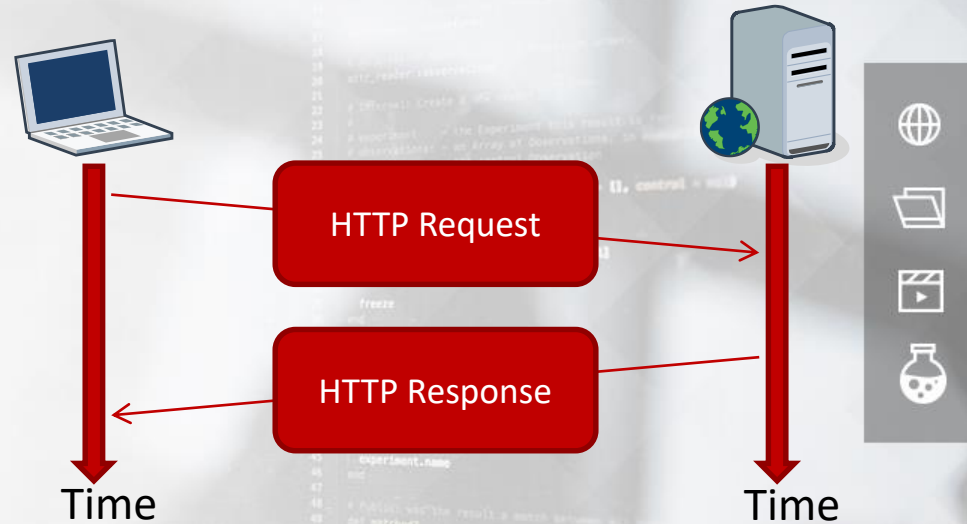
In HTTP, the client, usually a web browser, connects to a web server such as **MS IIS** or **Apache HTTP Server**. HTTP is also used under the hood by many mobile and modern applications.



## 3.2 HTTP Protocol Basics

During an HTTP communication, the client and the server exchange **messages**.

The client sends **requests** to the server and gets back **responses**.



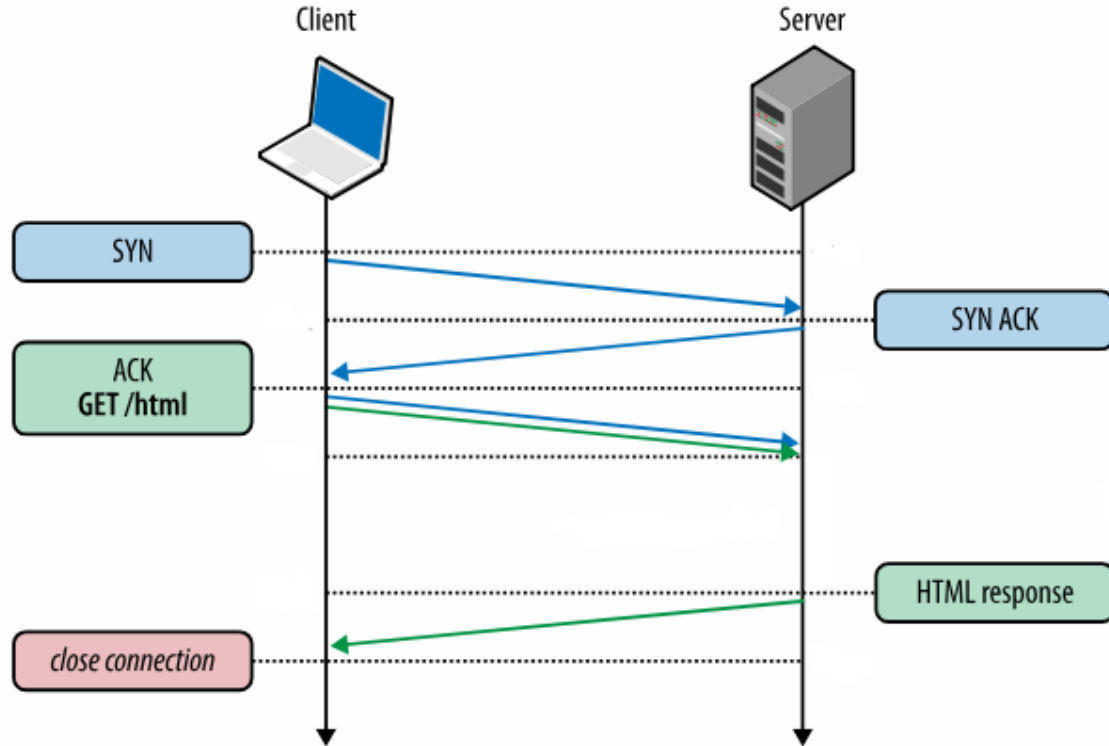
## 3.2 HTTP Protocol Basics

HTTP works on top of TCP protocol.

That means, first a TCP connection is established, and then the client sends its request, and waits for the answer. The server processes the request and sends back its answer, providing a status code and appropriate data.



## 3.2 HTTP Protocol Basics



## 3.2 HTTP Protocol Basics

The format of an HTTP message is:

Headers\r\n

\r\n

Message Body\r\n



## 3.2 HTTP Protocol Basics

To end lines in HTTP, you have to use the `\r` (carriage return) and the `\n` (newline) characters.

The header contains a request followed by some header fields. Every header field has the following format:

- `Header-name: header value`



### 3.2.1 HTTP Requests

The following is an HTTP request example.



**GET** / HTTP/1.1

**Host:** www.elearnsecurity.com

```
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:31.0) Gecko/20100101 Firefox/31.0
Iceweasel/31.2.0
```

**Accept:** text/html

**Accept-Language:** en-US,en;q=0.5

**Accept-Encoding:** gzip, deflate

**Connection:** keep-alive

```

10  # Create an experiment
11  # Experiment ID
12  # Experiment name
13  # Experiment description
14  # Experiment context
15  # Experiment control
16  # Experiment observations
17  # Experiment candidates
18  # Experiment evaluate_candidates
19  # Experiment freeze
20  # Experiment unfreeze
21  # Experiment get_context
22  # Experiment get_experiment_context
23  # Experiment get_name
24  # Experiment get_experiment_name
25  # Experiment get_experiment_name
26  # Experiment get_name
27  # Experiment get_name
28  # Experiment get_name
29  # Experiment get_name
30  # Experiment get_name
31  # Experiment get_name
32  # Experiment get_name
33  # Experiment get_name
34  # Experiment get_name
35  # Experiment get_name
36  # Experiment get_name
37  # Experiment get_name
38  # Experiment get_name
39  # Experiment get_name
40  # Experiment get_name
41  # Experiment get_name
42  # Experiment get_name
43  # Experiment get_name
44  # Experiment get_name
45  # Experiment get_name
46  # Experiment get_name
47  # Experiment get_name
48  # Experiment get_name
49  # Experiment get_name
50  # Experiment get_name
51  # Experiment get_name
52  # Experiment get_name
53  # Experiment get_name
54  # Experiment get_name
55  # Experiment get_name
56  # Experiment get_name
57  # Experiment get_name
58  # Experiment get_name
59  # Experiment get_name
60  # Experiment get_name
61  # Experiment get_name
62  # Experiment get_name
63  # Experiment get_name
64  # Experiment get_name
65  # Experiment get_name
66  # Experiment get_name
67  # Experiment get_name
68  # Experiment get_name
69  # Experiment get_name
70  # Experiment get_name
71  # Experiment get_name
72  # Experiment get_name
73  # Experiment get_name
74  # Experiment get_name
75  # Experiment get_name
76  # Experiment get_name
77  # Experiment get_name
78  # Experiment get_name
79  # Experiment get_name
80  # Experiment get_name
81  # Experiment get_name
82  # Experiment get_name
83  # Experiment get_name
84  # Experiment get_name
85  # Experiment get_name
86  # Experiment get_name
87  # Experiment get_name
88  # Experiment get_name
89  # Experiment get_name
90  # Experiment get_name
91  # Experiment get_name
92  # Experiment get_name
93  # Experiment get_name
94  # Experiment get_name
95  # Experiment get_name
96  # Experiment get_name
97  # Experiment get_name
98  # Experiment get_name
99  # Experiment get_name
100 # Experiment get_name

```



## 3.2.1 HTTP Requests

This request has an empty body, as there is nothing after the two empty lines following the headers.

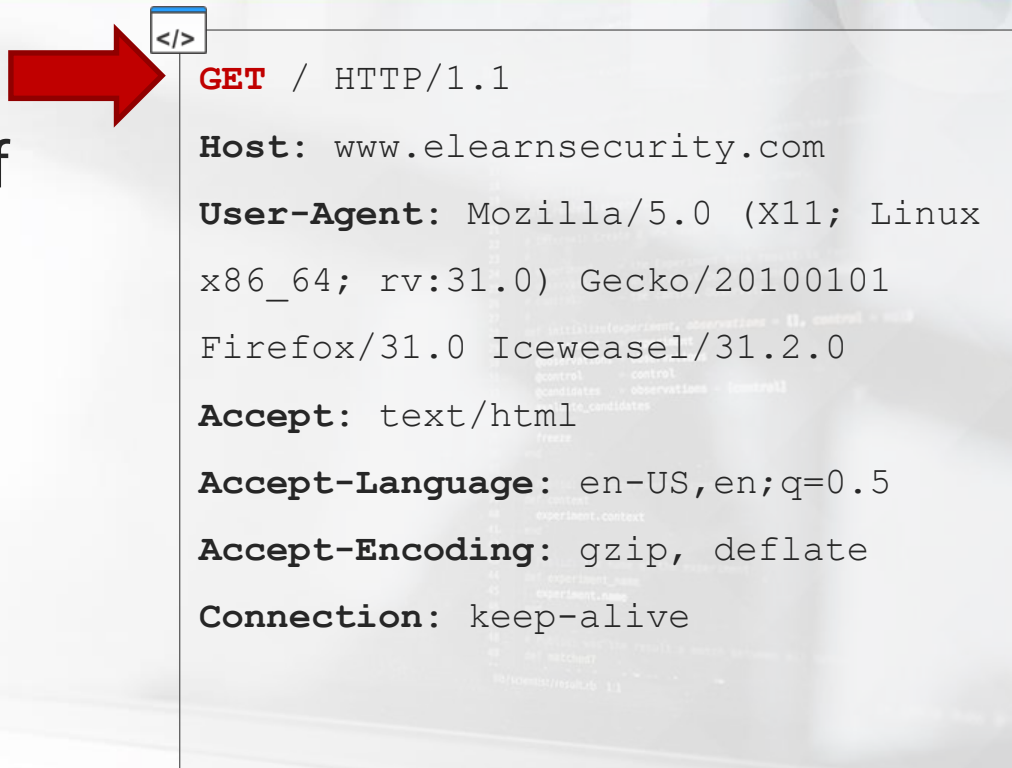


```
GET / HTTP/1.1
Host: www.elearnsecurity.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:31.0) Gecko/20100101 Firefox/31.0
Iceweasel/31.2.0
Accept: text/html
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

## 3.2.1 HTTP Requests

This is the **HTTP verb** of the request.


The HTTP verb, or request method, states the type of the request.



## 3.2.1 HTTP Requests

GET is used when opening web resources.

If you open a browser and type `www.elearnsecurity.com` in the address bar, your browser will send this very same request to the server.




```
</>  
GET / HTTP/1.1  
  
Host: www.elearnsecurity.com  
  
User-Agent: Mozilla/5.0 (X11; Linux  
x86_64; rv:31.0) Gecko/20100101  
Firefox/31.0 Iceweasel/31.2.0  
  
Accept: text/html  
  
Accept-Language: en-US,en;q=0.5  
  
Accept-Encoding: gzip, deflate  
  
Connection: keep-alive
```

## 3.2.1 HTTP Requests

After the HTTP VERB you can see the **path** (/) and the **protocol version** (HTTP 1.1).

The path tells the server which resource the browser is asking for. The protocol version tells the server how to communicate with the browser.




```
</>  
GET / HTTP/1.1  
  
Host: www.elearnsecurity.com  
  
User-Agent: Mozilla/5.0 (X11; Linux  
x86_64; rv:31.0) Gecko/20100101  
Firefox/31.0 Iceweasel/31.2.0  
  
Accept: text/html  
  
Accept-Language: en-US,en;q=0.5  
  
Accept-Encoding: gzip, deflate  
  
Connection: keep-alive
```

## 3.2.1 HTTP Requests

There are many HTTP methods, like:

- PUT
- TRACE
- HEAD
- POST

These are only a few, but know that there are many more out there.



```
</>  
GET / HTTP/1.1  
  
Host: www.elearnsecurity.com  
  
User-Agent: Mozilla/5.0 (X11; Linux  
x86_64; rv:31.0) Gecko/20100101  
Firefox/31.0 Iceweasel/31.2.0  
  
Accept: text/html  
  
Accept-Language: en-US,en;q=0.5  
  
Accept-Encoding: gzip, deflate  
  
Connection: keep-alive
```

## 3.2.1 HTTP Requests

The **Host** header field specifies the Internet hostname and port number of the resource being requested.

A web server can host multiple websites. This header field tells the server which site the client is asking for.



```
GET / HTTP/1.1
Host: www.elearnsecurity.com
User-Agent: Mozilla/5.0 (X11; Linux
x86_64; rv:31.0) Gecko/20100101
Firefox/31.0 Iceweasel/31.2.0
Accept: text/html
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```




## 3.2.1 HTTP Requests

The host value is obtained from the URI of the resource.

In this case:

`www.elearnsecurity.com`



```
</>  
GET / HTTP/1.1  
Host: www.elearnsecurity.com  
User-Agent: Mozilla/5.0 (X11; Linux  
x86_64; rv:31.0) Gecko/20100101  
Firefox/31.0 Iceweasel/31.2.0  
Accept: text/html  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive
```

## 3.2.1 HTTP Requests

**User-Agent** tells the server what client software is issuing the request.

A client could be:

- **Firefox**
- **Internet Explorer**
- **Safari**
- **Opera**
- **Chrome**
- **A mobile app...**



```
GET / HTTP/1.1
Host: www.elearnsecurity.com
User-Agent: Mozilla/5.0 (X11; Linux
x86_64; rv:31.0) Gecko/20100101
Firefox/31.0 Iceweasel/31.2.0
Accept: text/html
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

## 3.2.1 HTTP Requests

It also reveals to the server the operating system version.



```
GET / HTTP/1.1  
  
Host: www.elearnsecurity.com  
  
User-Agent: Mozilla/5.0 (X11; Linux  
x86_64; rv:31.0) Gecko/20100101  
Firefox/31.0 Iceweasel/31.2.0  
  
Accept: text/html  
  
Accept-Language: en-US,en;q=0.5  
  
Accept-Encoding: gzip, deflate  
  
Connection: keep-alive
```

## 3.2.1 HTTP Requests

The browser sends the Accept header field to specify which document type it is expecting in the response.



```
GET / HTTP/1.1
Host: www.elearnsecurity.com
User-Agent: Mozilla/5.0 (X11; Linux
x86_64; rv:31.0) Gecko/20100101
Firefox/31.0 Iceweasel/31.2.0
Accept: text/html
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

## 3.2.1 HTTP Requests

Similarly, with **Accept-Language**, the browser can ask for a specific (human) language in the response.



```
GET / HTTP/1.1
Host: www.elearnsecurity.com
User-Agent: Mozilla/5.0 (X11; Linux
x86_64; rv:31.0) Gecko/20100101
Firefox/31.0 Iceweasel/31.2.0
Accept: text/html
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

## 3.2.1 HTTP Requests

**Accept-Encoding** works similarly to Accept but restricts the content encoding, not the content itself.

In this case, the browser accepts two types of compression:

- gzip
- deflate



```
GET / HTTP/1.1

Host: www.elearnsecurity.com

User-Agent: Mozilla/5.0 (X11; Linux
x86_64; rv:31.0) Gecko/20100101
Firefox/31.0 Iceweasel/31.2.0

Accept: text/html

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Connection: keep-alive
```

## 3.2.1 HTTP Requests

The **Connection** header field allows the sender to specify options that are desired for that particular connection.

Future communications with the server will reuse the current connection.



```
</>  
GET / HTTP/1.1  
Host: www.elearnsecurity.com  
User-Agent: Mozilla/5.0 (X11; Linux  
x86_64; rv:31.0) Gecko/20100101  
Firefox/31.0 Iceweasel/31.2.0  
Accept: text/html  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive
```



## 3.2.2 HTTP Responses

When the server receives a request, it processes it and then sends an **HTTP response** to the client. The response has its own header format.

```
</>  
HTTP/1.1 200 OK  
Date: Wed, 19 Nov 2014 10:06:45 GMT  
Cache-Control: private, max-age=0  
Content-Type: text/html; charset=UTF-8  
Content-Encoding: gzip  
Server: Apache/2.2.15 (CentOS)  
Content-Length: 99043
```

< PAGE CONTENT > ...

```
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
221
```

## 3.2.2 HTTP Responses

As you can see, this response has a message body (< PAGE CONTENT >). The header and message body are separated by two empty lines (\r\n\r\n).

```
</>  
HTTP/1.1 200 OK  
Date: Wed, 19 Nov 2014 10:06:45 GMT  
Cache-Control: private, max-age=0  
Content-Type: text/html; charset=UTF-8  
Content-Encoding: gzip  
Server: Apache/2.2.15 (CentOS)  
Content-Length: 99043
```

```
< PAGE CONTENT > ...
```

```
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  

```

## 3.2.2 HTTP Responses

The first line of a Response message is the **Status-Line**, which consists of the **protocol version** (HTTP 1.1) followed by a numeric **status code** (200) and its relative **textual meaning** (OK).



**HTTP/1.1 200 OK**

**Date:** Wed, 19 Nov 2014 10:06:45 GMT

**Cache-Control:** private, max-age=0

**Content-Type:** text/html;  
charset=UTF-8

**Content-Encoding:** gzip

**Server:** Apache/2.2.15 (CentOS)

**Content-Length:** 99043

**< PAGE CONTENT > ...**

...

## 3.2.2 HTTP Responses

The more common status codes are:

- **200 OK:** the resource is found.
- **301 Moved Permanently:** the requested resource has been assigned a new permanent URI.
- **302 Found:** the resource is temporarily under another URI.



**HTTP/1.1 200 OK**

**Date:** Wed, 19 Nov 2014 10:06:45 GMT

**Cache-Control:** private, max-age=0

**Content-Type:** text/html;  
charset=UTF-8

**Content-Encoding:** gzip

**Server:** Apache/2.2.15 (CentOS)

**Content-Length:** 99043

**< PAGE CONTENT > ...**

...

## 3.2.2 HTTP Responses

- **403 Forbidden:** the client does not have enough privileges, and the server refuses to fulfill the request.
- **404 Not Found:** the server cannot find a resource matching the request.
- **500 Internal Server Error:** the server does not support the functionality required to fulfill the request.



**HTTP/1.1 200 OK**

**Date:** Wed, 19 Nov 2014 10:06:45 GMT

**Cache-Control:** private, max-age=0

**Content-Type:** text/html;  
charset=UTF-8

**Content-Encoding:** gzip

**Server:** Apache/2.2.15 (CentOS)

**Content-Length:** 99043

**< PAGE CONTENT > ...**

...

## 3.2.2 HTTP Responses

**Date** represents the date and time at which the message was originated.



**HTTP/1.1 200 OK**

**Date:** Wed, 19 Nov 2014 10:06:45 GMT

**Cache-Control:** private, max-age=0

**Content-Type:** text/html;  
charset=UTF-8

**Content-Encoding:** gzip

**Server:** Apache/2.2.15 (CentOS)

**Content-Length:** 99043

**< PAGE CONTENT > ...**

...

## 3.2.2 HTTP Responses

With the **Cache-Control** header, the server informs the client about cached content.



Using cached content saves bandwidth, as it prevents the client from re-requesting unmodified content.



