المقدمة

المؤلف

د.م/ أحمد هاشم الفقي

استشارى أمن المعلومات و التحول الرقمي

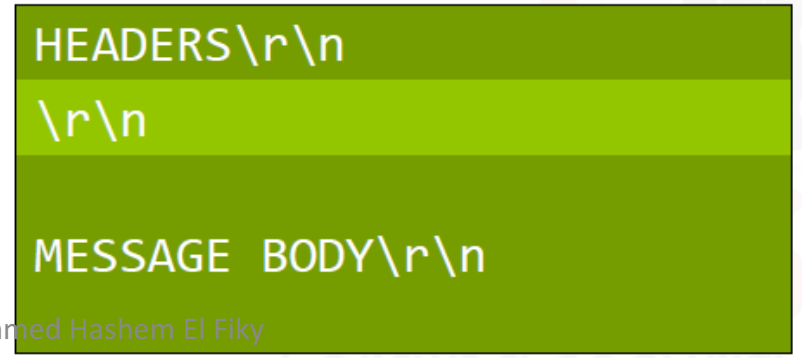
Ahmed.Hashem.ElFiky@outlook.com

محتويات هذا الفصل:

- اساسيات بروتوكول HTTP/S
- فهم الترميز Encoding
- فهم سياسة نفس الاصل Same Origin Policy
- فهم ال Cookies
- فهم ال Session
- فهم Web Application Proxies

أساسيات بروتوكول HTTP/S

- بروتوكول ال HTTP هو اختصار لكلمة Hyper Test Transfer Protocol
- وهو برتوكول اساسى للتصفح على الويب او الانترنت
- و هو عبارة عن Client-Server Protocol حيث يطلب المستخدم او ال Client صفحة الويب من على المتصفح ثم ياتى الرد من على الخادم او ال Server و يقوم هذا للبروتوكول بعرضها على المتصفح فهو يستخدم فى الاتصال بين المستخدم و الخادم و تبادل الرسائل (requests/responses) بينهما
- يتكون HTTP Request من مجموعة من Headers و ال Message Body كما وضح بالشكل التالى: حيث (r\n\) تساوى ضربة Enter من الكيبورد



أساسيات بروتوكول HTTP/S (تكلمة....)

- المتصفح يرسل من مجموعة ال Headers تتألف من كذا Parameter الى الخادم كما بالشكل الاتى:


```
GET / HTTP/1.1
Host: www.google.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64;
rv:36.0) Gecko/20100101 Firefox/36.0
Accept: text/html,application/xhtml+xml
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

أساسيات بروتوكول HTTP/S (تكلمة....)

- ف أول Parameter هوال Request method وهو فى المثال السابق كلمة GET التى تعنى ان المستخدم يريد الحصول على شئ ما من الخادم. كما يوجد انواع اخرى من Request Method مثل POST, PUT, DELETE, OPTIONS, TRACE و غيرهما
- ثانى Parameter هو ال Path اى المسار التى تريده من الخادم و هو فى المثال السابق (/)
- ثالث Parameter هو البروتوكول المستخدم وهو فى المثال السابق HTTP/1.1
- رابع Parameter و هو ال Host اى عنوان website الى تريده. ملحوظة عنوان website + المسار Full URL = (Path)
- خامس Parameter وهو ال User-Agent حيث يحتوى على اسم المتصفح وال version الخاص به و نوع نظام التشغيل و هو فى المثال السابق اسم المتصفح Firefox و نظام التشغيل windows

أساسيات بروتوكول HTTP/S (تكلمة....)

- سادس Parameter هو ال Accept و هنا يحدد المتصفح نوع الملفات التى سوف ترجع من الخادم ثم يقوم بعرضها للمستخدم و هو فى المثال السابق نوع الملفات التى سيقوم بعرضها HTML
- سابع Parameter هو ال Accept-Encoding و هنا يحدد المتصفح نوع ضغط الملفات المقبول و الذى سوف يرجع من الخادم بدون فقدان للمعلومات ثم يقوم المتصفح بفك الضغط و عرض الملفات للمستخدم و هو فى المثال السابق نوع ضغط الملفات المقبول من المتصفح هو gzip
- ثامن Parameter هو ال Connection وهو يستخدم للحفاظ على الاتصال بين المستخدم و الخادم لوقت طويل لو استخدم كلمة Keep-alive كما هو فى المثال السابق بدلا من إنشأ اتصال جديد بين المستخدم و الخادم كل مره كما هو فى الحال فى بوتوكول HTTP 1.0

أساسيات بروتوكول HTTP/S (تكلمة....)

- الخادم يرسل من مجموعة ال Headers تتألف من كذا سطر الى المتصفح كما بالشكل الاتى:

```
HTTP/1.1 200 OK
Date: Fri, 13 Mar 2015 11:26:05 GMT
Cache-Control: private, max-age=0
Content-Type: text/html; charset=UTF-8
Content-Encoding: gzip
Server: gws
Content-Length: 258