```
HTTP/1.1 200 OK
Date: Wed, 19 Nov 2014 10:06:45 GMT
Cache-Control: private, max-age=0
Content-Type: text/html;
charset=UTF-8
Content-Encoding: gzip
Server: Apache/2.2.15 (CentOS)
Content-Length: 99043
```

```
< PAGE CONTENT > ...
...
```





## 3.2.2 HTTP Responses

**Content-Type** lets the client know how to interpret the body of the message.



**HTTP/1.1 200 OK**

**Date:** Wed, 19 Nov 2014 10:06:45 GMT

**Cache-Control:** private, max-age=0

**Content-Type:** text/html;  
charset=UTF-8

**Content-Encoding:** gzip

**Server:** Apache/2.2.15 (CentOS)

**Content-Length:** 99043

**< PAGE CONTENT > ...**

...

## 3.2.2 HTTP Responses

**Content-Encoding**  
extends Content-Type.

In this case, the  
message body is  
compressed with gzip.



```
HTTP/1.1 200 OK
Date: Wed, 19 Nov 2014 10:06:45 GMT
Cache-Control: private, max-age=0
Content-Type: text/html;
charset=UTF-8
Content-Encoding: gzip
Server: Apache/2.2.15 (CentOS)
Content-Length: 99043

< PAGE CONTENT > ...
...
```

## 3.2.2 HTTP Responses

The **Server** header field simply contains the header of the server that generated the content.

This (optional) field is very useful during a pentest to identify the software running on a server.



```
HTTP/1.1 200 OK
Date: Wed, 19 Nov 2014 10:06:45 GMT
Cache-Control: private, max-age=0
Content-Type: text/html;
charset=UTF-8
Content-Encoding: gzip
Server: Apache/2.2.15 (CentOS)
Content-Length: 99043
```

```
< PAGE CONTENT > ...
...
```



## 3.2.2 HTTP Responses

**Content-Length**  
indicates the length, in  
bytes, of the message  
body.



```
HTTP/1.1 200 OK
Date: Wed, 19 Nov 2014 10:06:45 GMT
Cache-Control: private, max-age=0
Content-Type: text/html;
charset=UTF-8
Content-Encoding: gzip
Server: Apache/2.2.15 (CentOS)
Content-Length: 99043
```

```
< PAGE CONTENT > ...
...
```



## 3.2.3 HTTPS

Now that you know how HTTP works, let's see how to **protect it!**

HTTP content, as in every clear-text protocol, can be easily intercepted or mangled by an attacker on the path. Moreover, HTTP does not provide strong authentication between the parties.

## 3.2.3 HTTPS

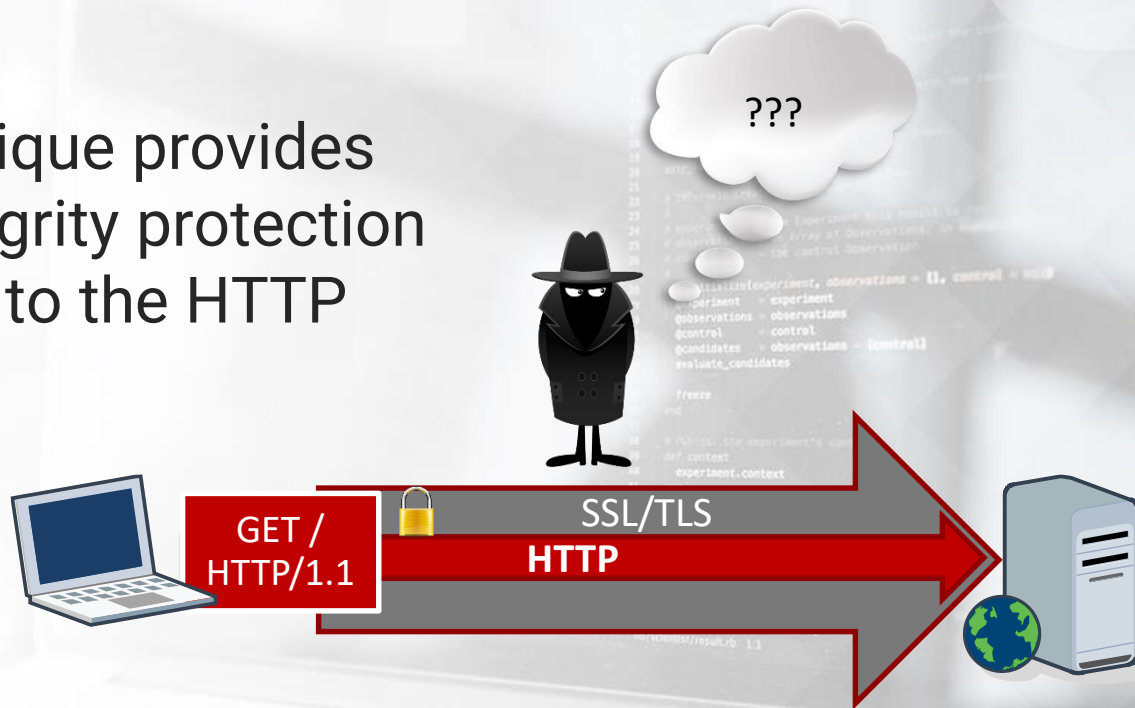
In the following slides, you will see how to **protect HTTP** using an **encryption layer**.

**HTTP Secure** (HTTPS), or HTTP over SSL/TLS, is a method to run HTTP which is a clear-text protocol over SSL/TLS, a cryptographic protocol.



## 3.2.3 HTTPS

This layering technique provides confidentiality, integrity protection and authentication to the HTTP protocol.





## 3.2.3 HTTPS

In other words, when using HTTPS:

- An attacker on the path cannot sniff the application layer communication.
- An attacker on the path cannot alter the application layer data.
- The client can tell the real identity of the server and, sometimes, vice-versa.



## 3.2.3 HTTPS

HTTPS offers encryption, which means that a network adjacent user is able to sniff the traffic, but he will not know:

- HTTP Request headers, body, target domain
- HTTP Response headers, body

On the other hand, when inspecting HTTPS, one cannot know what domain is contacted and what data is exchanged.



### 3.2.3 HTTPS

A network adjacent user might recognize:

- Target IP address
- Target port
- DNS or similar protocols may disclose which domain user tries to resolve

### 3.2.3 HTTPS

# HTTPS does not protect against web application flaws! All the attacks against an application happen regardless of SSL/TLS.

The extra encryption layer just protects data exchanged between the client and the server. It does not protect from an attack against the application itself.

## 3.2.3 HTTPS

Attacks such as XSS and SQL injections will still work.

Understanding how HTTP and web applications work is fundamental to mount stealthy and effective attacks!



## 3.2.4 Video – HTTP and HTTPS Protocols Basics

### HTTP and HTTPS Protocols Basics

In this video, you will see HTTP and HTTPS in action both via command line tools or with Burp Proxy.

You will learn how to:

- Perform requests, receive responses and analyze HTTP(s) messages both manually and with an automated tool.
- Use Burp at the end of this module.



*\*Videos are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the resources drop-down in the appropriate module line.*

## 3.2.5 References

[HTTP Status Code Reference](#): a document referencing HTTP status codes and their meaning.

[SSL/TLS Strong Encryption: An Introduction](#): Apache's introduction on protecting HTTP by means of SSL/TLS.

[HTTP Overview, History, Versions and Standards](#): History and evolution of HTTP.

<http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

[http://httpd.apache.org/docs/2.2/ssl/ssl\\_intro.html](http://httpd.apache.org/docs/2.2/ssl/ssl_intro.html)

[http://www.tcpipguide.com/free/t\\_HTTPOverviewHistoryVersionsandStandards.htm](http://www.tcpipguide.com/free/t_HTTPOverviewHistoryVersionsandStandards.htm)



# HTTP Cookies



## 3.3 HTTP Cookies

HTTP is a **stateless** protocol; this means that websites cannot keep the state of a visit across different HTTP requests.

In other words, every HTTP request is **completely unrelated** to the ones preceding and following it.





## 3.3 HTTP Cookies

How does this support my pentesting career?

- Cookies are foundation of authorization of many applications
- Often exploits rely on stealing cookies

## 3.3 HTTP Cookies

Cookies are not rocket science. They are just textual information installed by a website into the "cookie jar" of the web browser.

The **cookie jar** is the storage space where a web browser stores the cookies.



## 3.3.1 Cookies Format

A server can set a cookie via the **Set-Cookie** HTTP header field in a response message. A cookie contains the following attributes:

- The actual content
- An expiration date
- A path
- The domain
- Optional flags:
  - Http only flag
  - Secure flag



```
HTTP/1.1 200 OK
```

```
Date: Wed, 19 Nov 2014 10:06:45 GMT
```

```
Cache-Control: private, max-age=0
```

```
Content-Type: text/html;
```

```
charset=UTF-8
```

```
Content-Encoding: gzip
```

```
Server: Apache/2.2.15 (CentOS)
```

```
Set-Cookie: ID=Value; expires=Thu,  
21-May-2015 15:25:20 GMT; path=/;
```

```
domain=.example.site; HttpOnly
```

```
Content-Length: 99043
```

## 3.3.1 Cookies Format

Cookie content	{ ID=Value;
Expiration date	{ <b>expires</b> =Thu, 21-May-2015 15:25:20 GMT;
Path	{ <b>path</b> =/example/path;
Domain	{ <b>domain</b> =.example.site;
Flag-setting attributes	{ <b>HttpOnly</b> ; <b>Secure</b>





## 3.3.2 Cookies Handling

Browsers use domain, path, expires and flags attributes to choose whether or not to send a cookie in a request.

Cookies are sent only to the **valid domain/path** when they are **not expired** and according to their **flags**.



## 3.3.3 Cookie Domain

The **domain** field and the **path** field set the **scope** of the cookie. The browser sends the cookie only if the request is for the right domain.

When a web server installs a cookie, it sets the domain field, e.g., `elearnsecurity.com`. Then, the browser will use the cookie for every request sent to that domain and **all its subdomains**.



### 3.3.3 Cookie Domain

When a cookie has the domain attribute set to:

- **domain**=elearnsecurity.com
- or
- **domain**=.elearnsecurity.com

The browser will send the cookie to:

- www.elearnsecurity.com
- whatever.subdomain.elearnsecurity.com
- elearnsecurity.com



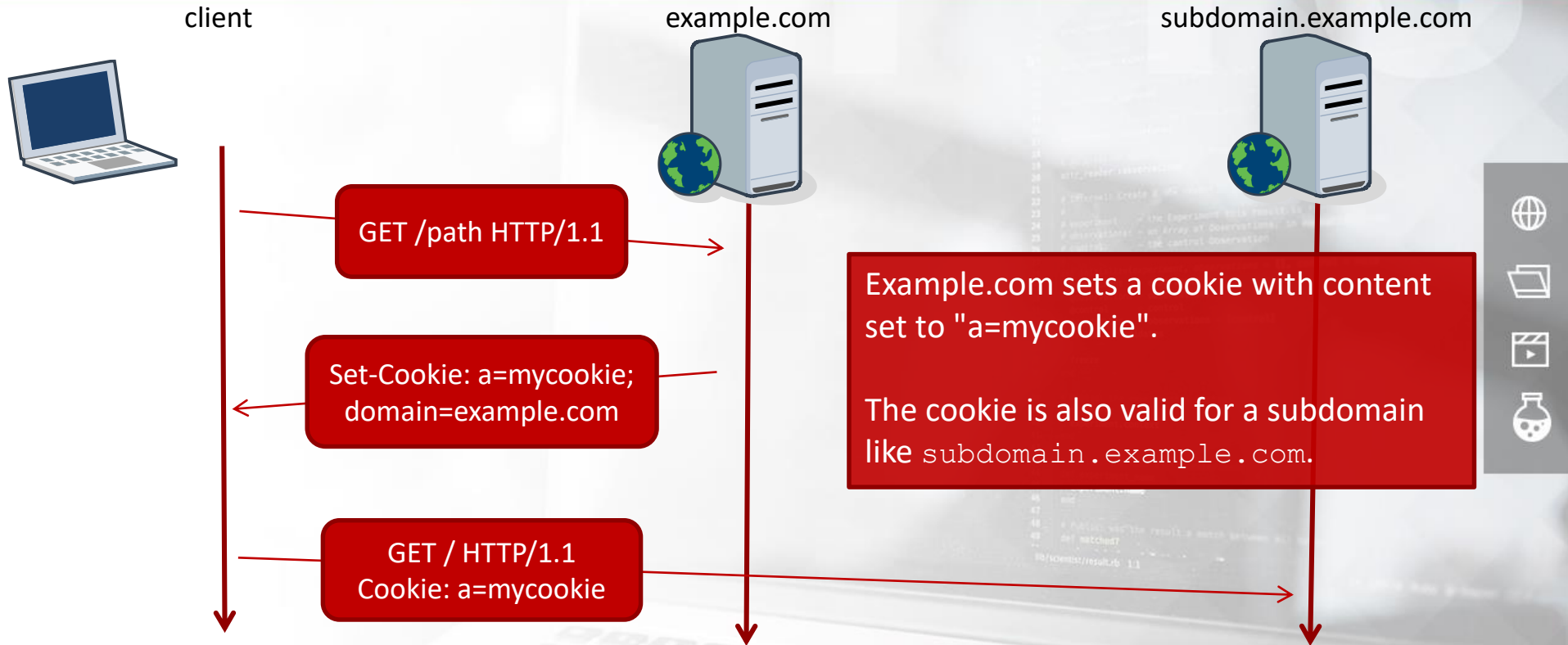
### 3.3.3 Cookie Domain

If the **server does not specify** the domain attribute, the browser will automatically set the domain as the server domain and set the cookie **host-only** flag; this means that the cookie will be sent **only to that precise hostname**.

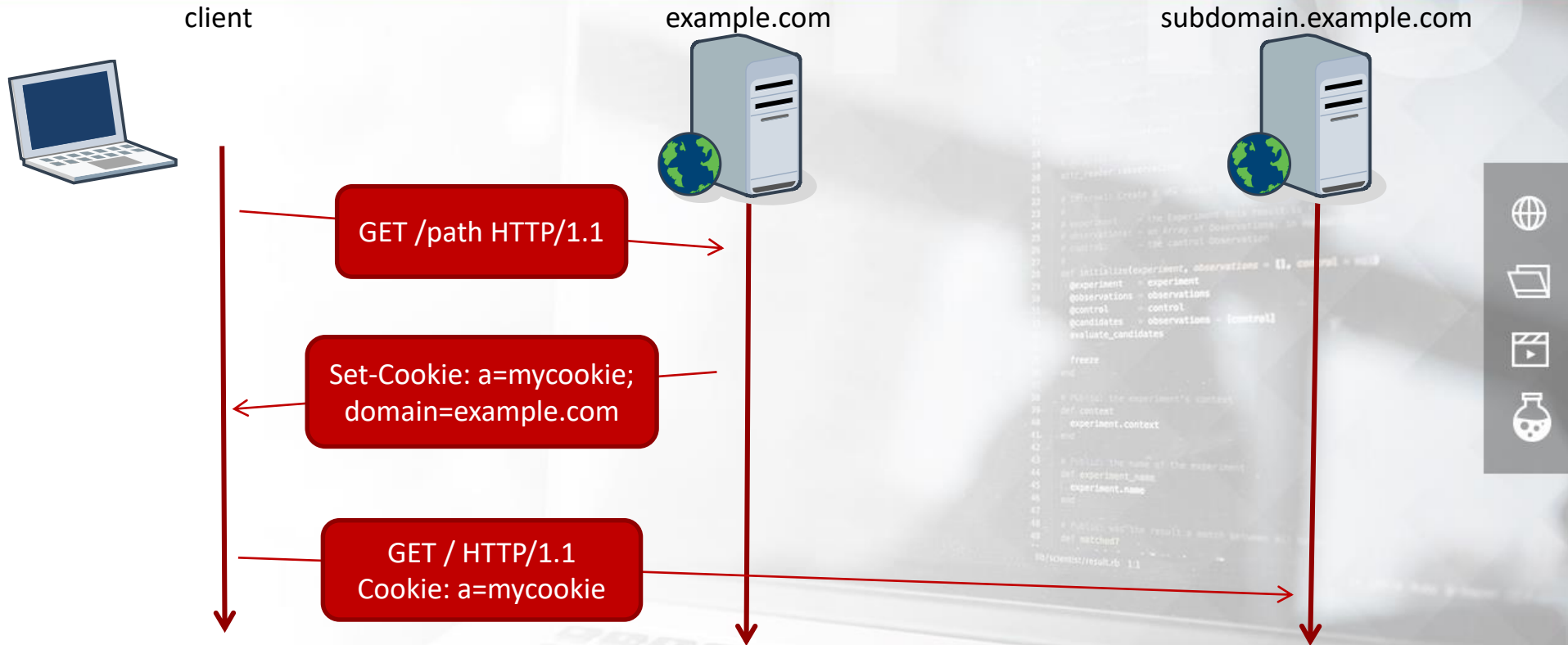
In the following examples, you will see how a browser chooses to send or not to send a cookie.



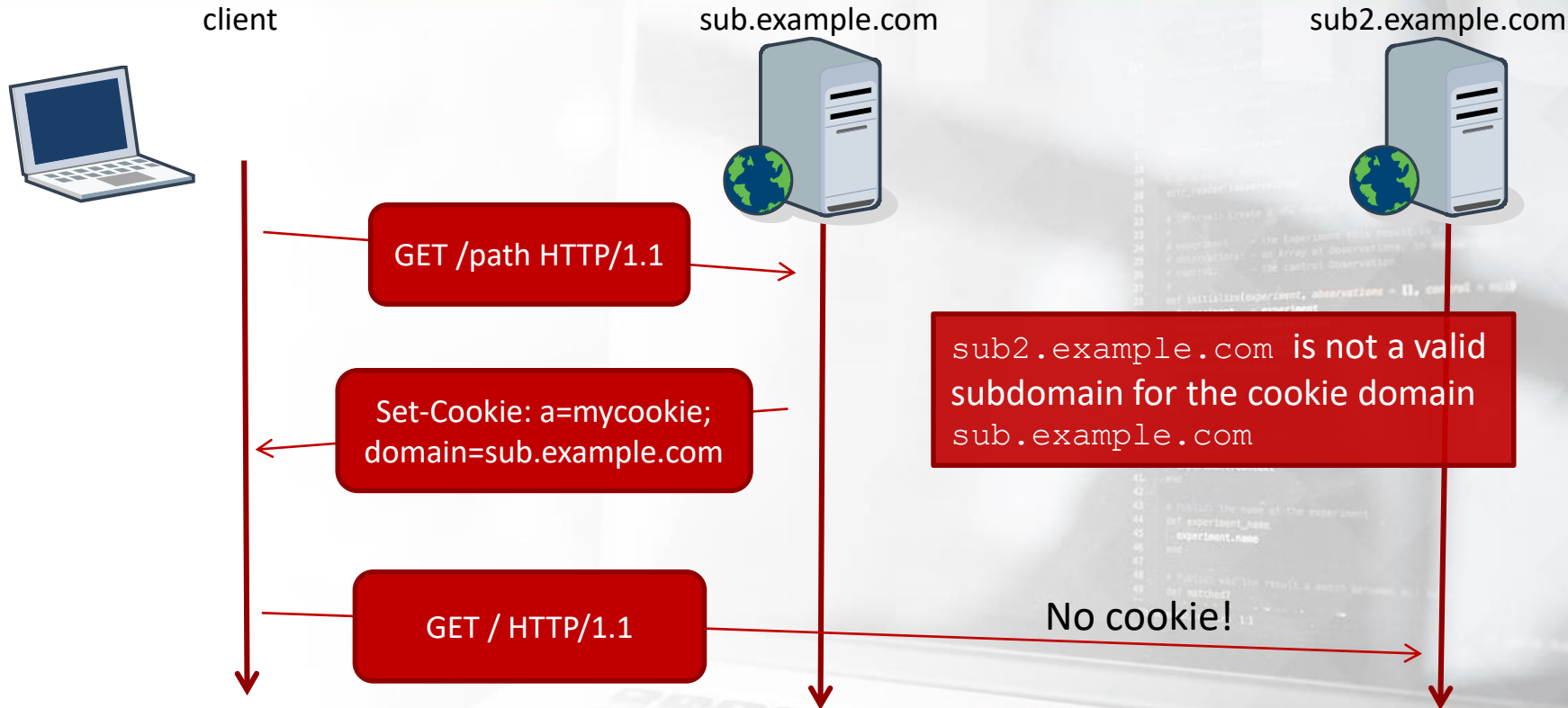
# 3.3.3.1 Cookie Domain Examples



# 3.3.3.1 Cookie Domain Examples

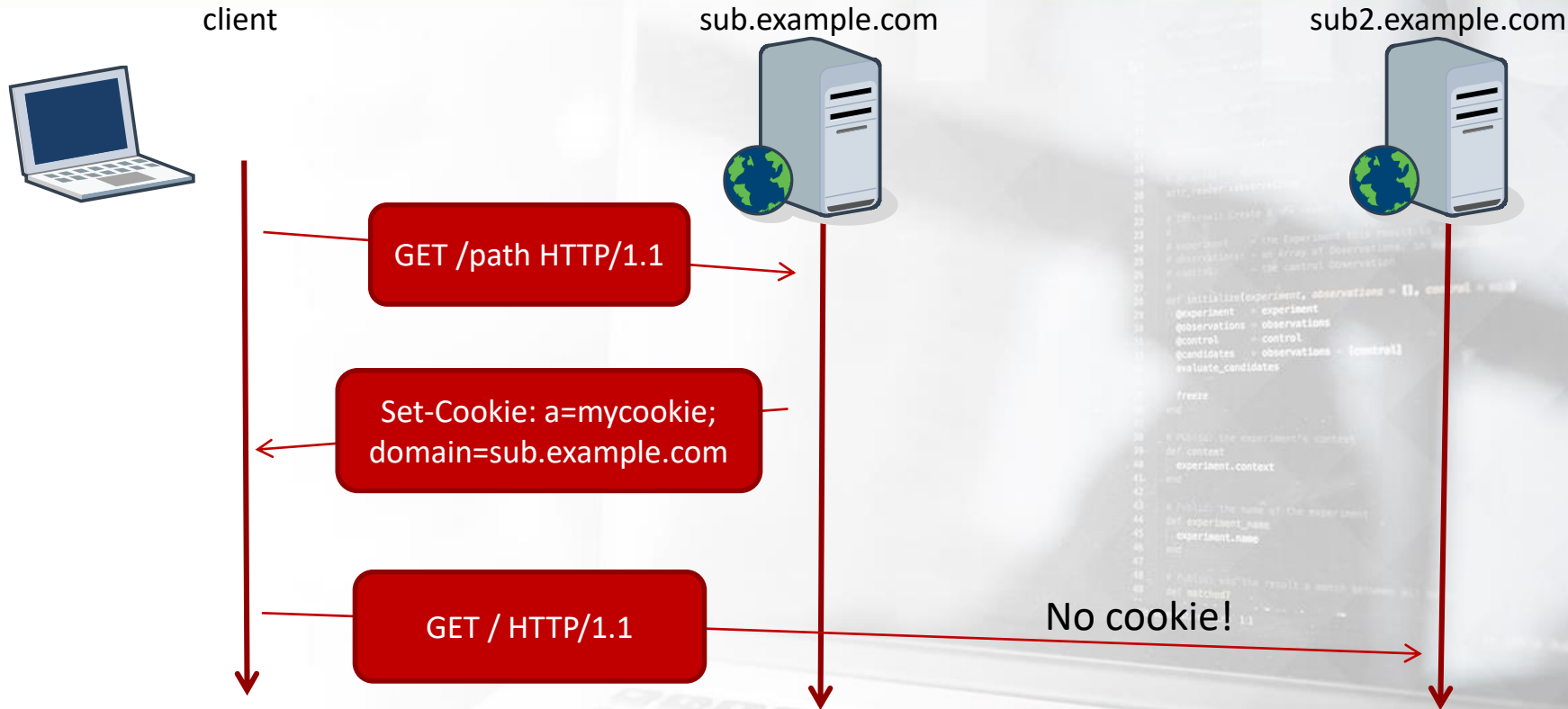


# 3.3.3.1 Cookie Domain Examples

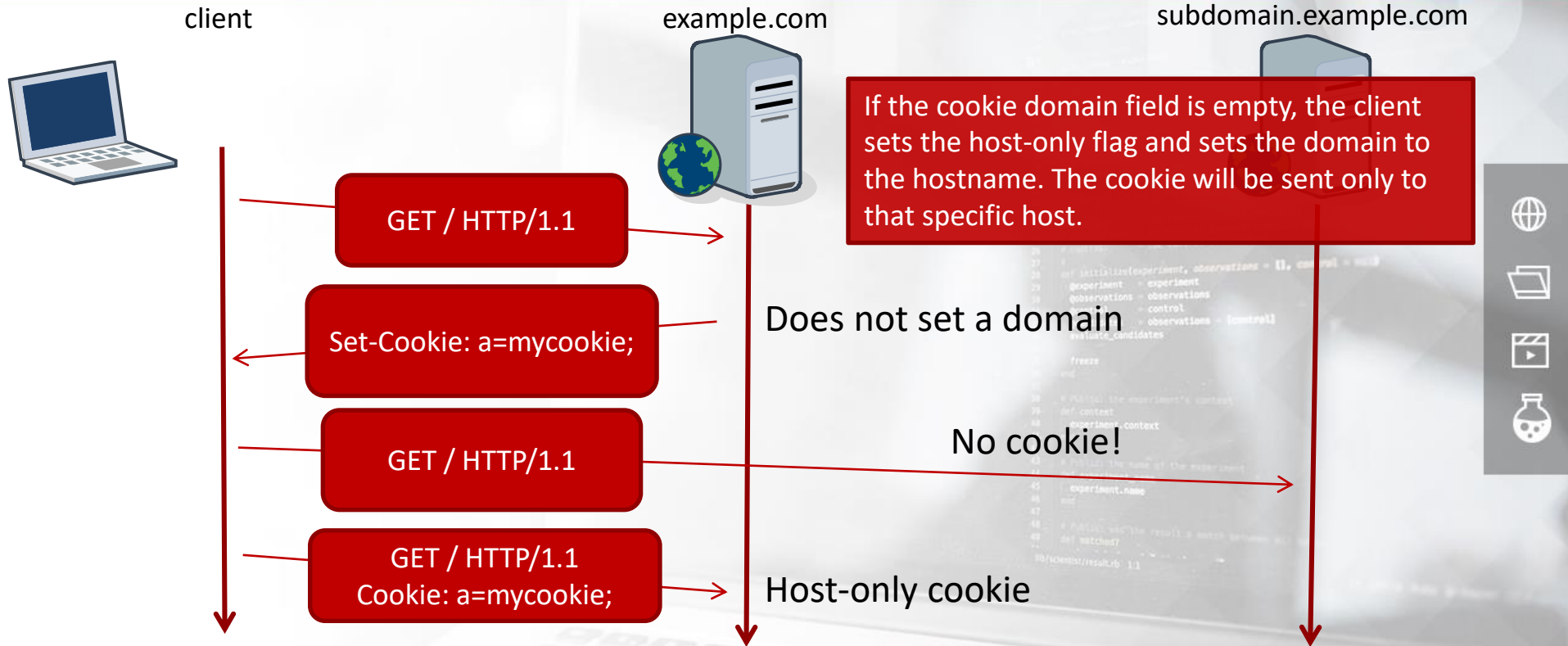




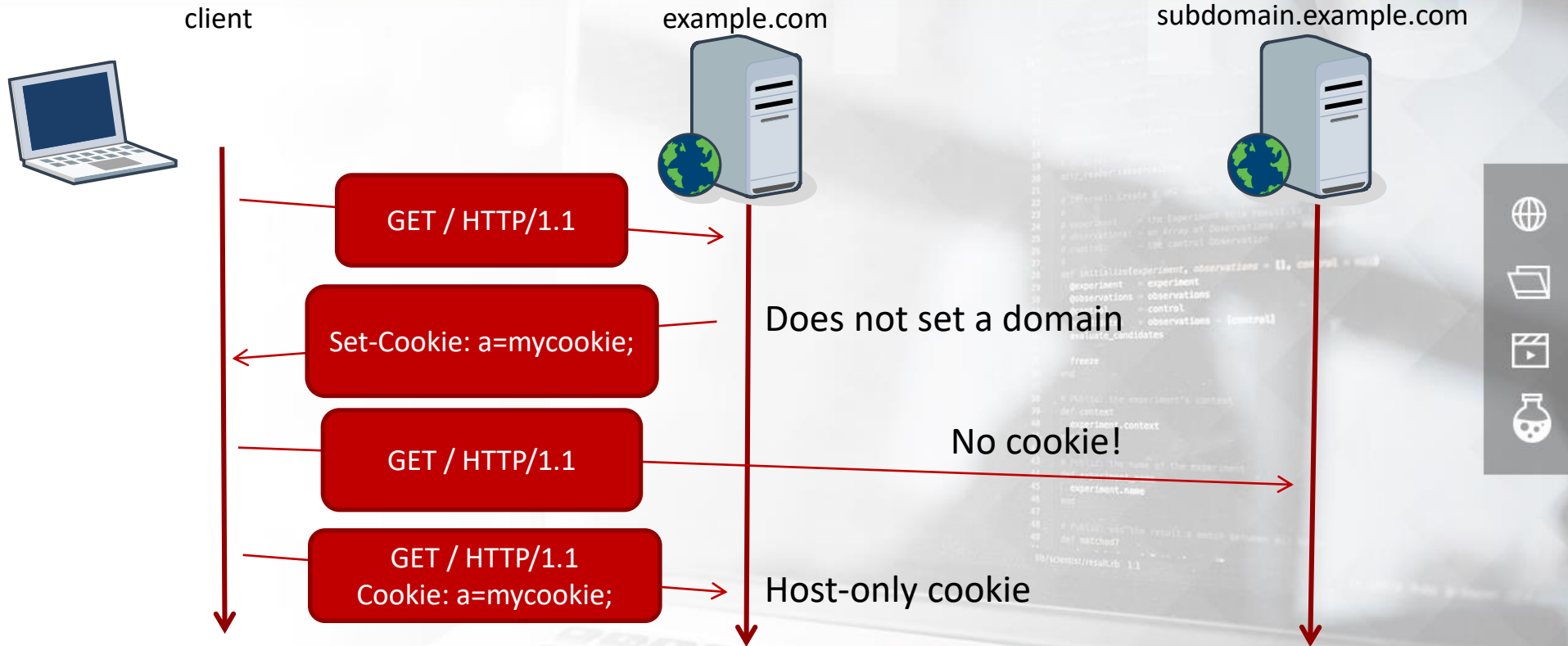
# 3.3.3.1 Cookie Domain Examples



# 3.3.3.1 Cookie Domain Examples



# 3.3.3.1 Cookie Domain Examples



## 3.3.4 Cookie Path

As we saw previously, the **path** and the **domain** attributes set the **scope** of a cookie.

The browser will send a cookie to the **right domain** and to **any subpath of the path** field value.

```
24 # @context: the experiment will result in 2
25 # observations: an array of Observations, in this case
26 # control: the control observation
27
28 def initialize(experiment, observations = [], control = null)
29   @experiment = experiment
30   @observations = observations
31   @control = control
32   @candidates = observations - [control]
33   evaluate_candidates
34
35   freeze
36
37   per context
38     experiment.context
39   end
40
41   # build the name of the experiment
42   def experiment_name
43     experiment.name
44   end
45
46   # build the result a match between an observation
47   def matches?
48     # ...
49   end
50
51   # ...
52 end
```



## 3.3.4 Cookie Path

When a cookie has the path attribute set to:

- `path=/the/path`

The browser will send the cookie to the right domain and to the resources in:

- `/the/path`
- `/the/path/sub`
- `/the/path/sub/sub/sub/path`

But, it will not send it to `/otherpath`.



### 3.3.5 Cookie Expires Attribute

The **expires** attribute sets the **validity time window** of a cookie.

A browser will not send an expired cookie to the server. Session cookies expire with the HTTP session; you will see more about that later in this module.

[illegible]

## 3.3.6 Cookie Http-Only Attribute

When a server installs a cookie into a client with the **http-only attribute**, the client will set the **http-only flag** for that cookie. This mechanism prevents JavaScript, Flash, Java and any other non-HTML technology from reading the cookie, thus preventing cookie stealing via XSS.

You will see how to exploit XSS vulnerabilities in the *Web Application Attacks* module.



## 3.3.7 Cookie Secure Attribute

**Secure flag** creates secure cookies that will only be sent over an **HTTPS** connection (they will not be sent over HTTP).



### 3.3.8 Cookie Content

A cookie can carry a number of values. A server can set multiple values with a single `Set-Cookie` header by specifying multiple `KEY=Value` pairs.