<PAGE CONTENT>
```

أساسيات بروتوكول HTTP/S (تكلمة....)

- ف أول سطر هو ال Status line حيث يتكون من البرتوكول المستخدم و ال version الخاص به ثم كود الرد او ما يعرف ب Status Code ثم المعنى المراد لهذا الكود كما موضح بعض الاكواد و المعنى المراد لها فى الجدول الاتى:

كود الرد	المعنى المراد
200	ان المصدر الذى طلبته موجود
301	ان ال مصدر الذى طلبته تم انتقله للينك جديد بشكل دائم
302	ان المصدر الذى طلبته تم انتقله للينك جديد بشكل مؤقت
403	ان المستخدم لا يمتلك الصلاحيات الكافيه للحصول على هذا المصدر من الخادم
404	ان الخادم لا يحتوى على هذا المصدر الذى يطلبه المستخدم

أساسيات بروتوكول HTTP/S (تكلمة....)

- ثانى سطر هو التاريخ او ال Date و هو يعرض التاريخ و الوقت للرسالة او الرد الذى انشأت من ناحية الخادم على طلب المستخدم
- ثالث سطر هو ال Cache-control حيث يسمح لكل من المتصفح و الخادم على الموافقة على بعض قواعد ال Caching كما فى الجدول التالى:

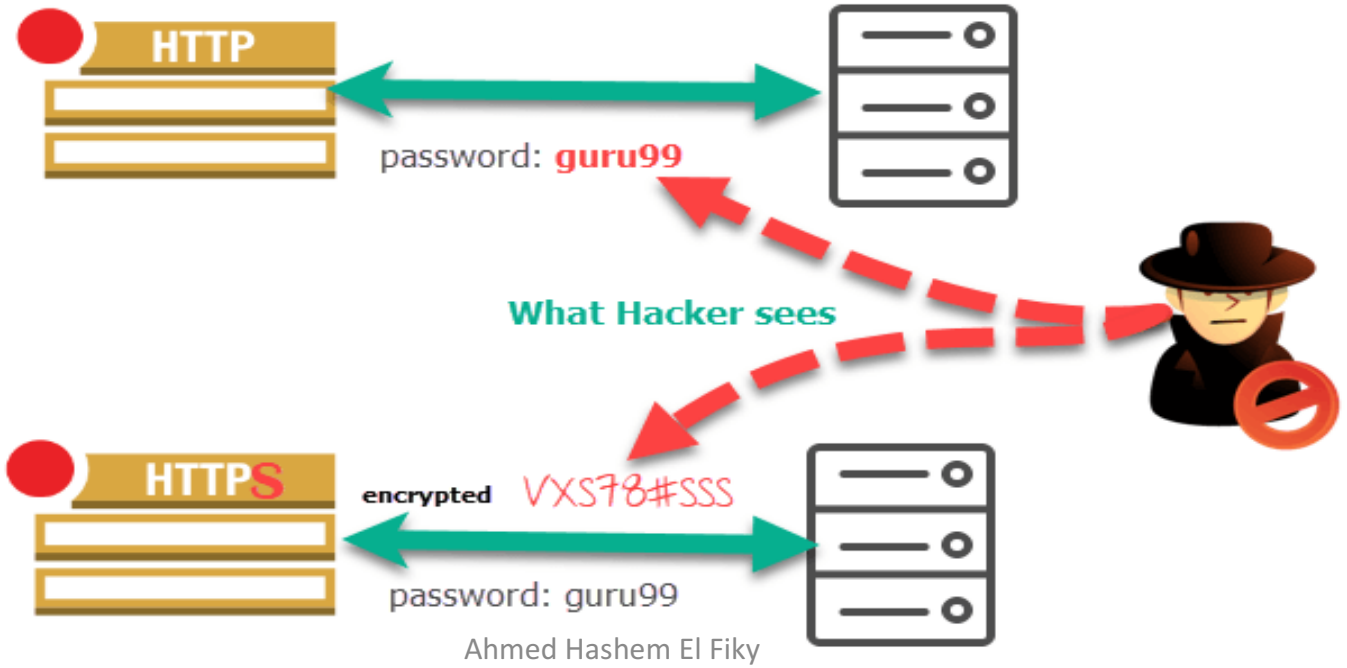
Caching Rule	المعنى المراد
no-cache	يحدد هذا الامر على انه يجب التحقق من أى محتوى مخزن مؤقتا عند كل طلب قبل تقديمه الى العميل
no-store	يشير هذا الامر الى انه لا يمكن التخرين المؤقت للمحتوى بأى طريقة
public	يقوم هذا الامر بتحديد نوع المحتوى على انه عام مما يعنى انه يمكن التخزين المؤقت له من قبل المتصفح او من قبل اى تخزينات مؤقته وسيطة
Private	يقوم هذا الامر بتحديد نوع المحتوى على انه خاص مما يعنى انه يمكن التخزين المؤقت له من قبل المتصفح فقط

أساسيات بروتوكول HTTP/S (تكلمة....)

- رابع سطر هو ال نوع المحتوى الذى سوف يعرض على متصفح المستخدم او ما يسمى بال Content-type
- خامس سطر هو ال Content-Encoding يوضح نوع الضغط المقبول للرسالة المرسلة من الخادم للمتصفح حيث فى مثالنا السابق كان نوع الضغط هو gzip
- سادس سطر هو ال Server حيث يعرض معلومات عن الخادم و فى مثالنا السابق هو gws او Google Web Server
- سابع سطر هو طول المحتوى او الرسالة او ما يعرف باسم Content-Length و الوحدة المستخدمة هى ال byte
- ثامن سطر هو محتوى الرسالة المرسلة من الخادم الى المتصفح للعرض على المستخدم

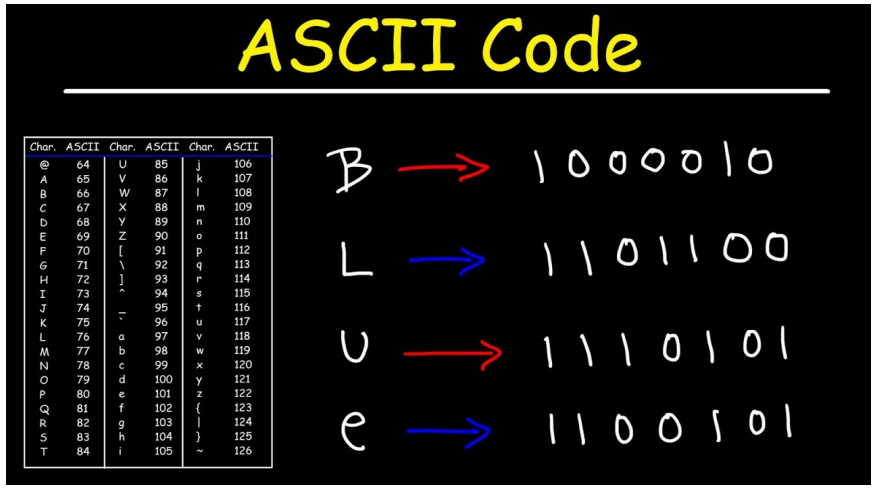
الفرق بين بروتوكول HTTP و بروتوكول HTTPS

- بالنسبة لبروتوكول ال HTTP ترسل البيانات بين المتصفح و الخادم او العكس بدون تشفير
- اما بالنسبة لبرتوكول ال HTTPS ترسل البيانات بين المتصفح و الخادم او العكس بشكل مشفر يصعب فهمه حيث تستخدم طبقة للشفير اسمها SSL/TLS



فهم ال Encoding او ما يعرف بالترميز

- يهدف الترميز إلى تحويل بيانات ليصبح بإمكان أنظمة مختلفة التعامل معها بطريقة صحيحة. على سبيل المثال: إرسال ملفات تنفيذية في بريد إلكتروني أو عرض حروف Charactersخاصّة على صفحة ويب. **ليس** الغرض هنا إبقاءَ المعلومة سريّة بل التأكد من أن التعامل معها سيكون على النحو الأمثل.
- يحوّل الترميز البيانات من صيغة إلى أخرى بآلية ***متاحة للعموم*** ويمكن بالتالي عكسُ التحويل بسهولة. لا تحتاج البيانات بعد ترميزها لمفتاح سري حتى يمكن التعامل معها، إذ أن المطلوب الوحيد ليمكنَ فك الترميز هو الخوارزمية Algorithmالمستخدمة فيه
- أمثلة: ASCII و Unicode و URL و Base64



فهم سياسة نفس الاصل Same Origin Policy

- تعريف 1: هذه السياسة تسمح للأكواد النصية ان تعمل فقط على صفحات الموقع الواحد والموقع يتحدد عن طريق نوع البرتوكول واسم المضيف ورقم المنفذ
- تعريف 2: هو حماية متواجدة في كل المتصفحات وظيفتها تمنع الموقع B بأنه يقرأ Response Data موقع A اذا لم تتطابق الشروط الثلاث بين A و B وهي (اسم الدومين/بورت او منفذ الدومين/نوع البرتوكول المستخدم)
- مثال: لدينا موقعين الاول اسمة https://Boom.com و الثانى اسمة https://Null.com حيث ان موقع Null.com يحتوى على API وظيفته استخراج معلومات العميل و ارسالها لموقع Boom.com إذن تعالوا نشوف هل المتصفح سوف يسمح بذلك حيث انه سوف يسمح بذلك فى حالة إذا تطابقت الشروط الثلاثة السابقة و لن يسمح إذا لم تطابق اى واحد من الشروط الثلاثة السابقة

فهم سياسة نفس الاصل Same Origin Policy (تكمله...)

A Site	B Site	نتيجة المقارنة
//:https	//:https	تطابقت بنجاح (URI Scheme)
Boom.com	Null.com	فشل التطابق (Domain/اسم الموقع)
443	443	تطابقت بنجاح (Port/المنفذ)

- للأسف ان (اسم الموقع) لم يتطابق فالمتصفح بحماية SOP سوف يقف عائق للمبرمج المسكين :(!! حاليا المبرمج في حيرة ويحتاج ان يريد موقع Boom.com معرفة معلومات العميل من ال Null.com (API) فما الحل !؟ هنا يأتي دور مايسمى CORS
- CORS هى اختصار لكلمة Cross-Origin Resource Sharing

فهم سياسة نفس الاصل Same Origin Policy (تكمله...)

- تعريف CORS هو المسؤول عن السماح ورفض تمرير وقراءة Response data بين موقع A وموقع B
- لنكمل ماتوقفنا عنده في جزئية شرح SOP حيث ال CORS هو بالواقع عبارة عن Headers Response فمن أجل ان يستطيع موقع Boom.com من اخذ Response Data من موقع Null.com (API) لازم ان نعرف ان فيه بعض Headers Response لازم يتم تعيينها في موقع Null.com
- اول Response Header اسمه Access-Control-Allow-Origin سوف يتم وضعه ضمن ال Headers الخاص بموقع Null.com حيث نضع فيه اسم الموقع الذى يسمح له Null.com بأخذ معلومات منه و هو فى مثالنا https://Boom.com
- ثانى Response Header اسمه Access-Control-Allow-Credentials و هو قيمته True/False فإذا كانت True فإننا نسمح للموقع Boom.com بأخذ المعلومات من موقع Null.com و إذا كانت False فإننا لن نسمح للموقع Boom.com بأخذ المعلومات من موقع Null.com

فهم سياسة نفس الاصل Same Origin Policy (تكمله...)

- على سبيل المثال ده شكل Request من نوع GET حيث يريد موقع platform.twitter.com بأخذ معلومات من موقع Syndication.twitter.com حيث تم وضع الموقع الذى ياخذ المعلومات فى parameter اسمه Origin فهل يسمح له الموقع Syndication بذلك تعالو نشوف ال Response

```
GET https://syndication.twitter.com/settings HTTP/1.1
Host: syndication.twitter.com
Origin: https://platform.twitter.com
```

فهم سياسة نفس الاصل Same Origin Policy (تكمله...)

- بالفعل سمح موقع Syndication.twitter.com لموقع platform.twitter.com ان يأخذ هذه المعلومات المظلاله باللون الاصفر

```
HTTP/1.1 200 OK
access-control-allow-credentials: true
access-control-allow-origin: https://platform.twitter.com
cache-control: must-revalidate, max-age=600
content-length: 97
content-type: application/json; charset=utf-8
date: Sat, 04 May 2019 09:58:49 GMT
last-modified: Sat, 04 May 2019 09:58:49 GMT
server: tsa_o
set-cookie: tfw_exp=0; Max-Age=86400; Expires=Sun, 5 May 2019 09:58:49 GMT; Path=/; Domain=.twitter.com
strict-transport-security: max-age=631138519
vary: Origin
x-connection-hash: 5448a75fbb4145c52757431d95ea9c71
x-response-time: 116

{"should_obtain_cookie_consent":false,"is_bucketed":false,"experiments":{},"is_allowed_ads":true}
```

ثغرة Misconfigured CORS

- للأسف الشديد مع كل تلك الحمايات والأساليب المعقدة من اجل رفع مستوى أمان أعمال المبرمج في الويب الا أن المبرمج يتسبب بنفسه بثغرة خطيرة قد تسبب الى سحب معلومات مستخدمي موقعه او تطبيقه او عملاءه ويفقد سمعته في الخصوصية وأشياء ممكن ان تكون أخطر أخطر
- **سبب حدوث هذه الثغرة : عدم علم او معرفة كافية للمبرمج عن حماية CORS بحيث يسمح لأي موقع ويقوم بتمرير البيانات له مباشرة**
- **خطورة الثغرة : تسبب الثغرة الى سحب معلومات حساسة بدون أذن الضحية**
- **رقم الثغرة : CWE-942**

```
access-control-allow-credentials: true
access-control-allow-origin: *
```

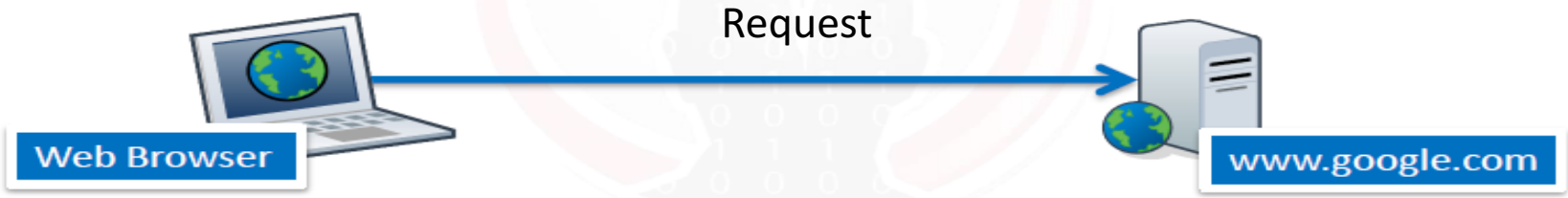
فهم ال Cookies

- تضع معظم مواقع الويب، عندما يتم زياراتها ملفاً صغيراً على القرص الصلب الخاص بجهاز الزائر (المتصفح)، هذا الملف يسمى "كوكي" ((Cookie، وملفات الكوكيز هي عبارة عن ملفات نصية، اذ أنها ليست برامج أو شفرات برمجية
- ويهدف هذا الكوكي إلى جمع بعض المعلومات عنك، وهو مفيد أحياناً، خاصة إذا كان الموقع يتطلب منك إدخال كلمة مرور تخولك بزيارته. ففي هذه الحالة لن تضطر في كل زيارة لإدخال تلك الكلمة، إذ سيتمكن الموقع من اكتشافها بنفسه عن طريق "الكوكي"، الذي تم وضعه على القرص الصلب في الجهاز وذلك من اول زيارة بمعنى أخر تحتوي هذه الملفات النصية (الكوكيز) على معلومات تتيح للموقع الذي أودعها أن يسترجعها عند الحاجة، أي عند زيارتكم المقبلة للموقع
- حيث يتم ارسال الCookie فى رسالة الرد على المتصفح ضمن ال Response Headers

فهم ال Cookies (تكمله ...)

- يتكون ال Cookie من مجموعة من الحقول هى
- 1- Domain هو يحدد اسم الدومين الذى جاء منه ال Cookie للمتصفح
- 2- Path هو المسار المستخدم فى الدومين و يحدد استخدام ال Cookie بالتحديد فى انهى مسار
- 3- Content هو قيمة ال Cookie و هى على شكل name=value
- 4- Expires هو يحدد وقت انتهاء استخدام ال Cookie
- 5- HTTP Only Flag هو يجبر المتصفح على ارسال ال Cookie خلال برتوكول HTTP حيث يمنع من قرائتها بواسطة JS, Flash ,Java او اى تكنولوجى غير HTML
- 6- Secure Flag معناه ان ال Cookie سوف ترسل مشفرة من خلال بروتوكول HTTPS

فهم ال Cookies (تكمله ...) مثال

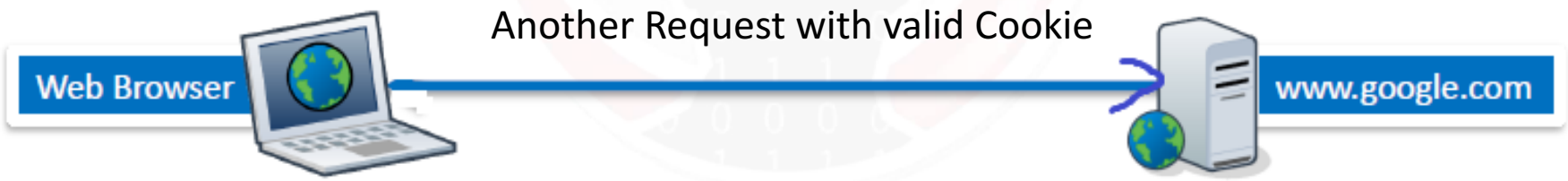



```
POST /login.php
Host: www.google.com

usr=John&Pass=mypass
```


```
HTPP /1.1 200OK
...
Set-Cookie: domain=google.com; path=/; expires=espires=Mon;
16-Apr-2013 19:03:22 GMT; authenticated='1'; httpOnly;
secure;
<PAGE CONTENT>
```

فهم ال Cookies (تكمله ...) مثال