```
Set-Cookie: Username="john"; auth=1
```

Sets two values: one for username and one for auth.



## 3.3.9 Cookie Protocol

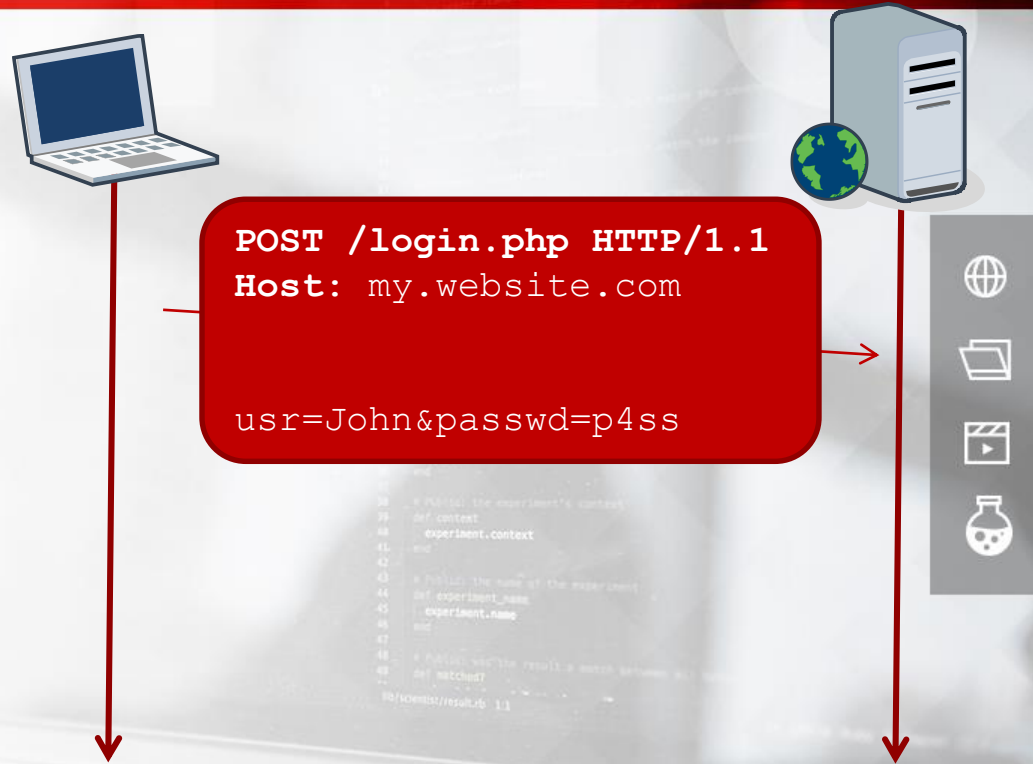
[RFC6265](#) states cookies format, how a server can install cookies and how a client can use them.

Let's see cookies in action with a simple example.

## 3.3.9 Cookie Protocol

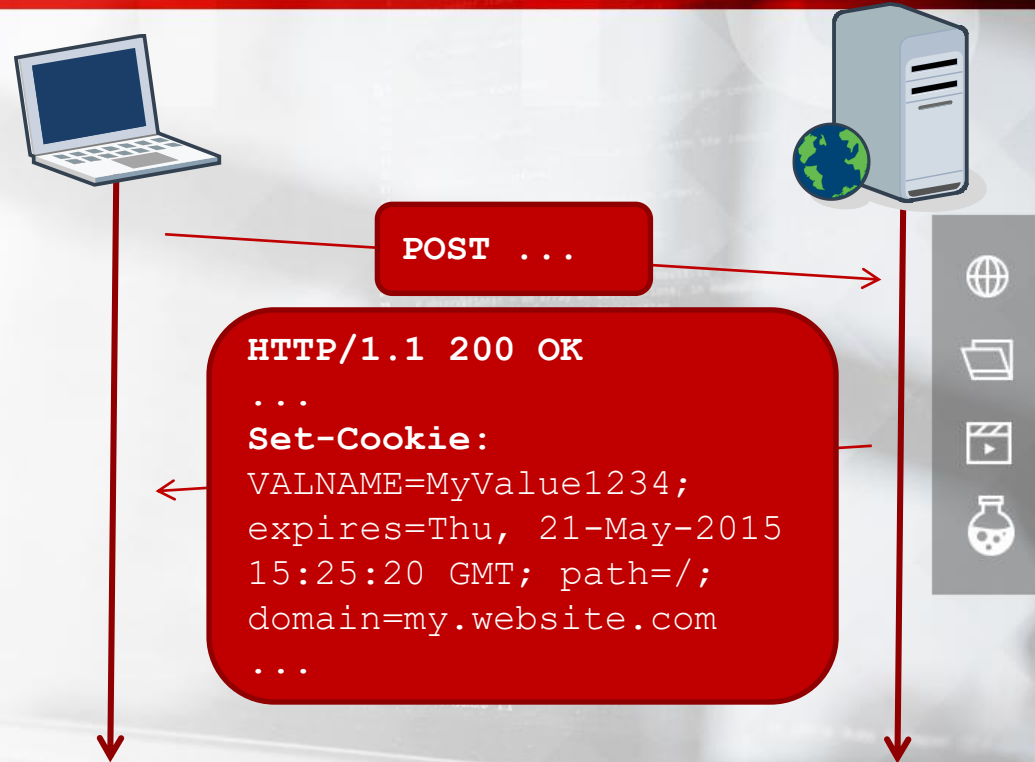
Cookies are often installed during a login.

In this example, the browser sends a **POST** request with the username and password.



## 3.3.9 Cookie Protocol

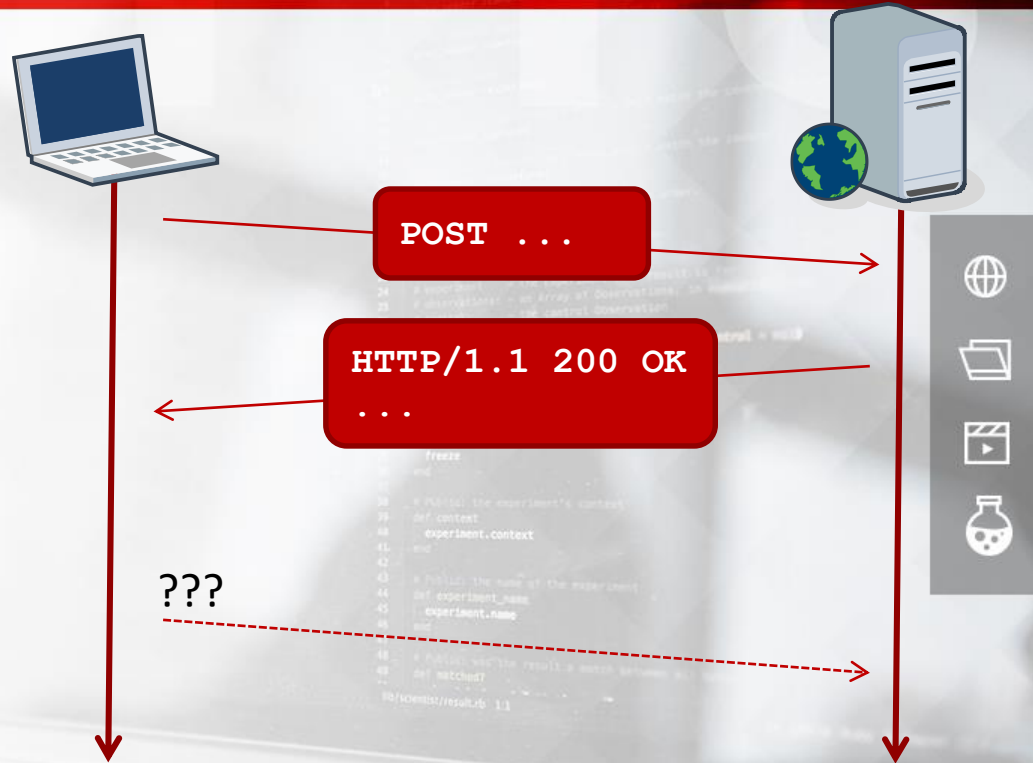
The server sends a response with a **Set-cookie** header field, thus telling the browser to install the cookie.



## 3.3.9 Cookie Protocol

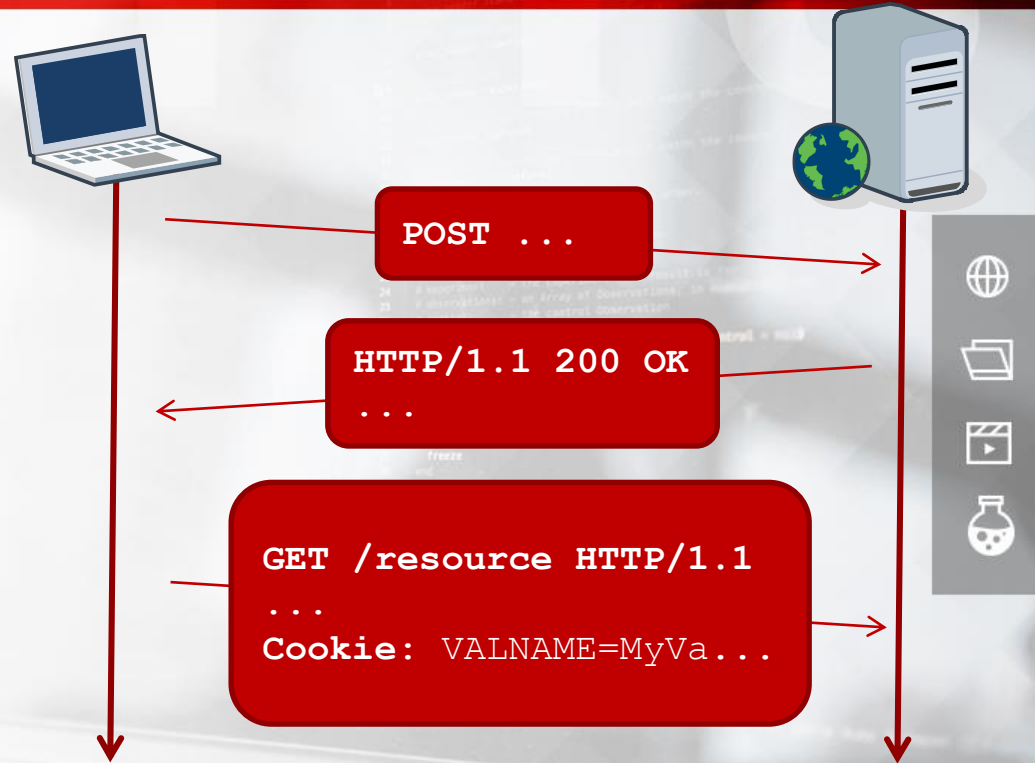
For every subsequent request, the browser considers:

- Domain
- Path
- Expiration
- Flags



## 3.3.9 Cookie Protocol

If all the checks pass, the browser will insert a **cookie**: header in the request.





# Sessions



## 3.4 Sessions

Sometimes the web developer prefers to store some information on the **server side** instead of the client side; this happens to hide the application logic or just to avoid the back and forth data transmission typical of cookies.



## 3.4 Sessions

**Sessions** are a mechanism that lets the website store variables specific for a given visit on the **server side**.

Each user session is identified by a **session id**, or token, which the server assigns to the client.



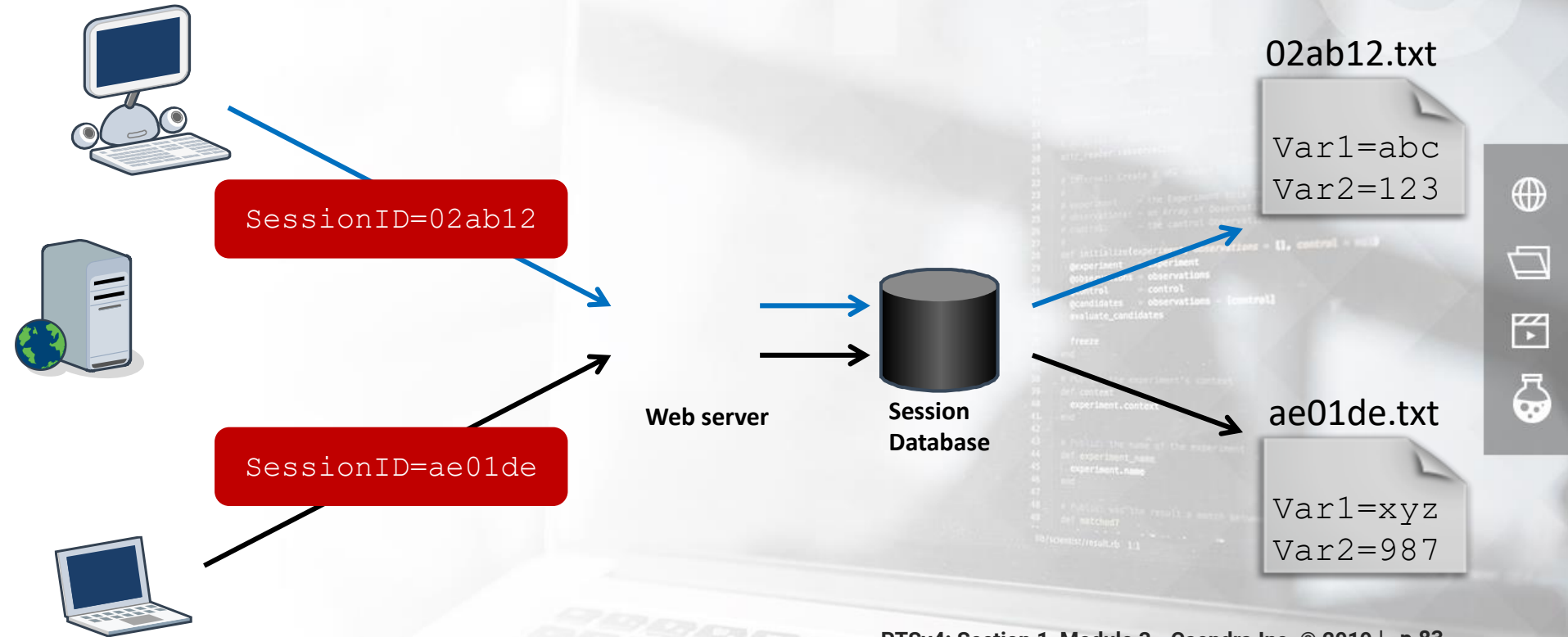
## 3.4 Sessions

The client then presents this ID for each subsequent request, thus being recognized by the server.

By means of the session ID, the **server retrieves the state of the client** and all its associated variables. The server stores Session IDs inside text files in its storage. You can find an example in the next slide.



## 3.4.1 Sessions Example



## 3.4.2 Session Cookies

Q

*How does a web application install session IDs on a web browser?*

A

*By using **session cookies**.*

```
24 # ...
25 # ...
26 # ...
27 # ...
28 # ...
29 # ...
30 # ...
31 # ...
32 # ...
33 # ...
34 # ...
35 # ...
36 # ...
37 # ...
38 # ...
39 # ...
40 # ...
41 # ...
42 # ...
43 # ...
44 # ...
45 # ...
46 # ...
47 # ...
48 # ...
49 # ...
50 # ...
51 # ...
52 # ...
53 # ...
54 # ...
55 # ...
56 # ...
57 # ...
58 # ...
59 # ...
60 # ...
61 # ...
62 # ...
63 # ...
64 # ...
65 # ...
66 # ...
67 # ...
68 # ...
69 # ...
70 # ...
71 # ...
72 # ...
73 # ...
74 # ...
75 # ...
76 # ...
77 # ...
78 # ...
79 # ...
80 # ...
81 # ...
82 # ...
83 # ...
84 # ...
85 # ...
86 # ...
87 # ...
88 # ...
89 # ...
90 # ...
91 # ...
92 # ...
93 # ...
94 # ...
95 # ...
96 # ...
97 # ...
98 # ...
99 # ...
100 # ...
```

It's now time to see how to use **HTTP sessions!**

## 3.4.2 Session Cookies

Session cookies just contain a single parameter value pair referring to the session.



- SESSION=0wvCtOBWDH8w
- PHPSESSID=13Kn5Z6Uo4pH
- JSESSIONID=W7DPUBgh7kTM



## 3.4.2 Session Cookies

Websites running PHP install session cookies by using the "PHPSESSID" parameter name, while JSP websites use "JSESSIONID". Each development language has its own default session parameter name.

Of course, the web developer can also choose to use a custom parameter name.

```
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216  
2217  
2218  
2219  
2220  
2221  
2222  
2223  
2224  
2225  
2226  
2227  
2228  
2229  
22
```

## 3.4.2 Session Cookies

If needed, servers install session cookies after a browser performs some kind of activity, like:

- Opening a specific page
- Changing settings in the web application
- Logging in

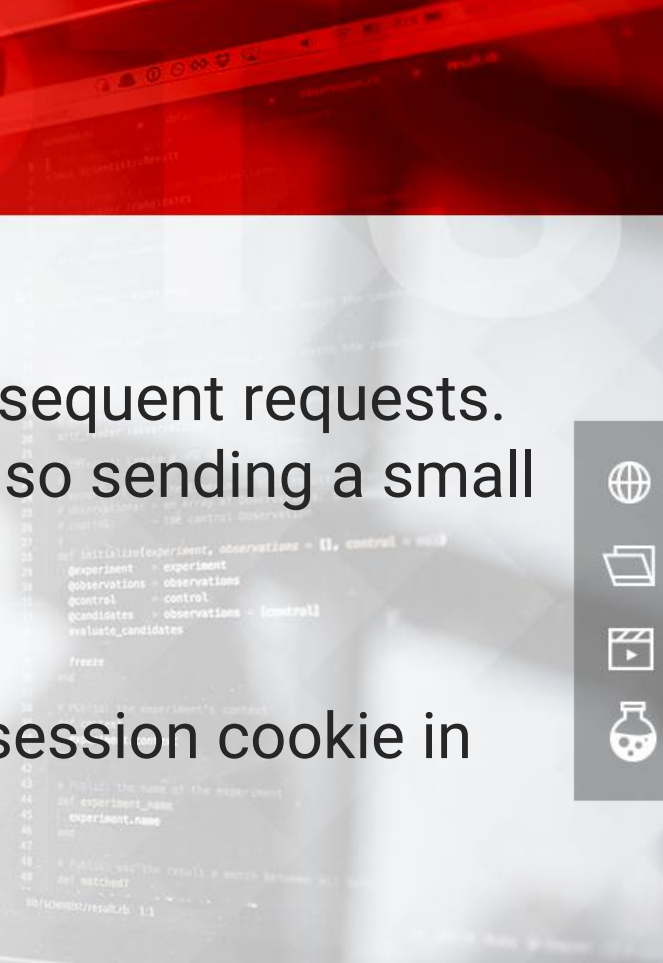
```
21 def initialize_experiment(experiment_name, observations = [], control = null)
22   # Initialize the experiment's context
23   @experiment = experiment
24   @observations = observations
25   @control = control
26   @candidates = observations + [control]
27   evaluate_candidates
28   freeze
29
30   # Return the experiment's context
31   @experiment.context
32 end
33
34 # Return the name of the experiment
35 def experiment_name
36   @experiment.name
37 end
38
39 # Return the result of a match between two candidates
40 def match
41   # ...
42 end
43
44 # Return the result of a match between two candidates
45 def match_result
46   # ...
47 end
```



## 3.4.2 Session Cookies

The browser then uses the cookie in subsequent requests. A session could contain many variables, so sending a small cookie keeps the bandwidth usage low.

In the following example, you can see a session cookie in action.



## 3.4.2.1 Session Cookies Example

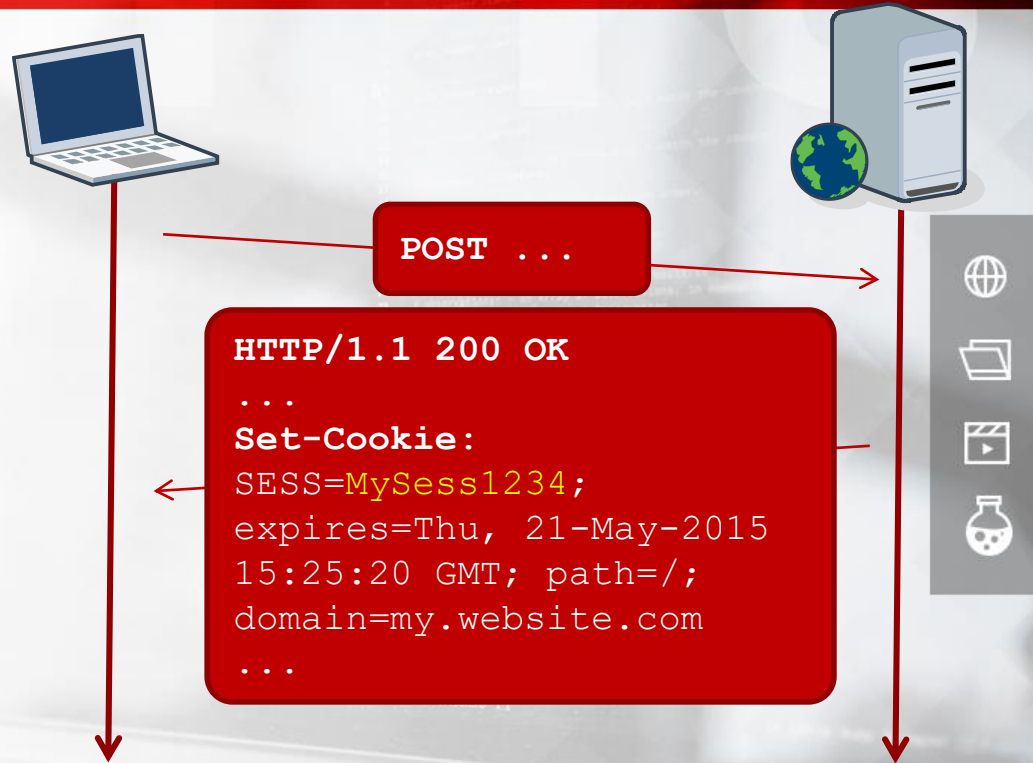
The client uses a login form to POST the user's credentials.



## 3.4.2.1 Session Cookies Example

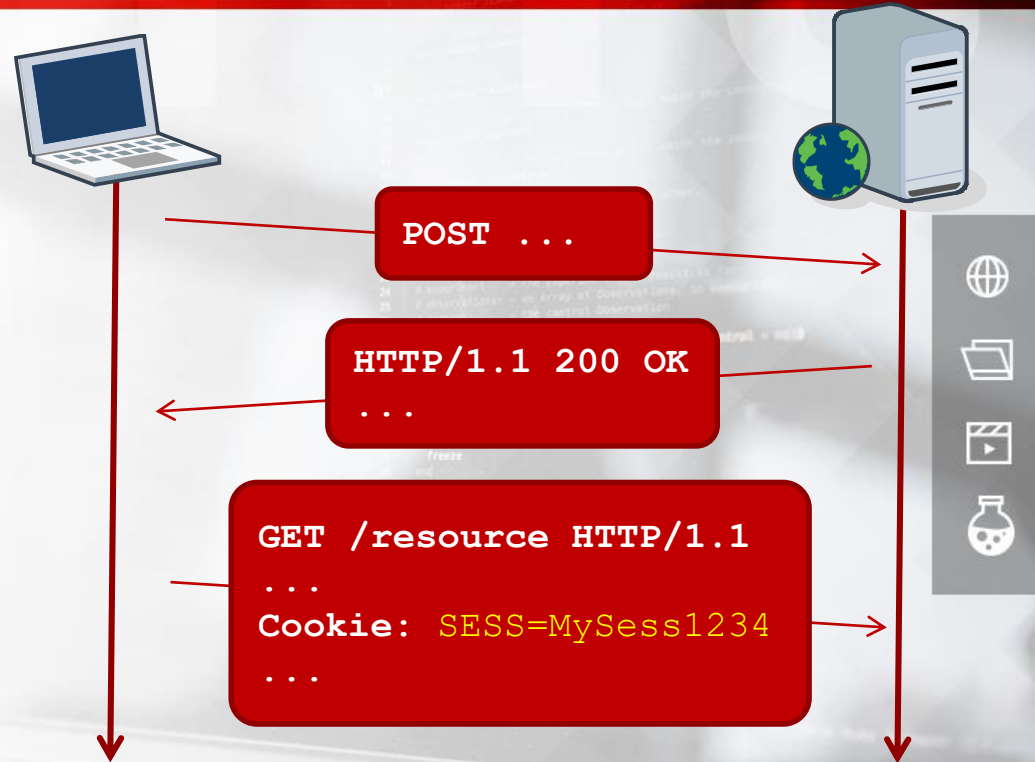
The server sends back a response with a Set-cookie header field.

The cookie contains the **session ID**.



## 3.4.2.1 Session Cookies Example

The browser will send back the cookie according to the cookie protocol, thus sending the **session ID**.



### 3.4.3 GET Requests

Session IDs can also be transmitted via **GET requests**.