```
GET /mail.php
Host: www.google.com
Cookie=authenticated="1";
```

Examples of Correct Cookie Installation

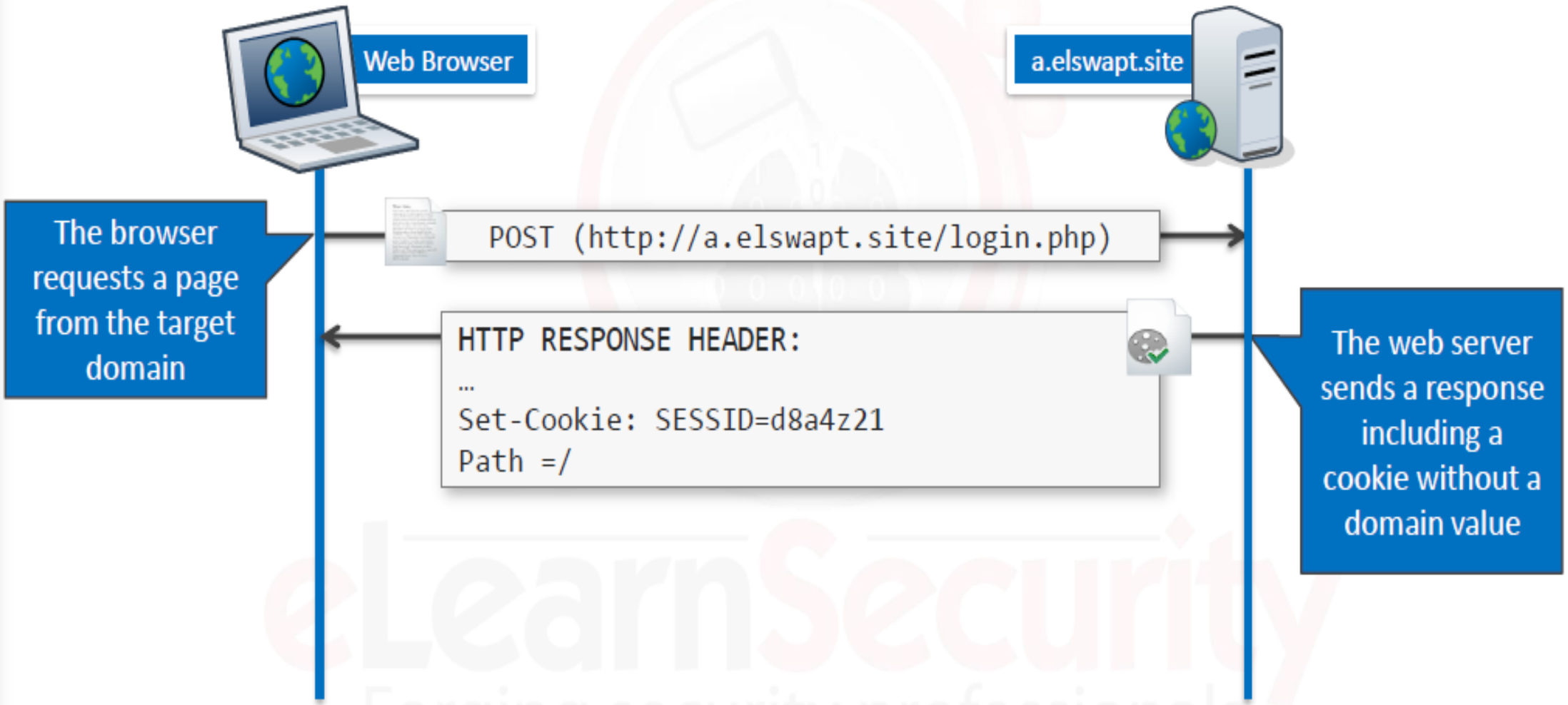
Examples of Incorrect Cookie Installation

Correct Cookie Installation Examples

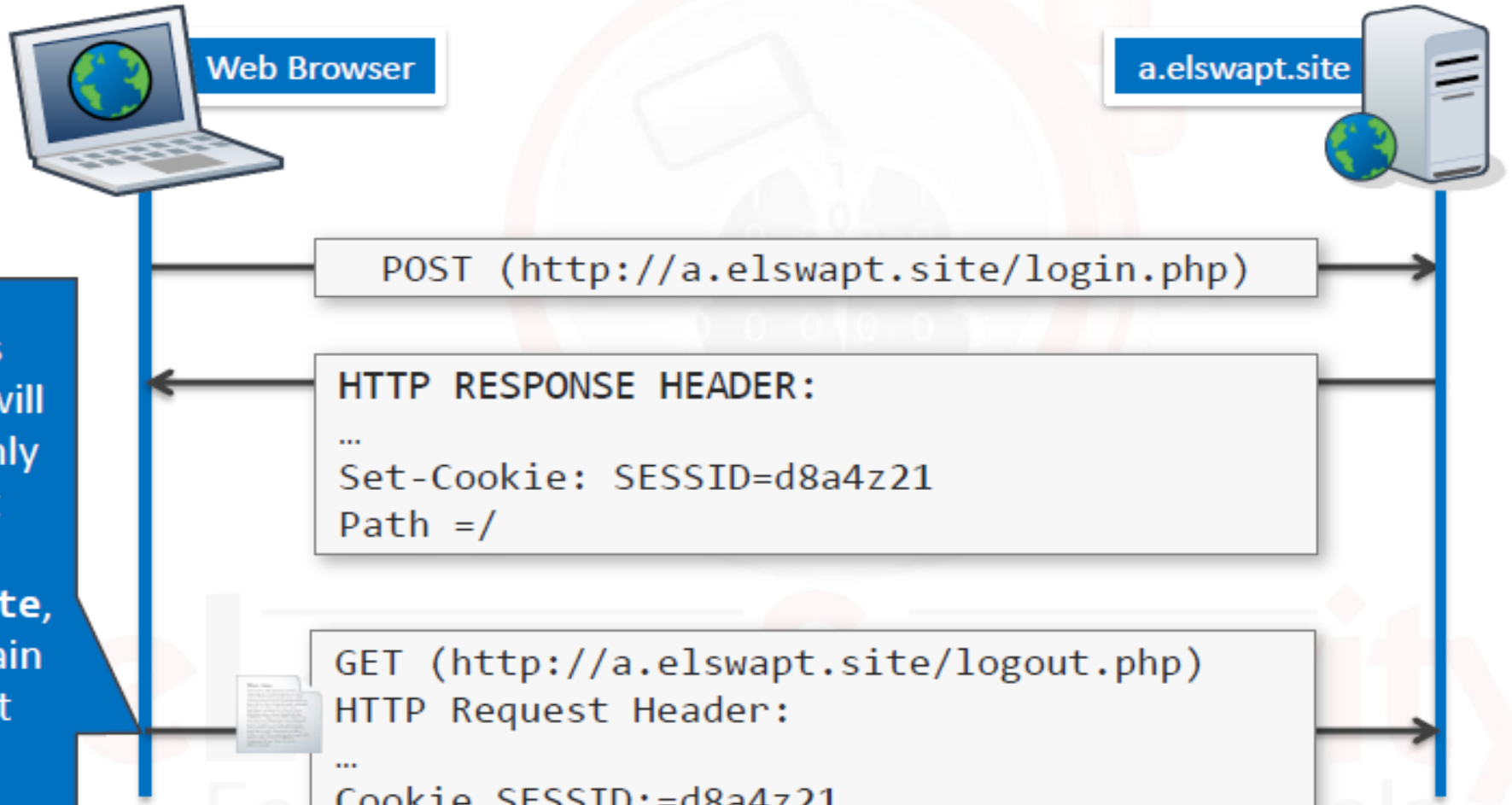
Example #1


Web Browser
a.elswapt.site
The browser requests a page from the target domain
POST (http://a.elswapt.site/login.php)

Example #1



Example #1


Web Browser
a.elswapt.site
POST (http://a.elswapt.site/login.php)
HTTP RESPONSE HEADER:
…
Set-Cookie: SESSID=d8a4z21
Path =/
GET (http://a.elswapt.site/logout.php)
HTTP Request Header:
…
Cookie SESSID:=d8a4z21
The cookie is accepted and will be available only to the target domain a.elswapt.site, since the domain value was not specified.

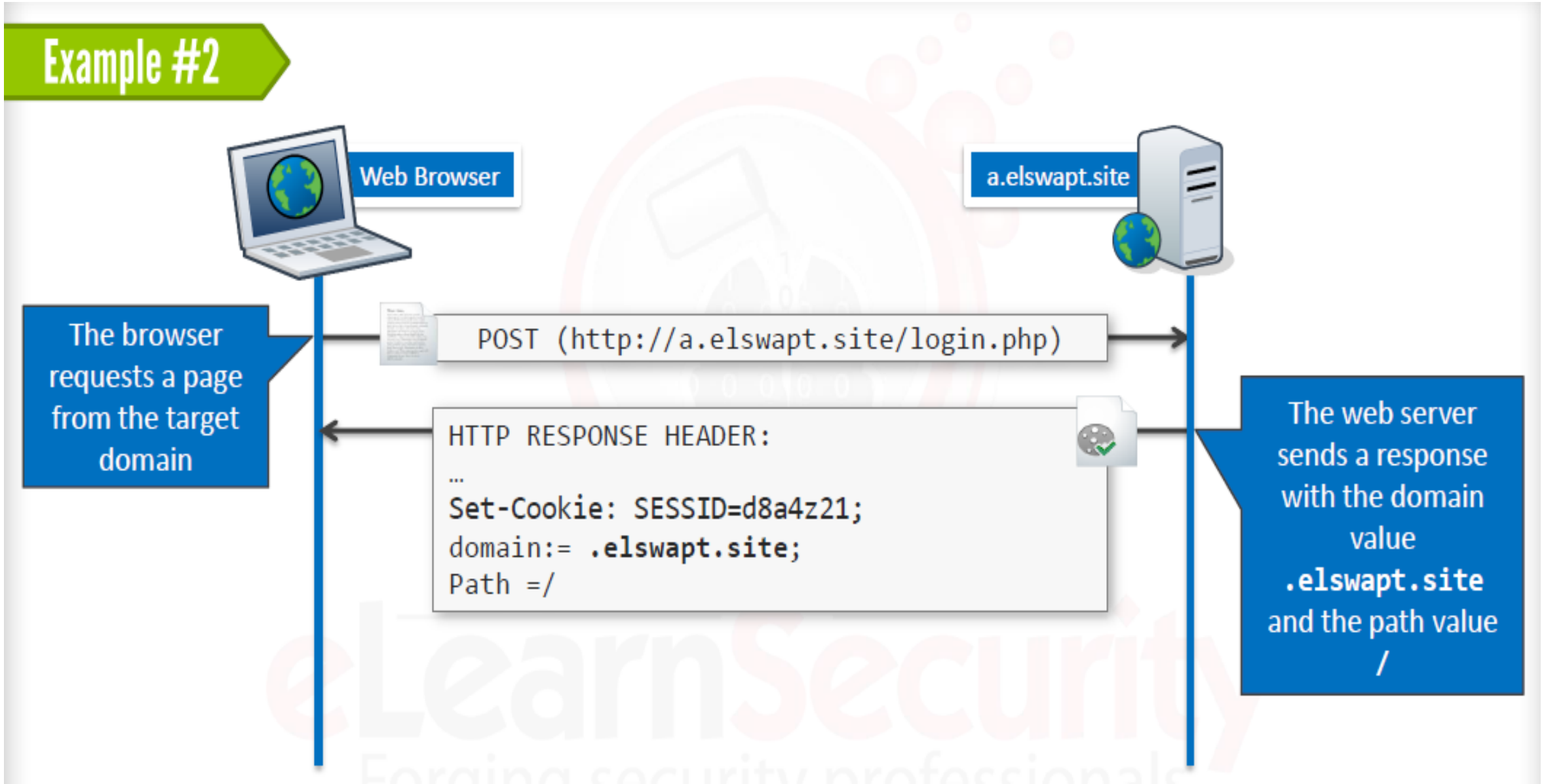
Example #1

This cookie will be sent in each HTTP request matching the following URLs:

- http://a.elswapt.site/*
- https://a.elswapt.site/*

Example #2

Example #2


Web Browser
a.elswapt.site
The browser requests a page from the target domain
POST (http://a.elswapt.site/login.php)
HTTP RESPONSE HEADER:
…
Set-Cookie: SESSID=d8a4z21;
domain:= .elswapt.site;
Path =/
The web server sends a response with the domain value .elswapt.site and the path value /

Example #2

The cookie is accepted because the domain value `.elswapt.site` is a suffix of the domain emitting the cookie, `a.elswapt.site`, therefore it will be accepted and sent in each request matching the following URLs:

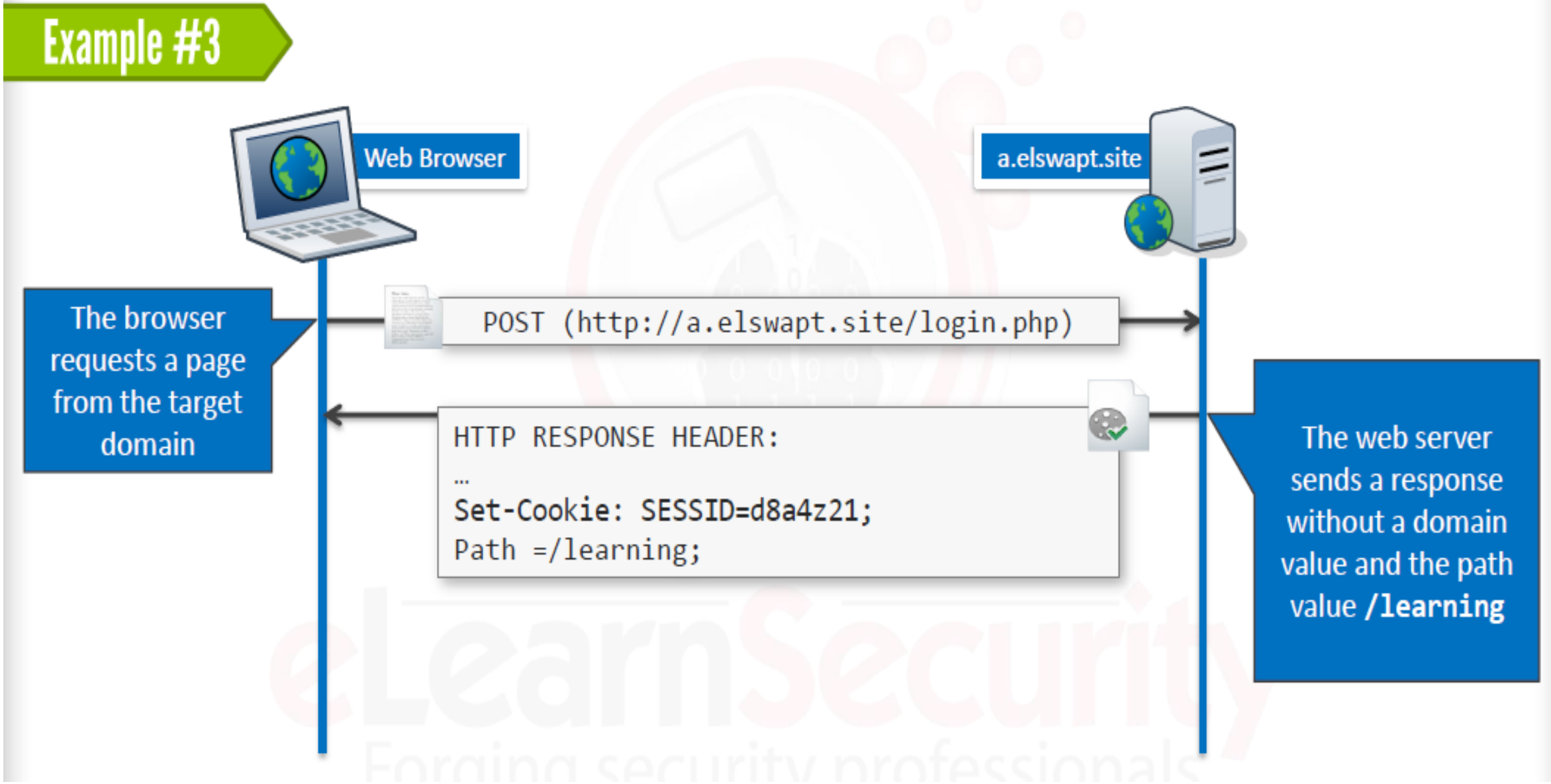
- http://elswapt.site/*
- https://elswapt.site/*
- http://*.elswapt.site/*