```
http://example.site/resource.php?sessid=k27rds7h8w
```





# 3.4.4 Video – HTTP Cookies and Sessions

## HTTP Cookies & Sessions

In this video, you will see how the cookie protocol works (installation of cookies, manipulation of cookies, and dissecting a cookie, as well as how HTTP sessions work and how they use cookies).

You will see reference to a tool called Firebug. At the time of recording, it was an extension for the Iceweasel browser. Modern Kali Linux comes preinstalled with the Firefox browser instead of Iceweasel, but technically, it is almost the same browser as Iceweasel. Additionally, functionalities of firebug are now part of Developer Tools of Firefox, so instead of downloading the addon, you can simply turn on the Developer Tools by visiting Options -> Web Developer -> Toggle Tools, or just press Ctrl + Shift + I in your browser window.



*\*Videos are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the resources drop-down in the appropriate module line.*

# Same Origin Policy



## 3.5 Same Origin Policy

**Same Origin Policy** (SOP) is a critical point of web application security.

This policy prevents JavaScript code from getting or setting properties on a resource coming from a **different origin**.



## 3.5 Same Origin Policy

The browser uses:

Protocol

Hostname

Port

To determine if JavaScript can access a resource:  
*Hostname, port, and protocol must match.*

# EXAMPLE

# A JavaScript script on

`https://www.elearnsecurity.com:345/`

protocol                      hostname                      port

can read resources from:

- `https://www.elearnsecurity.com:345/path`
- `https://www.elearnsecurity.com:345/path/2`

## 3.5 Same Origin Policy

But not from:

- `https://www.elearnsecurity.com/path`  
(same protocol and domain but different port)
- `http://www.elearnsecurity.com:345/path`  
(same port and domain but different protocol)
- `https://www.heralab.net:345/path`  
(same port and protocol but different domain)



## 3.5.1 HTML Tags

Note that SOP applies only to the **actual code of a script**.

It is still possible to include external resources by using HTML tags like `img`, `script`, `iframe`, `object`, etc.





## 3.5 Same Origin Policy

The entire web application security is based on Same Origin Policy.

If a script on domain A was able to read content on domain B, it would be possible to steal clients' information and mount a number of very dangerous attacks.



# Burp Suite



## 3.6 Burp Suite

### How does this support my pentesting career?

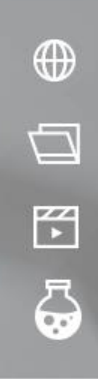
- Web application analysis
- Finding vulnerabilities
- Attacks
- Burp Suite is one of most used pentesting tools



## 3.6.1 Intercepting Proxies

An **intercepting proxy** is a tool that lets you analyze and modify any request, and any response exchanged between an HTTP client and a server.

By intercepting HTTP messages, a pentester can study a web application behavior and manually test for vulnerabilities.



## 3.6.1 Intercepting Proxies

The most used web application proxies are:

- The intercepting proxy feature of Burp Suite
- ZAP

Proxies are fundamental while analyzing web applications and will become your best friend for web-app testing.

## 3.6.1 Intercepting Proxies

Do not confuse intercepting proxies with common web proxy servers like [Squid](#). Proxy servers have different purposes: bandwidth optimization, content filtering and more.

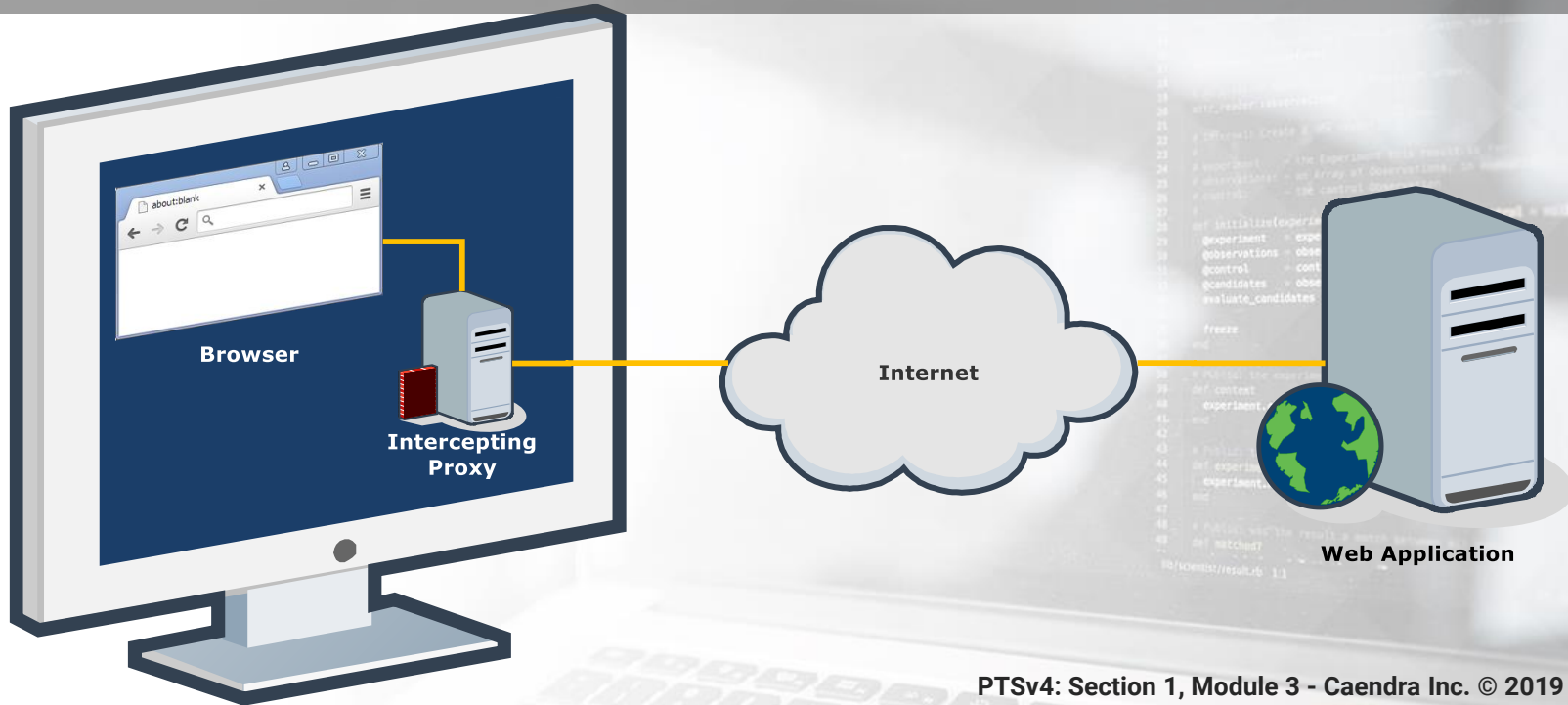
The next two images will make that clearer.





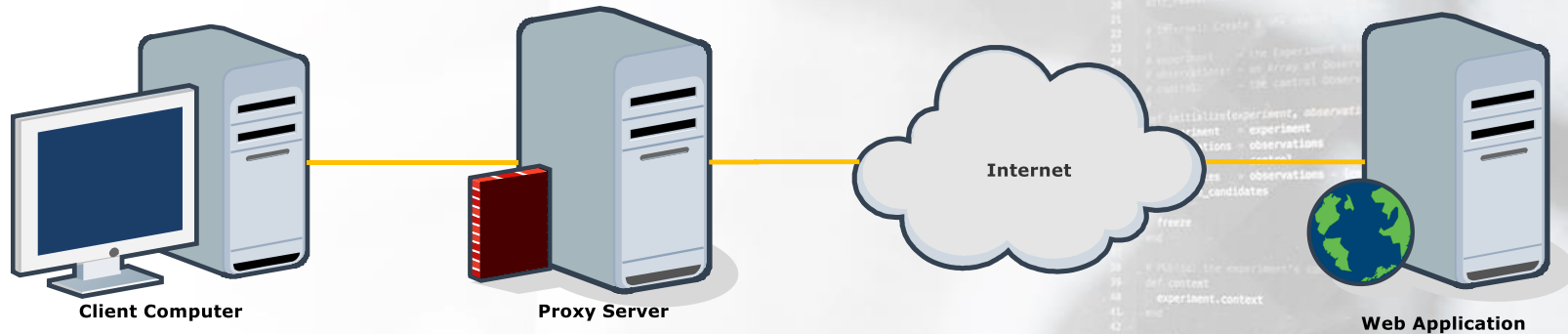
## 3.6.1.1 Intercepting Proxy Example

Here the proxy is an application which intercepts the penetration tester's browser traffic.



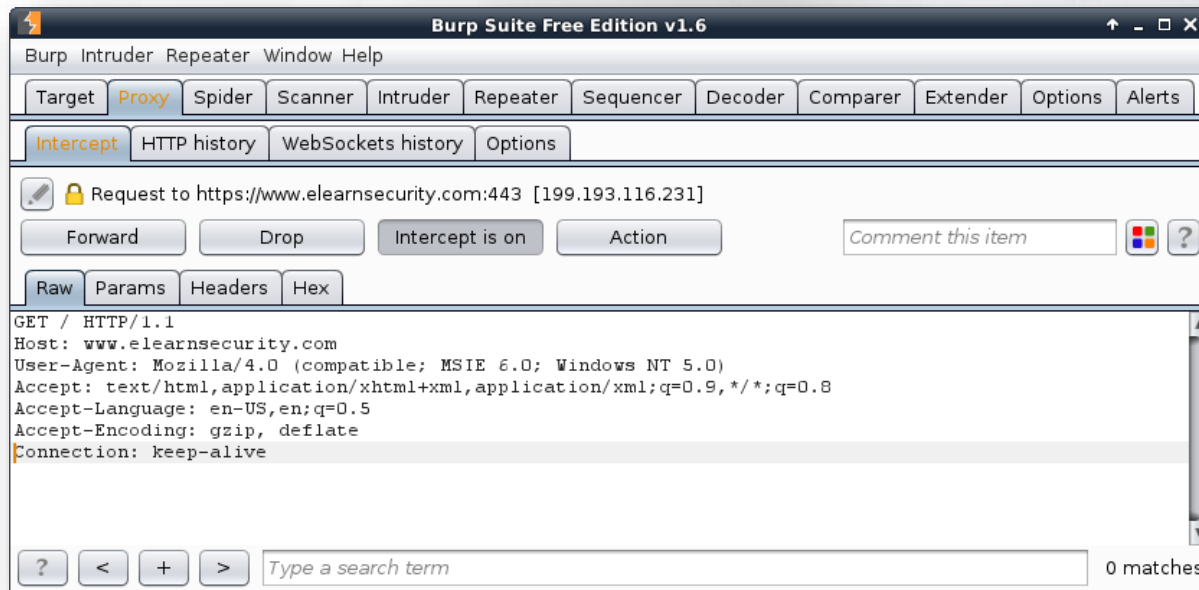
## 3.6.1.2 Proxy Server Example

Here the proxy server filters all the traffic coming from the internal network.



## 3.6.2 Burp Proxy

Burp suite offers one of the best proxies available. You can download the Free Edition [here](#).



## 3.6.2 Burp Proxy

Burp suite will let you:

- Intercept requests and responses between your browser and the web server.
- Build requests manually.
- Crawl a website by automatically visiting every page in a website.
- Fuzz web applications by sending them patterns of valid and invalid inputs to test their behavior.

## 3.6.2 Burp Proxy

By using Burp, you can **intercept and modify** requests coming from your browsers **before** they are sent to the remote server.

You can modify the **header** and the **body** of a message **by hand or automatically**.



## 3.6.2.1 Burp Proxy Configuration

In the following slides, you will see how to launch, configure and use Burp Suite with your browser.

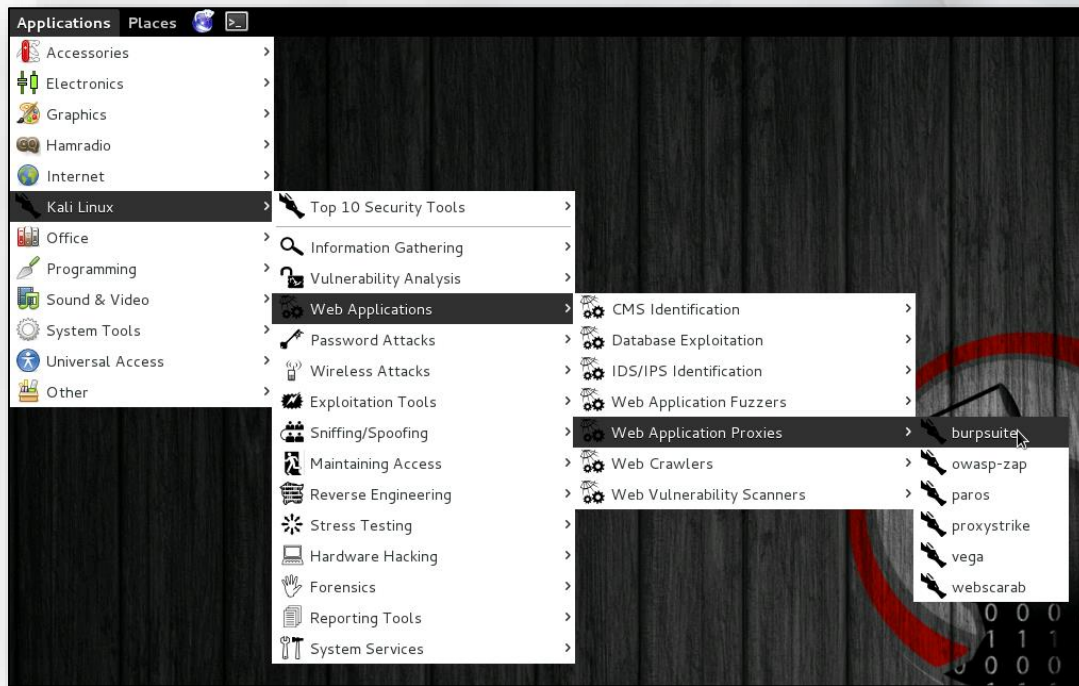
Try to understand all the settings by trying them on your computer!

```
23 # @param @observations - the Experiment observations to be evaluated
24 # @param @control - an array of Observations, in which the control is the first element
25 # @param @candidates - the control observation
26
27 def skillRanking(experiment, observations = [], control = null)
28   @experiment = experiment
29   @observations = observations
30   @control = control
31   @candidates = observations - [control]
32   evaluate_candidates
33
34   freeze
35   experiment.context
36 end
37
38 # @param @experiment - the name of the experiment
39 def experiment_name
40   @experiment.name
41 end
42
43 # @param @result - the result of a match between two experiments
44 def match?
45   @result == 1
46 end
```



## 3.6.2.1 Burp Proxy Configuration

**Launch Burp Suite:**  
In Kali you will find it under *Kali Linux > Web Applications > Web Application Proxies > burpsuite*.





## 3.6.2.1 Burp Proxy Configuration

If you want to run it on another operating system, you can download it from the [Portswigger website](https://portswigger.net/burp/communitydownload).

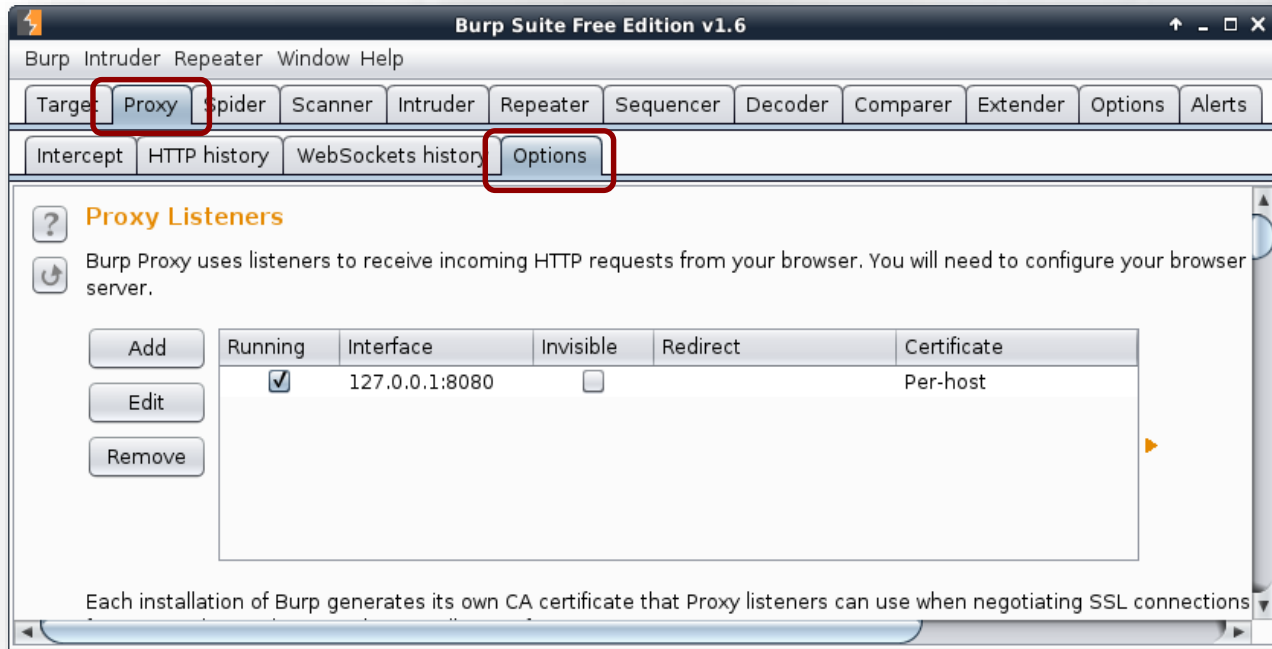
To run Burp, double click on the jar file you downloaded or run the below from the console:



```
java -jar burpsuite_free_v1.6.jar
```

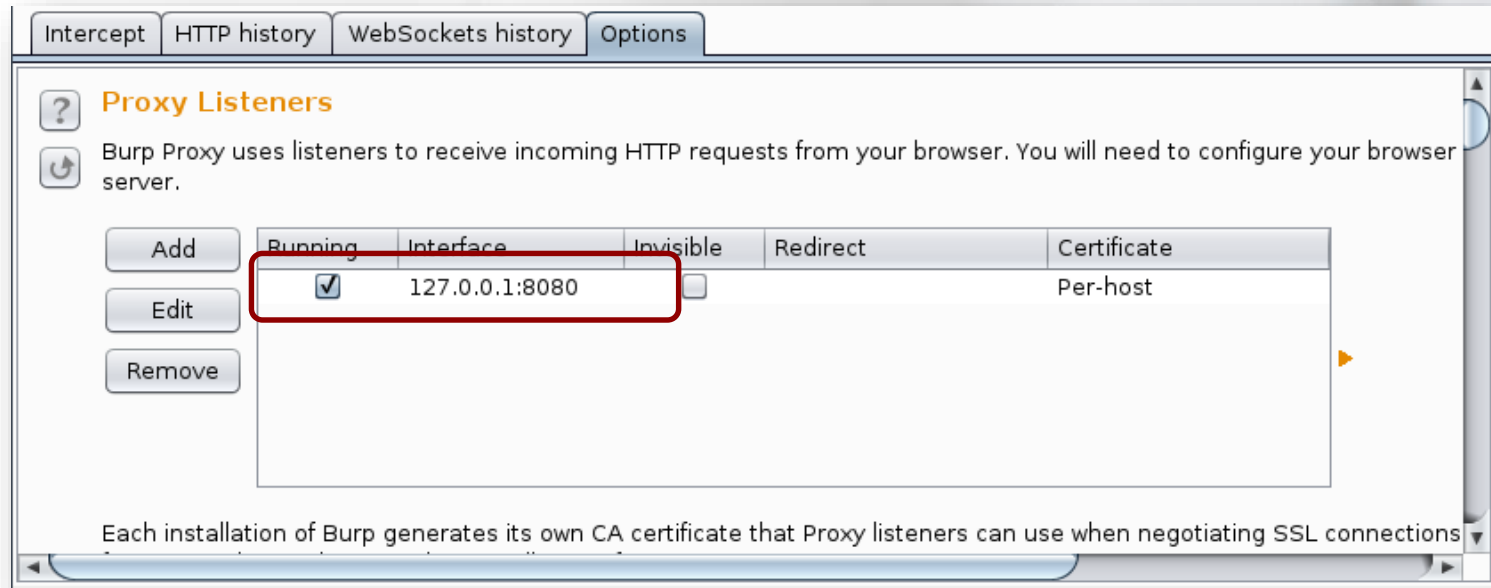
## 3.6.2.1 Burp Proxy Configuration

Now, go to the *Proxy* tab and then to the *Options* sub-tab.



## 3.6.2.1 Burp Proxy Configuration

Here you can start and stop the proxy and configure the *host:port* pair on which burp will listen.



## 3.6.2.1 Burp Proxy Configuration

Scrolling down you can find other configuration items to fine tune, which messages to intercept, how to automatically change message content and more.

For now, just leave the default options as they are, you will see how to use those features later on.



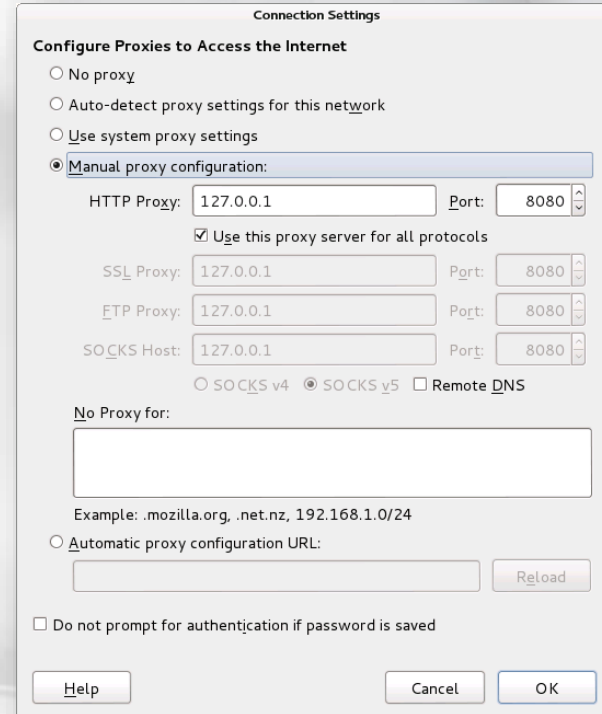
## 3.6.2.1 Burp Proxy Configuration

Once Burp Proxy is configured, you have to configure your browser to use it as the proxy for every protocol.



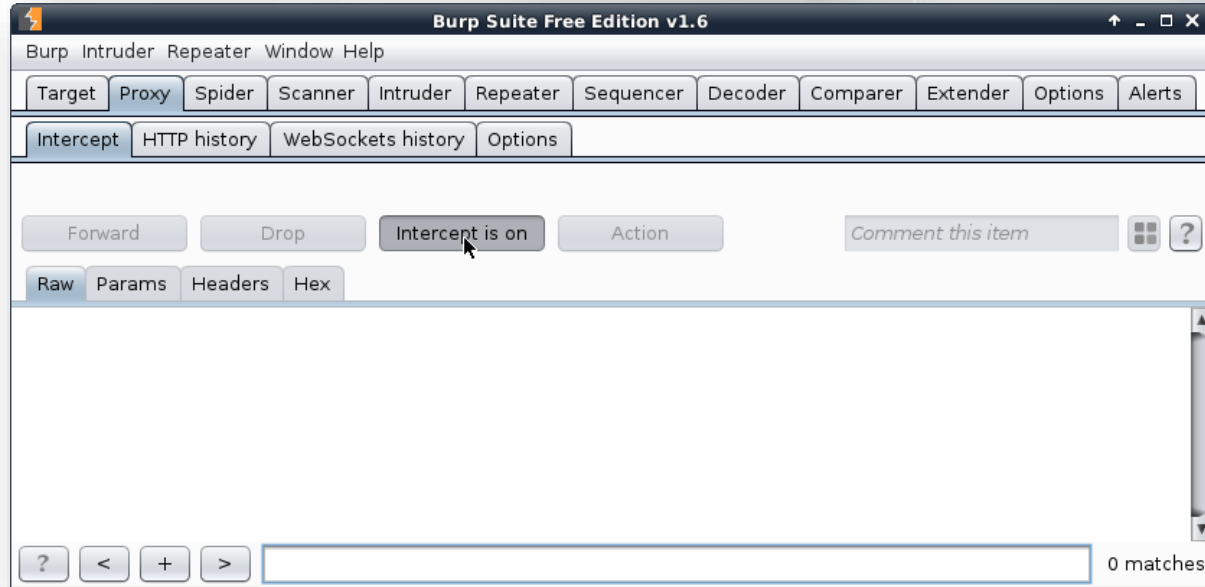
## 3.6.2.1 Burp Proxy Configuration

In Firefox, you have to open the *Preferences* window, go to the advanced tab, click on the *Network* sub-tab and finally open the *Connection settings* window.



## 3.6.2.1 Burp Proxy Configuration

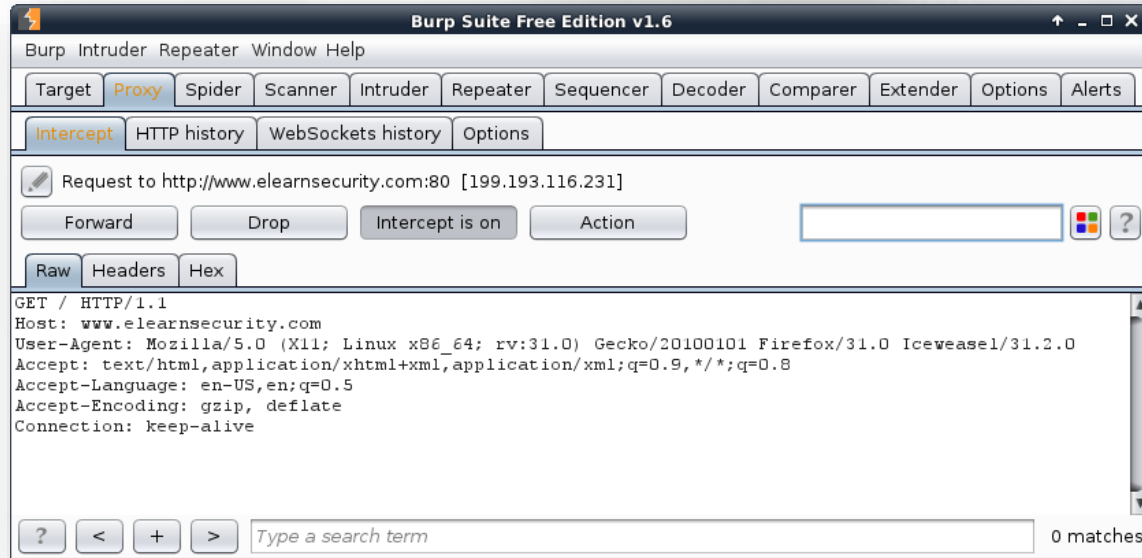
To intercept traffic, switch to Burp and go to *Proxy > Intercept* and click on the *Intercept is off* button to enable interception.





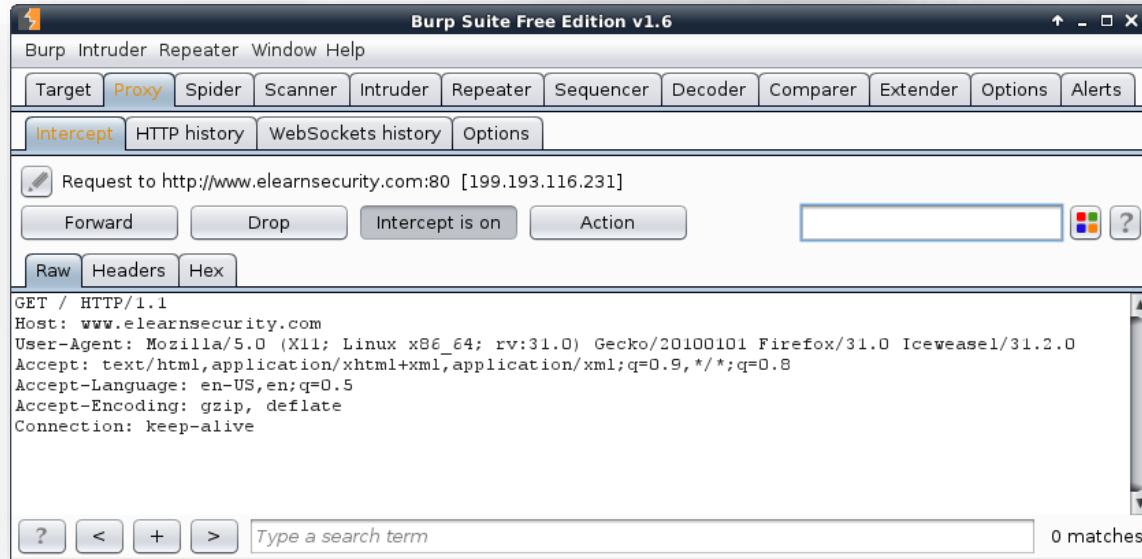
## 3.6.2.1 Burp Proxy Configuration

Now open a website with your browser; Burp will pop up intercepting the request. Optionally, you can modify and then forward it by clicking on *Forward*.



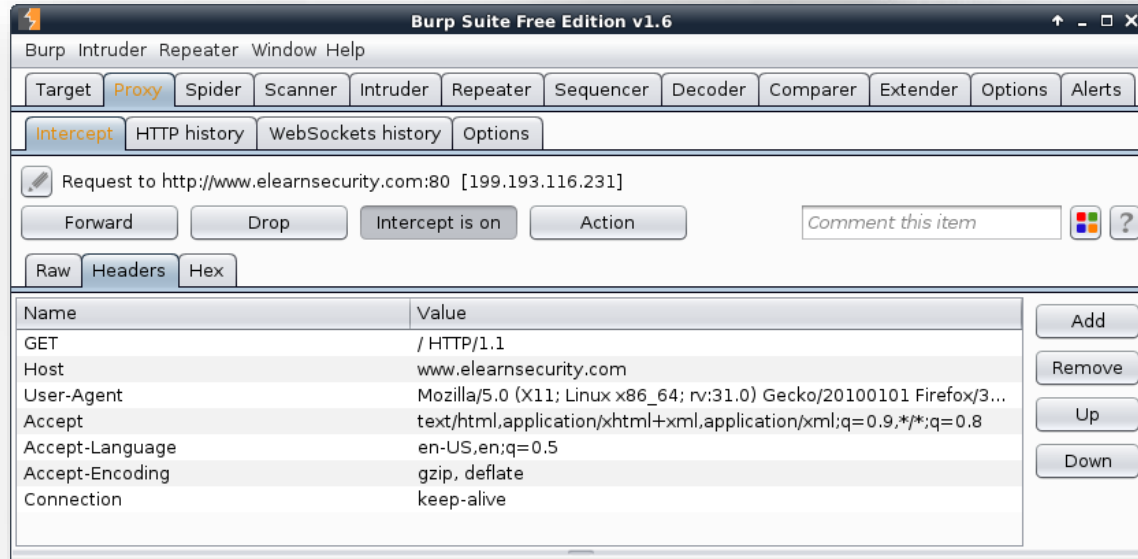
## 3.6.2.1 Burp Proxy Configuration

When *Intercept* is on, every browser request stops at Burp Proxy. You can modify the entire request or just its headers.



## 3.6.2.1 Burp Proxy Configuration

You can modify the headers both in the Raw tab or in the *Headers* tab. Remember to forward the request after editing it!



## 3.6.2.1 Burp Proxy Configuration

Q

*What is the difference between the Raw and the Headers tab?*

A

*They present the very same information with a different format. The Headers tab simply presents the headers in a tabular manner.*



## 3.6.2.1 Burp Proxy Configuration

You do not need to manually intercept and forward every request though. Even if you leave the master interception off, Burp will still collect information on the HTTP traffic coming to and from your browser.

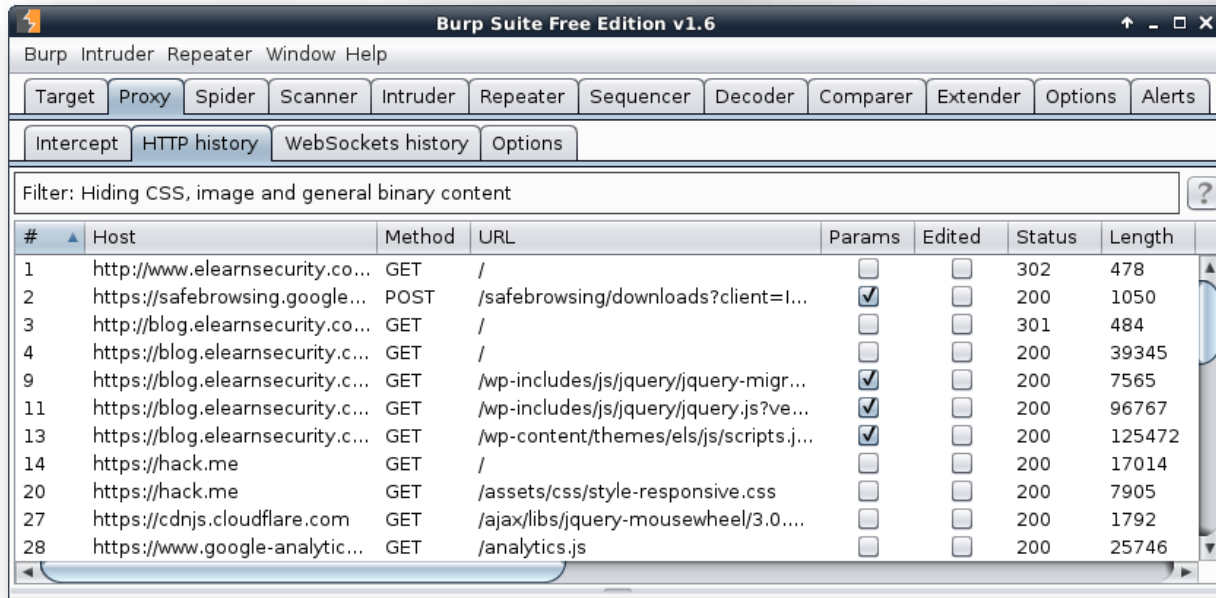
You can check what Burp is collecting in two ways:

- On the *Proxy > History* tab
- In the *Target > Site Map* tab