- https://*.elswapt.site/*

Example #2

This is what will happen. The cookie previously set is sent to both **a** and **b** subdomains.

Example #3

Example #3



Example #3

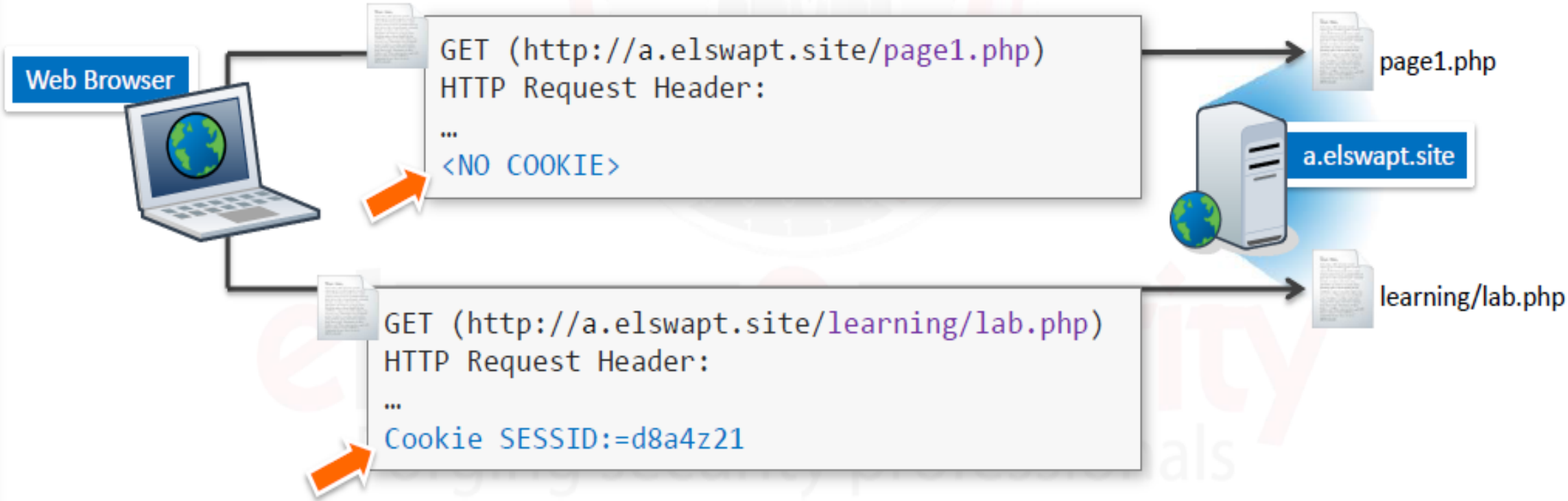
The cookie is accepted and will be available only to the target domain `a.elswapt.site` and path `/learning/*`.

So, this cookie will be sent in each request matching the following URLs:

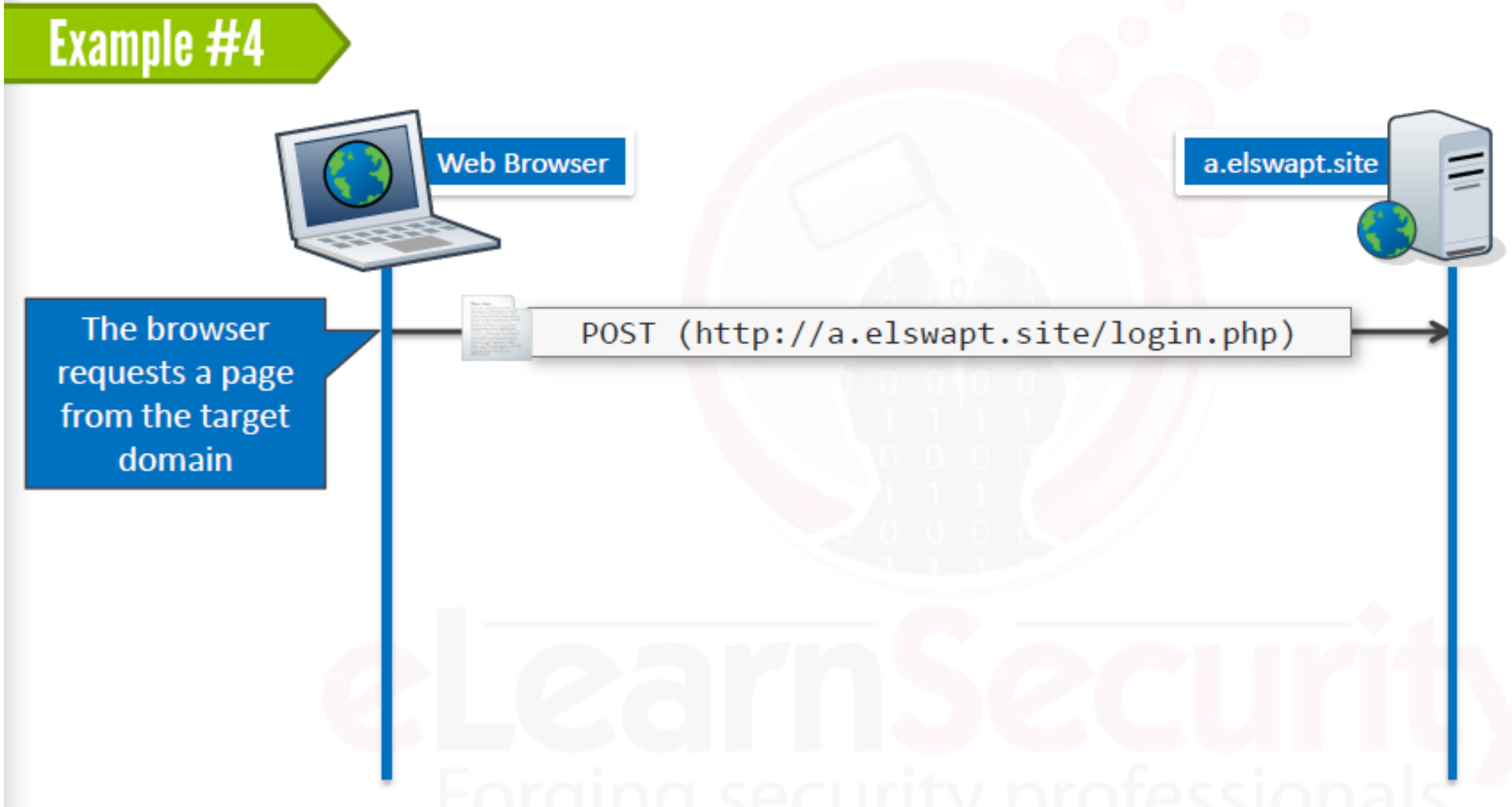
- http://a.elswapt.site/learning/*
- https://a.elswapt.site/learning/*

Example #3

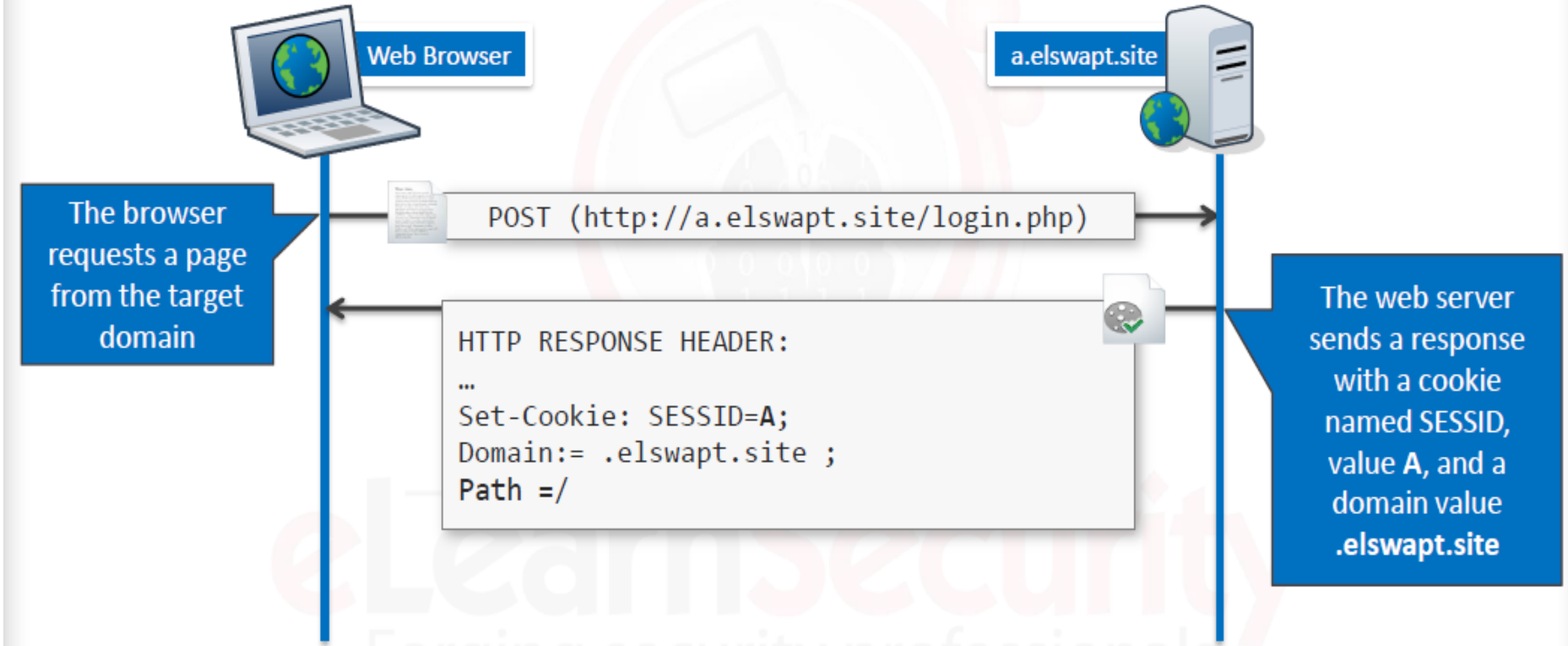
This is what will happen. The cookie will be sent for resources in the /learning/ path.



Example #4



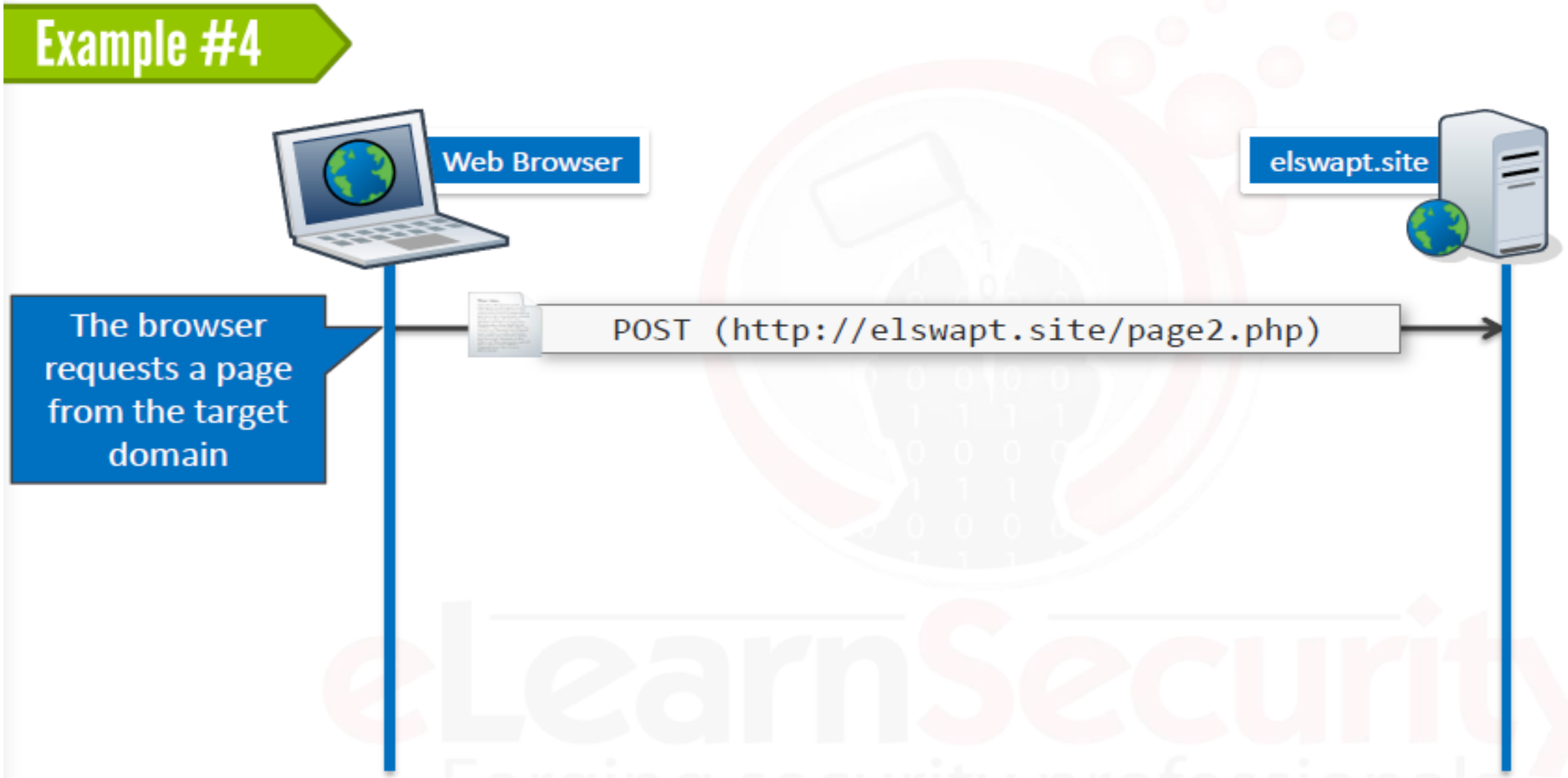
Example #4



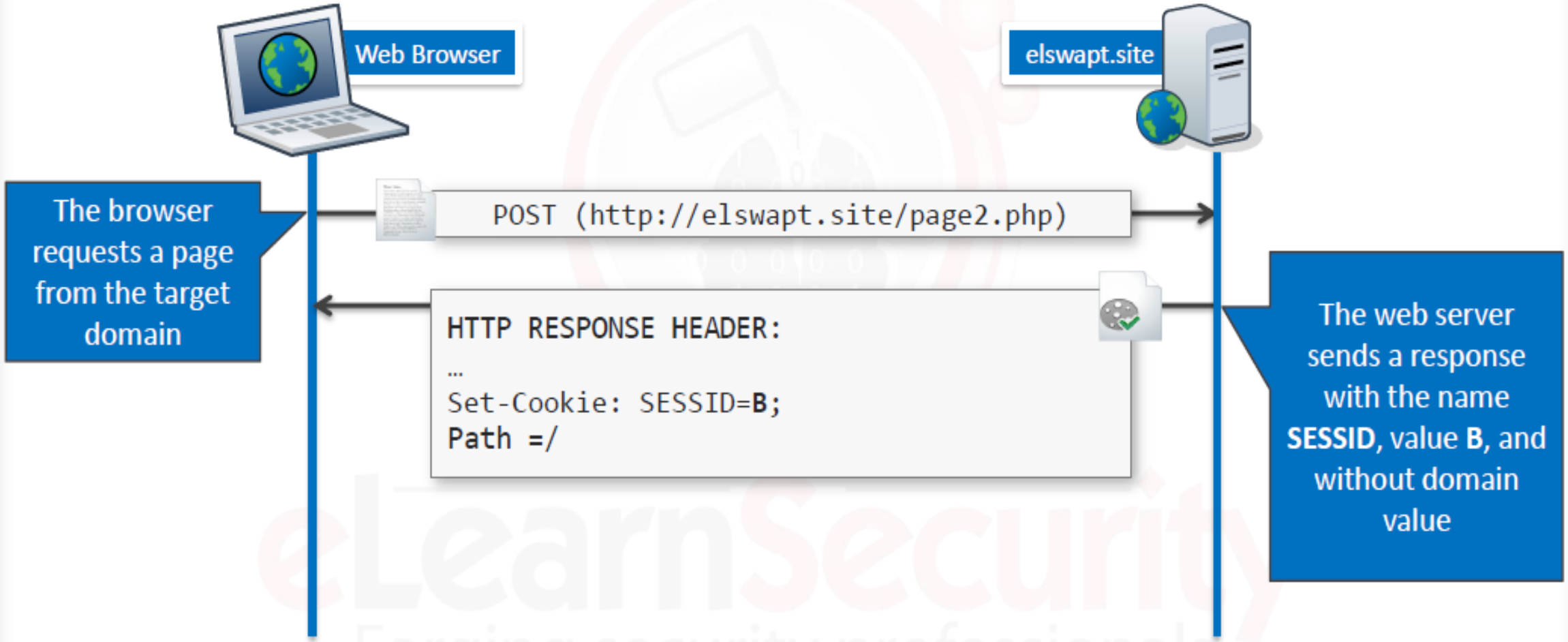
Example #4