```
24 # Note that the Experiment is a list of Observations, in which
25 # observations is an array of Observations, in which
26 # control is the control Observation
27
28 def initialize(experiment, observations = [], control = nil)
29   @experiment = experiment
30   @observations = observations
31   @control = control
32   @candidates = observations - [control]
33   evaluate_candidates
34
35   freeze
36 end
37
38 # Note that the Experiment is a list of Observations, in which
39 # observations is an array of Observations, in which
40 # control is the control Observation
41
42 def experiment_name
43   # Return the name of the experiment
44   @experiment_name
45   experiment_name
46 end
47
48 # Return the result of a match between an observation and the control
49 def match?
50   # Return the result of a match between an observation and the control
51   @match?
52 end
```

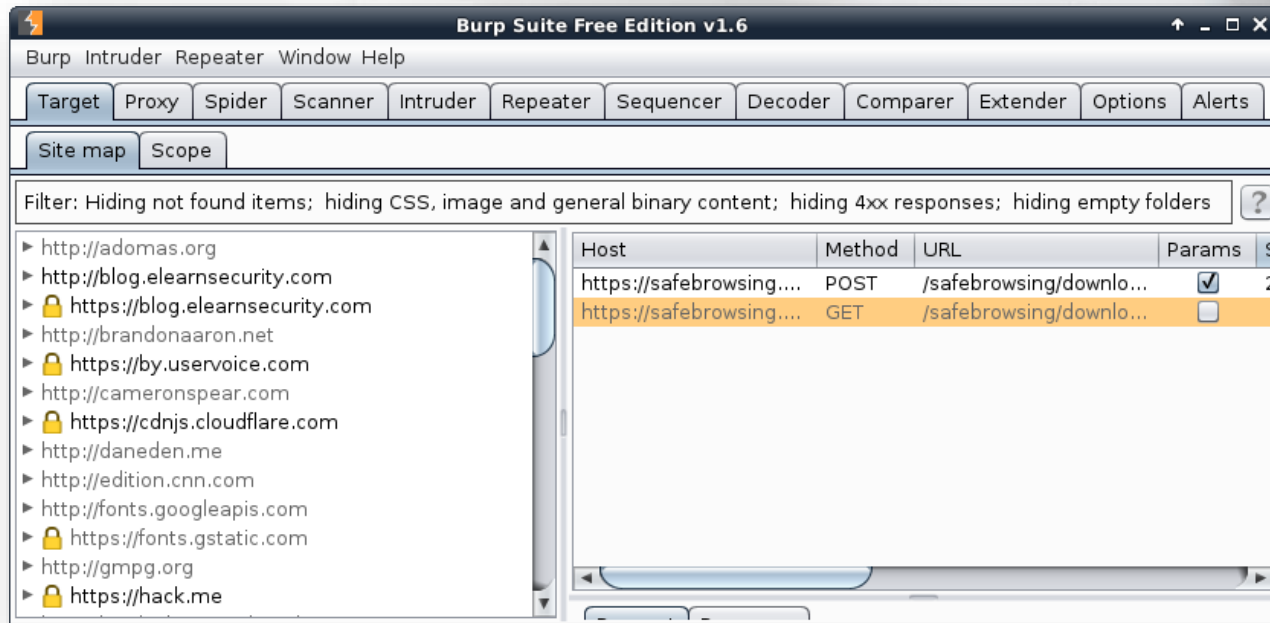
## 3.6.2.1 Burp Proxy Configuration

Burp Suite *Proxy* tab contains an *HTTP history* sub tab.



## 3.6.2.1 Burp Proxy Configuration

You can also check what Burp is collecting on *Target > Site Map*.





## 3.6.3 Burp Repeater

Another feature is **Burp Repeater**, which lets you manually build raw HTTP requests.

You can do the same thing by using other tools such as *netcat* or *telnet*, but *Burp* provides you:

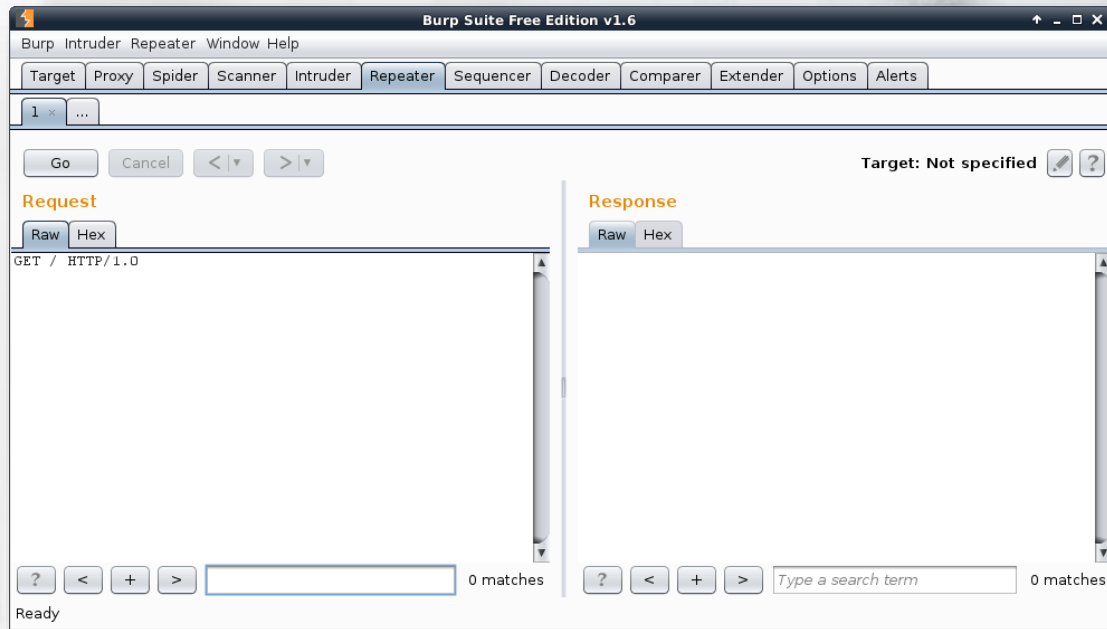
- Syntax highlighting
- Raw and rendered responses
- Integration with other Burp tools