After that, the browser requests a second page from the target domain `elswapt.site` and the web server sends a response including a cookie with the name `SESSID`, value `B`, and without domain value.

Example #4



Example #4

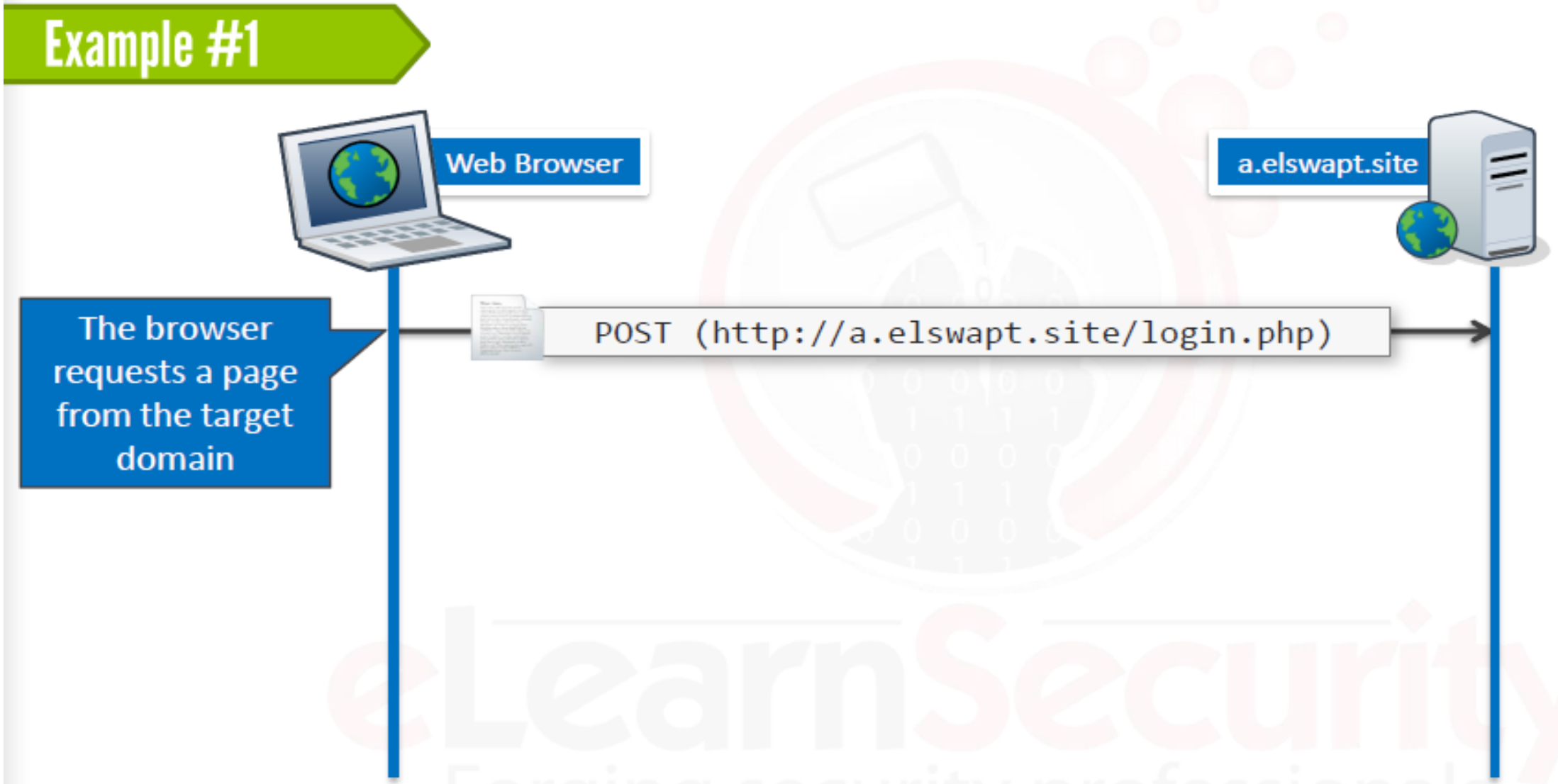


Example #4

Both cookies will be accepted and stored by the browser. They will not interfere with one another as they are two different cookies.

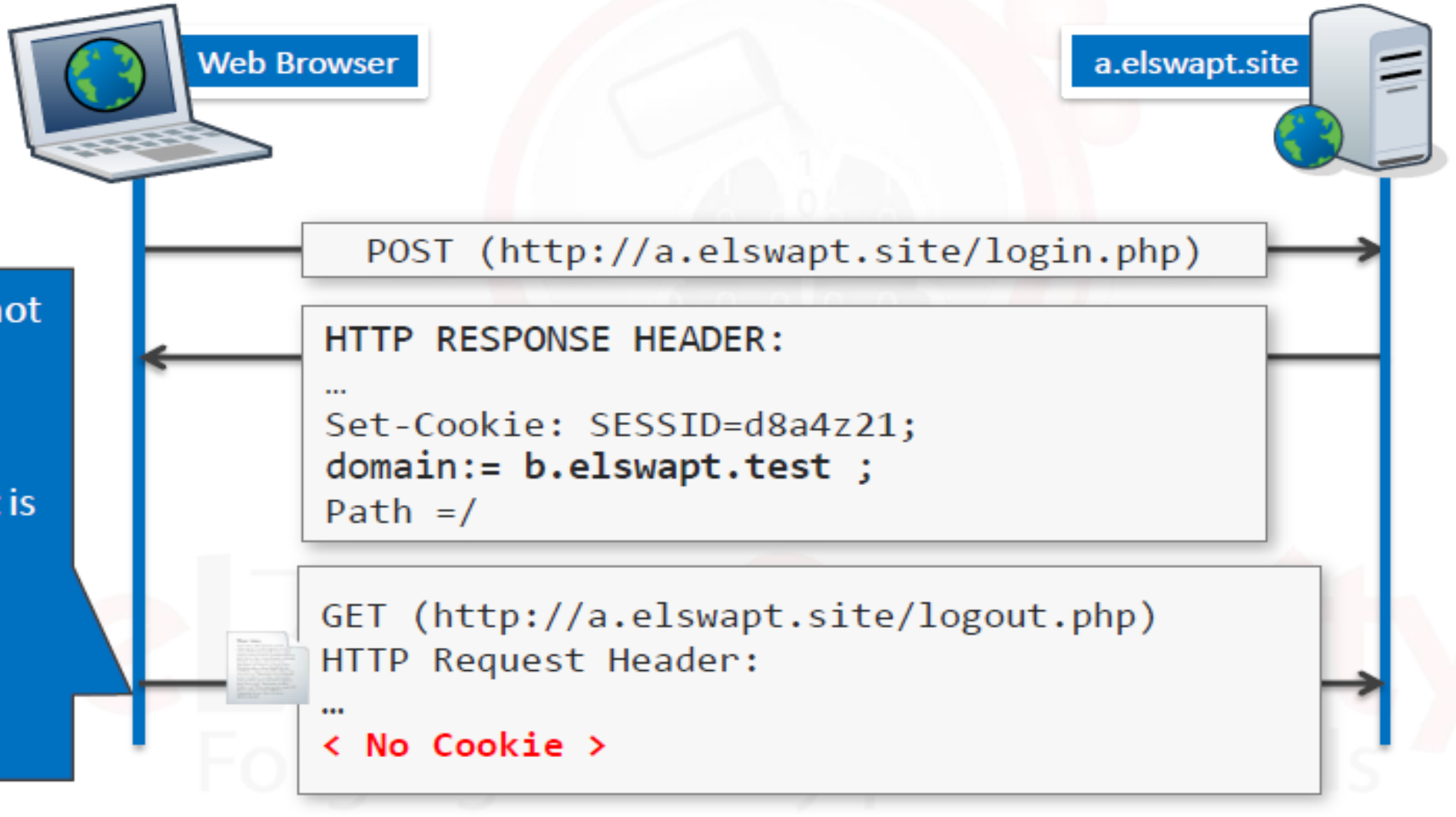
Incorrect Cookie Installation Examples

Example #1



Example #1

Example #1


Web Browser
a.elswapt.site
POST (http://a.elswapt.site/login.php)
HTTP RESPONSE HEADER:
…
Set-Cookie: SESSID=d8a4z21;
domain:= b.elswapt.test ;
Path =/
GET (http://a.elswapt.site/logout.php)
HTTP Request Header:
…
< No Cookie >
The cookie is not accepted because the domain b.elswapt.test is not a suffix of the domain a.elswapt.site that sent the cookie.