```
31 def __init__(self, url, method, headers, body):
32     self.url = url
33     self.method = method
34     self.headers = headers
35     self.body = body
36
37     # Parse the request into a dictionary of components
38     self.parse()
39
40     # Create a new observation
41     self.observation = Observation(self.url, self.method, self.headers, self.body)
42
43     # Add the observation to the list of observations
44     self.observations.append(self.observation)
45
46     # Evaluate the candidates
47     self.evaluate_candidates()
48
49     # Freeze the results
50     self.freeze()
51
52     # Return the results
53     return self.results
54
55 def parse(self):
56     """Parse the request into a dictionary of components"""
57     # Parse the URL
58     self.url = urlparse(self.url)
59
60     # Parse the method
61     self.method = self.url.scheme
62
63     # Parse the headers
64     self.headers = {}
65     for header in self.url.headers:
66         self.headers[header.name] = header.value
67
68     # Parse the body
69     self.body = self.url.body
70
71 def freeze(self):
72     """Freeze the results"""
73     self.results = {}
74
75     # Add the URL to the results
76     self.results[self.url] = self.observations
77
78     # Add the method to the results
79     self.results[self.method] = self.observations
80
81     # Add the headers to the results
82     self.results[self.headers] = self.observations
83
84     # Add the body to the results
85     self.results[self.body] = self.observations
86
87     # Return the results
88     return self.results
89
90 def evaluate_candidates(self):
91     """Evaluate the candidates"""
92     # Create a new candidate
93     candidate = Candidate(self.url, self.method, self.headers, self.body)
94
95     # Add the candidate to the list of candidates
96     self.candidates.append(candidate)
97
98     # Evaluate the candidate
99     self.evaluate_candidate(candidate)
100
101     # Return the results
102     return self.results
103
104 def evaluate_candidate(self, candidate):
105     """Evaluate the candidate"""
106     # Create a new observation
107     observation = Observation(candidate.url, candidate.method, candidate.headers, candidate.body)
108
109     # Add the observation to the list of observations
110     self.observations.append(observation)
111
112     # Evaluate the candidate
113     self.evaluate_observation(observation)
114
115     # Return the results
116     return self.results
117
118 def evaluate_observation(self, observation):
119     """Evaluate the observation"""
120     # Create a new candidate
121     candidate = Candidate(observation.url, observation.method, observation.headers, observation.body)
122
123     # Add the candidate to the list of candidates
124     self.candidates.append(candidate)
125
126     # Evaluate the candidate
127     self.evaluate_candidate(candidate)
128
129     # Return the results
130     return self.results
```



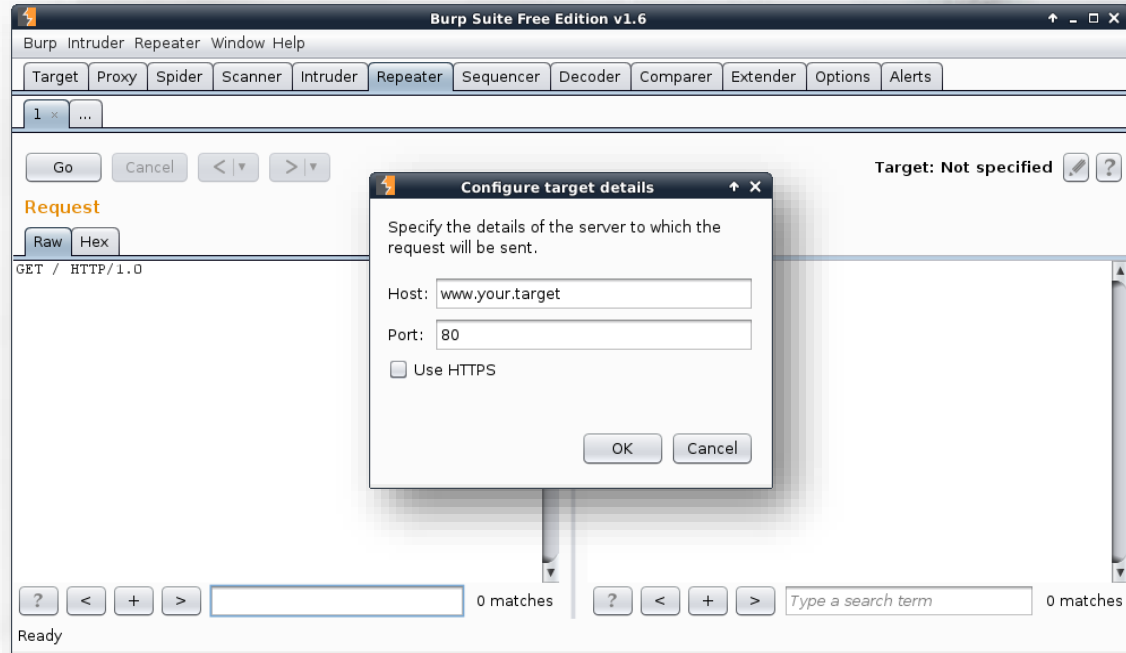
## 3.6.3 Burp Repeater

To create a request, first set your target by clicking on the pencil icon in the upper right corner of the tab.



## 3.6.3 Burp Repeater

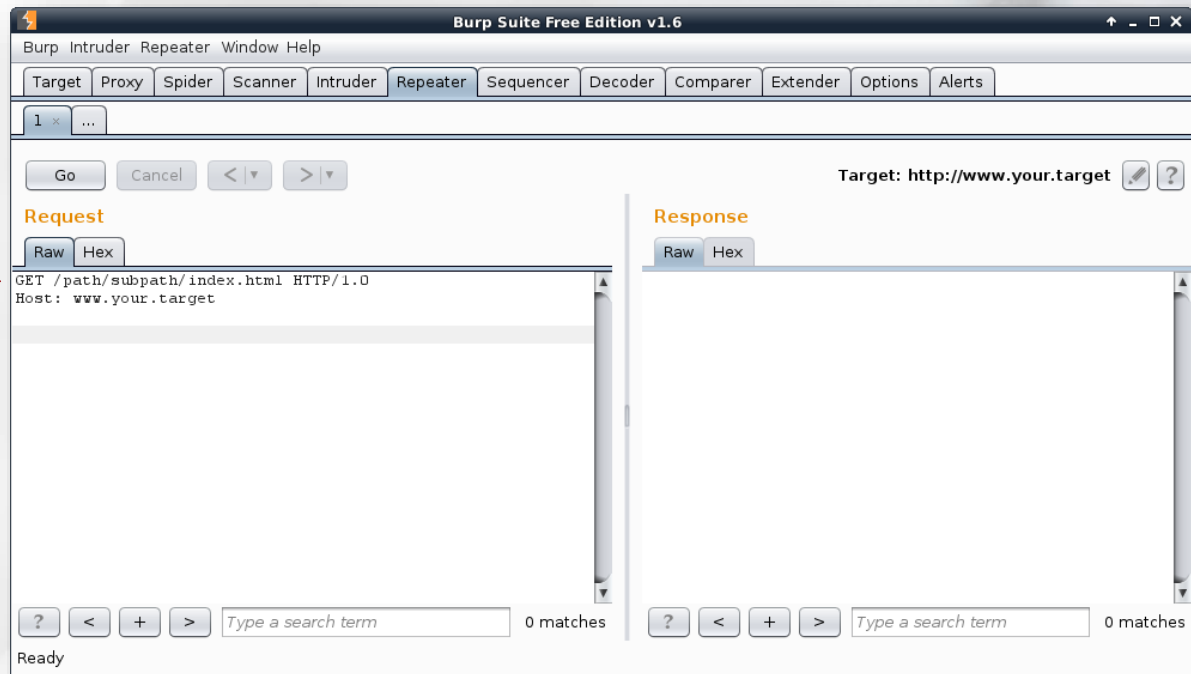
You can then set your target host and port.



## 3.6.3 Burp Repeater

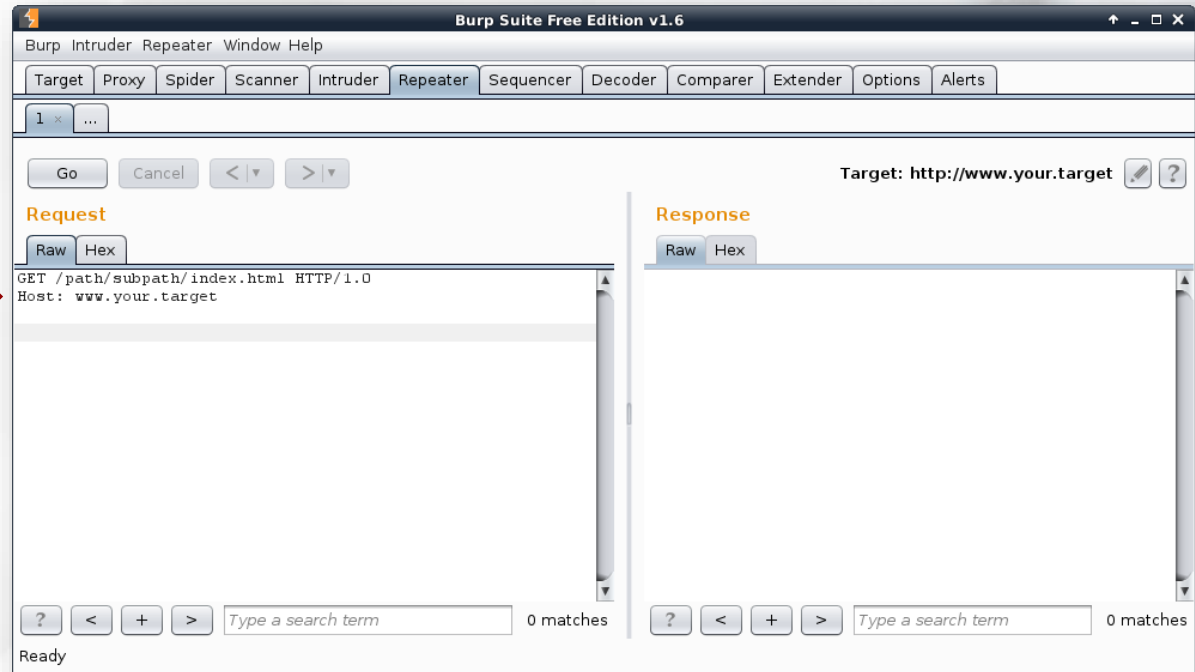
You can define your request by using this text area.

Every request must have at least an **HTTP VERB** (GET, POST, HEAD, ...)



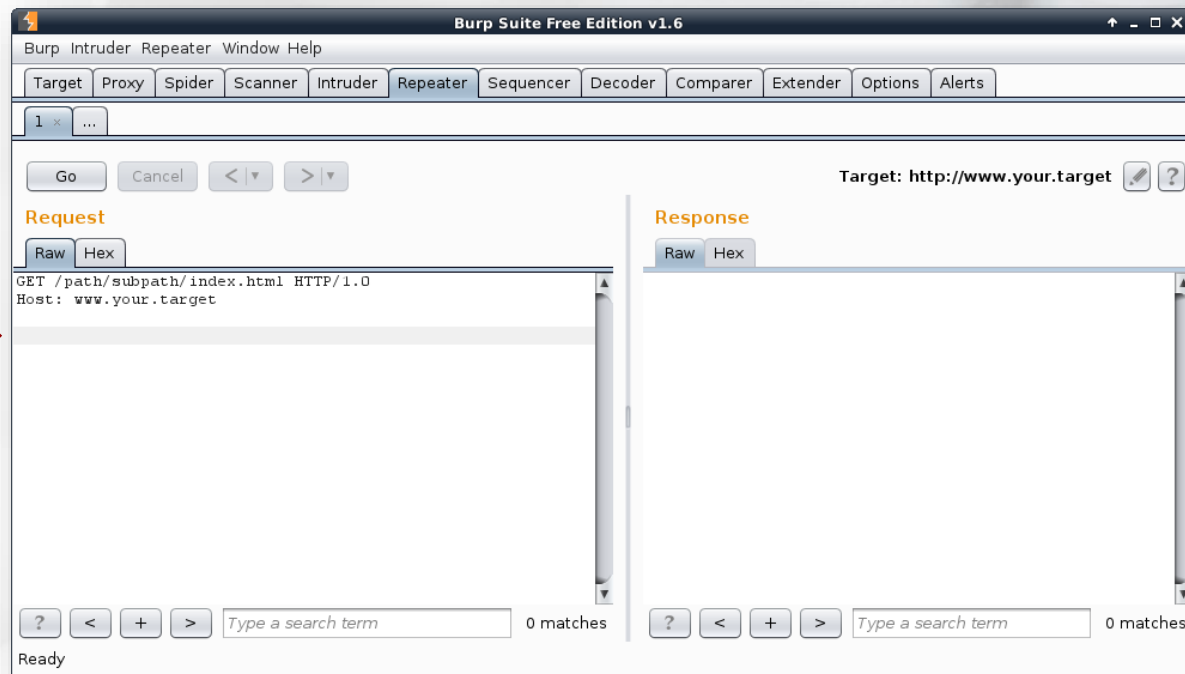
## 3.6.3 Burp Repeater

Here is the  
**Host** header.



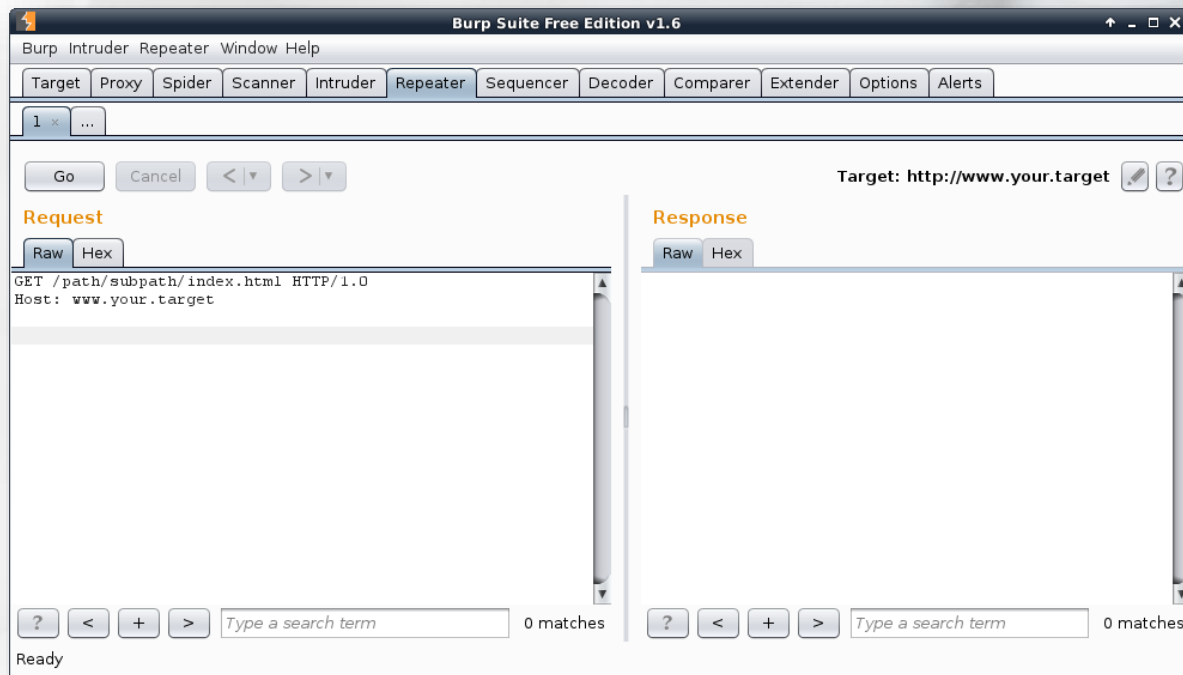
## 3.6.3 Burp Repeater

And here we see **two empty lines** after the headers.



## 3.6.3 Burp Repeater

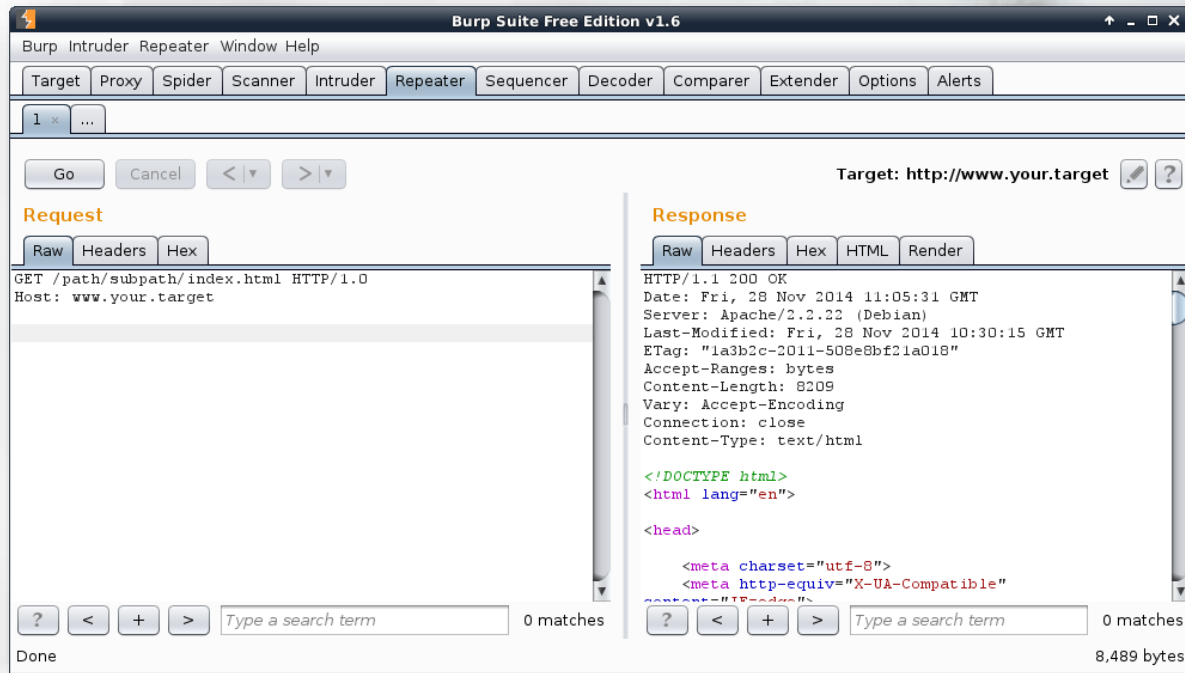
When your request is complete, you can click the **Go** button to send it to the server.





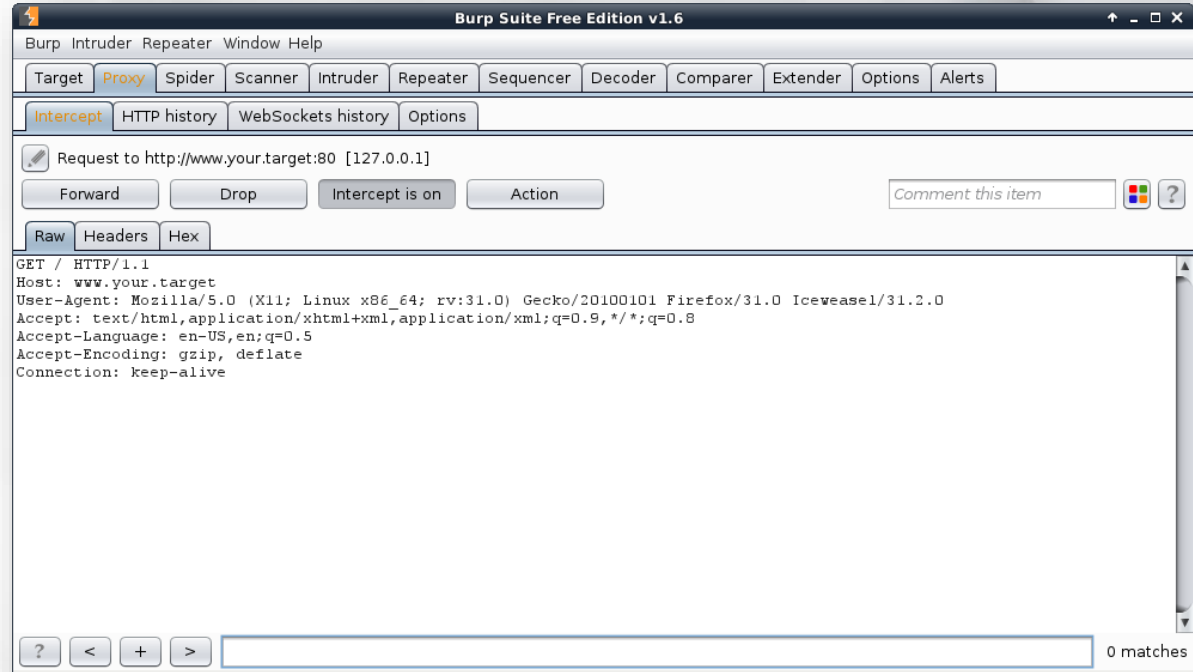
## 3.6.3 Burp Repeater

Burp will then display the response in the **Response** panel.



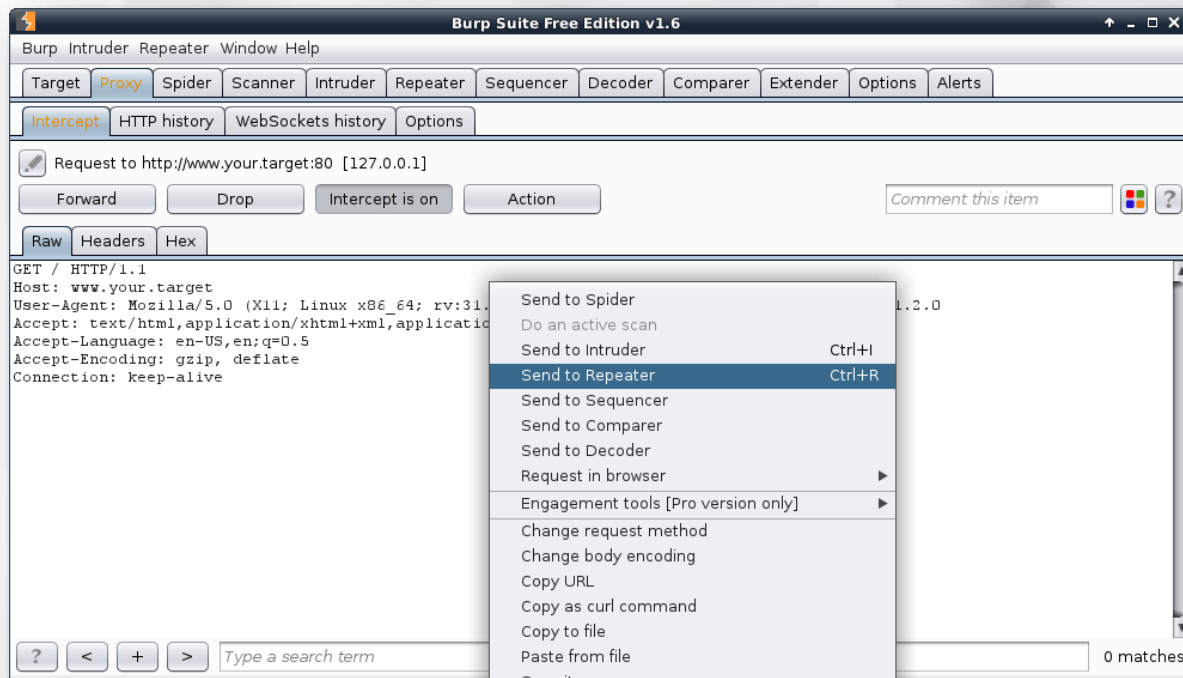
## 3.6.3 Burp Repeater

An easier method to build requests is to intercept a browser request with the proxy and send it to the Repeater function.



## 3.6.3 Burp Repeater

You can do that with the **Ctrl+R** shortcut or by right-clicking in the request body and selecting **Send to Repeater**.



## 3.6.4 Video – Burp Suite

### Burp Suite

In this video, you will learn how to define the target scope in Burp and how to configure the Proxy to intercept only the traffic in scope.

Moreover, you will see how to configure and use other features of Burp according to the penetration test engagement goals.



*\*Videos are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the resources drop-down in the appropriate module line.*

### 3.6.5 Hera Lab – Burp Suite Basics

Try to test your knowledge with a real task to be done with help of Burp Suite.

The task you are about to face is very similar to what pentesters do everyday when inspecting a web application using Burp Suite.

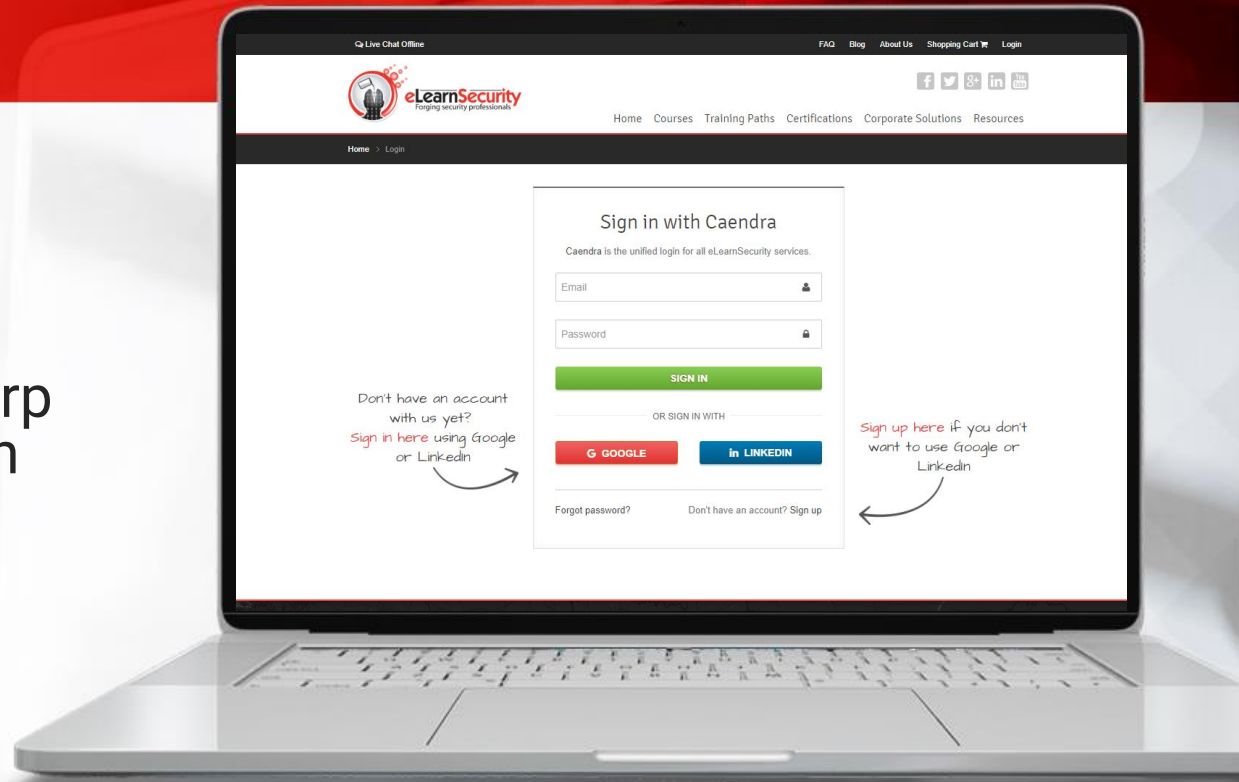


## 3.6.5 Hera Lab – Burp Suite Basics

### Burp Suite Basics

In this lab you will:

- Inspect a web application using Burp Suite and find hidden resources.
- Use Burp Intruder to attack a web application



*\*Labs are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the labs drop-down in the appropriate module line or to the virtual labs tabs on the left navigation.*

## 3.6.6 Hera Lab – Burp Suite

The best way to learn how to use a tool is through practice. Analyze a web application with Burp and hack it!

Try to reach the lab goal by yourself. If you really get stuck, you can always check the solutions in the lab manual.



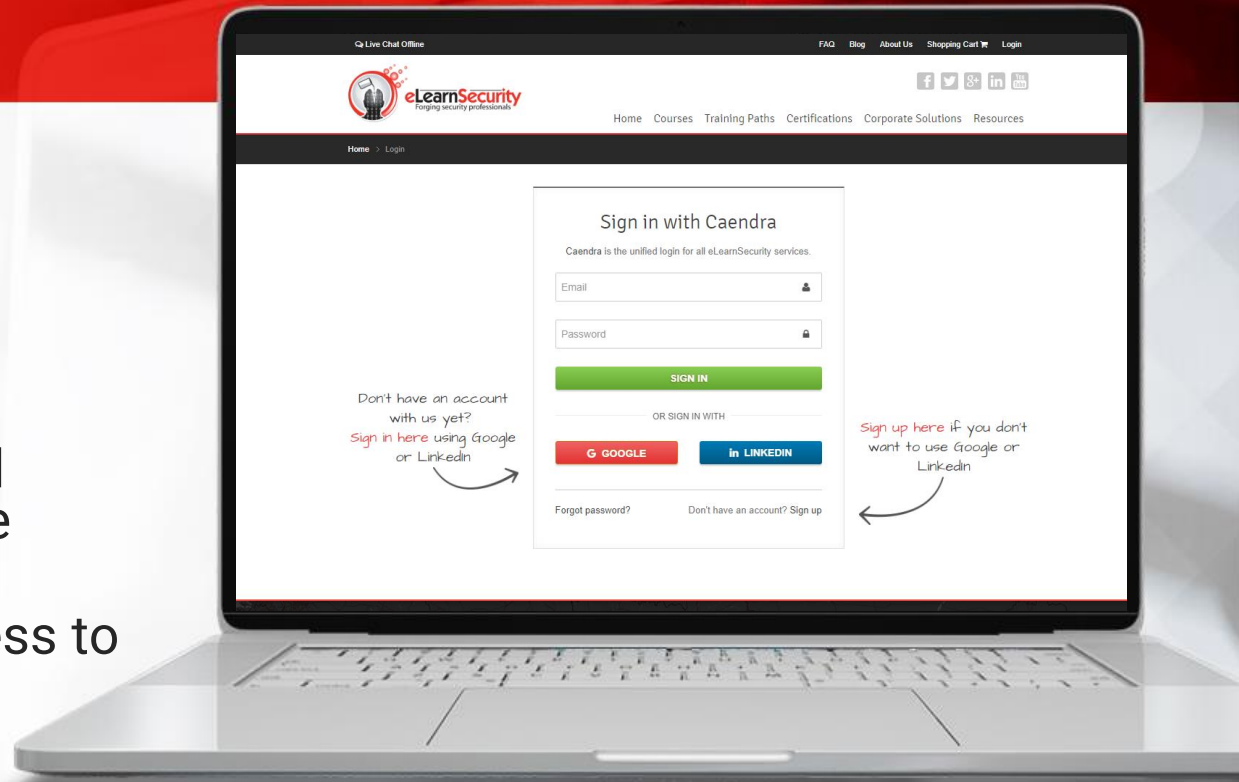


## 3.6.6 Hera Lab – Burp Suite

### Burp Suite

In this lab you will:

- Configure Burp Suite
- Crawl a web site
- Intercept requests and responses to study the web application
- Get unauthorized access to the development area



*\*Labs are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the labs drop-down in the appropriate module line or to the virtual labs tabs on the left navigation.*

# References



# References

## [URIs, URLs, and URNs: Clarifications and Recommendations 1.0](http://www.w3.org/TR/uri-clarification/)

<http://www.w3.org/TR/uri-clarification/>

## [SSL/TLS Strong Encryption: An Introduction](http://httpd.apache.org/docs/2.2/ssl/ssl_intro.html)

[http://httpd.apache.org/docs/2.2/ssl/ssl\\_intro.html](http://httpd.apache.org/docs/2.2/ssl/ssl_intro.html)

## [HTTP State Management Mechanism \(cookies\)](http://tools.ietf.org/html/rfc6265)

<http://tools.ietf.org/html/rfc6265>

## [ZAP](https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)

[https://www.owasp.org/index.php/OWASP\\_Zed\\_Attack\\_Proxy\\_Project](https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)



# References

## [HTTP Status Code Reference](http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html)

<http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

## [HTTP Overview, History, Versions and Standards](http://www.tcpipguide.com/free/t_HTTPOverviewHistoryVersionsandStandards.htm)

[http://www.tcpipguide.com/free/t\\_HTTPOverviewHistoryVersionsandStandards.htm](http://www.tcpipguide.com/free/t_HTTPOverviewHistoryVersionsandStandards.htm)

## [Burp Suite](https://portswigger.net/burp/communitydownload)

<https://portswigger.net/burp/communitydownload>

## [Portswigger Burp Suite Editions](http://portswigger.net/burp/)

<http://portswigger.net/burp/>



# References

## Squid: Optimising Web Delivery

<http://www.squid-cache.org/>

## HTTP/1.X

<https://hpbn.co/http1x/>



# Videos

## HTTP and HTTPS Protocols Basics

In this video, you will see HTTP and HTTPS in action both via command line tools or with Burp Proxy. You will learn how to perform requests, receive responses and analyze HTTP(s) messages both manually and with an automated tool. Additionally, you will learn how to use Burp at the end of this module.

## HTTP Cookies and Sessions

In this video, you will see how the cookie protocol works; specifically, installation of cookies, manipulation of cookies, and dissecting a cookie. Moreover, you will see how HTTP sessions work and how they use cookies.

*\*Videos are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the resources drop-down in the appropriate module line.*

# Videos

## Burp Suite

In this video, you will learn how to define the target scope in Burp and how to configure the Proxy to intercept only the traffic in scope. Moreover, you will see how to configure and use other features of Burp according to the penetration test engagement goals.

*\*Videos are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the resources drop-down in the appropriate module line.*



# Labs

## Burp Suite Basics

Learn how to use Burp Intruder and Repeater in order to attack a web application. In this lab you will use applications' own resources in order to hack it using Burp Intruder.

## Burp Suite

Learn how to use Burp by hacking a web application! In this lab you will configure Burp Suite, crawl a web site, intercept requests and responses to study the web application, and get unauthorized access to the development area.

*\*Labs are only available in Full or Elite Editions of the course. To upgrade, click [HERE](#). To access, go to the course in your members area and click the labs drop-down in the appropriate module line or to the virtual labs tabs on the left navigation.*