Example #2

Example #2

Web Browser

a.elswapt.site

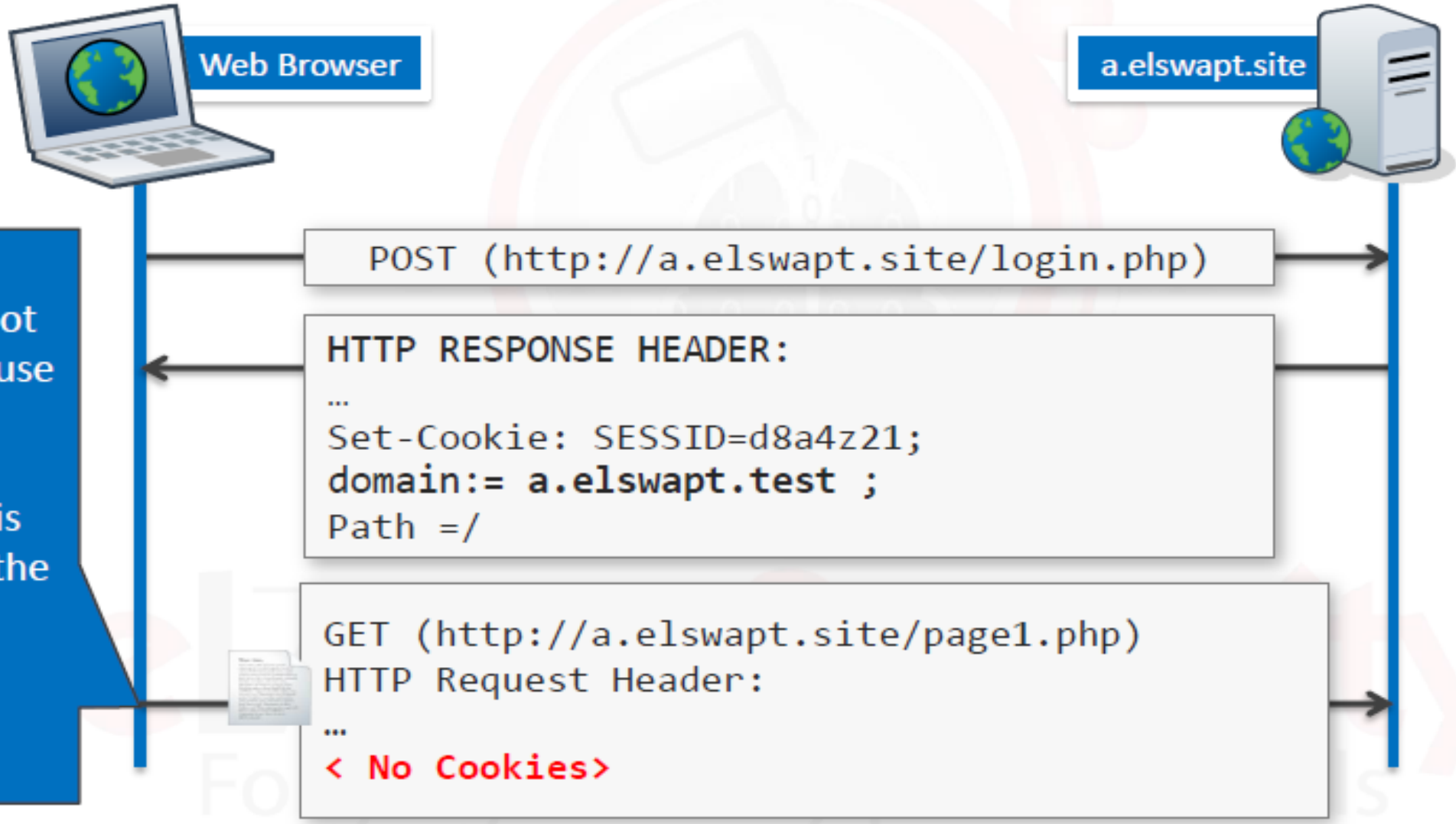
The browser requests a page from the target domain

POST (http://a.elswapt.site/login.php)

HTTP RESPONSE HEADER:
…
Set-Cookie: SESSID=d8a4z21;
domain:= b.elswapt.site ;
Path =/

The web server sends a response including a cookie with the domain b.elswapt.site

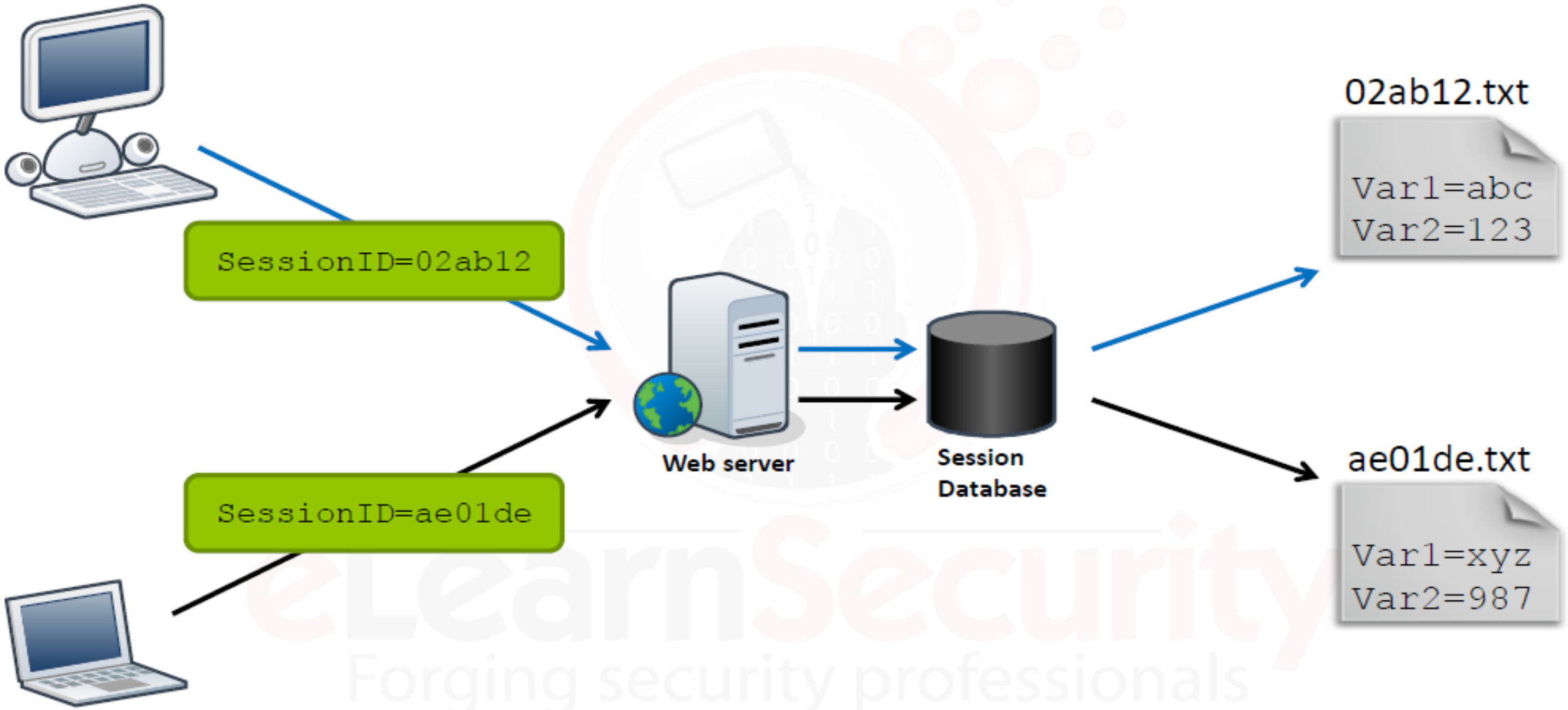
Example #2


Web Browser
a.elswapt.site
POST (http://a.elswapt.site/login.php)
HTTP RESPONSE HEADER:
…
Set-Cookie: SESSID=d8a4z21;
domain:= a.elswapt.test ;
Path =/
GET (http://a.elswapt.site/page1.php)
HTTP Request Header:
…
< No Cookies>
The cookie is not accepted because the cookie domain value b.elswapt.site is not a suffix of the domain a.elswapt.site emitting it.

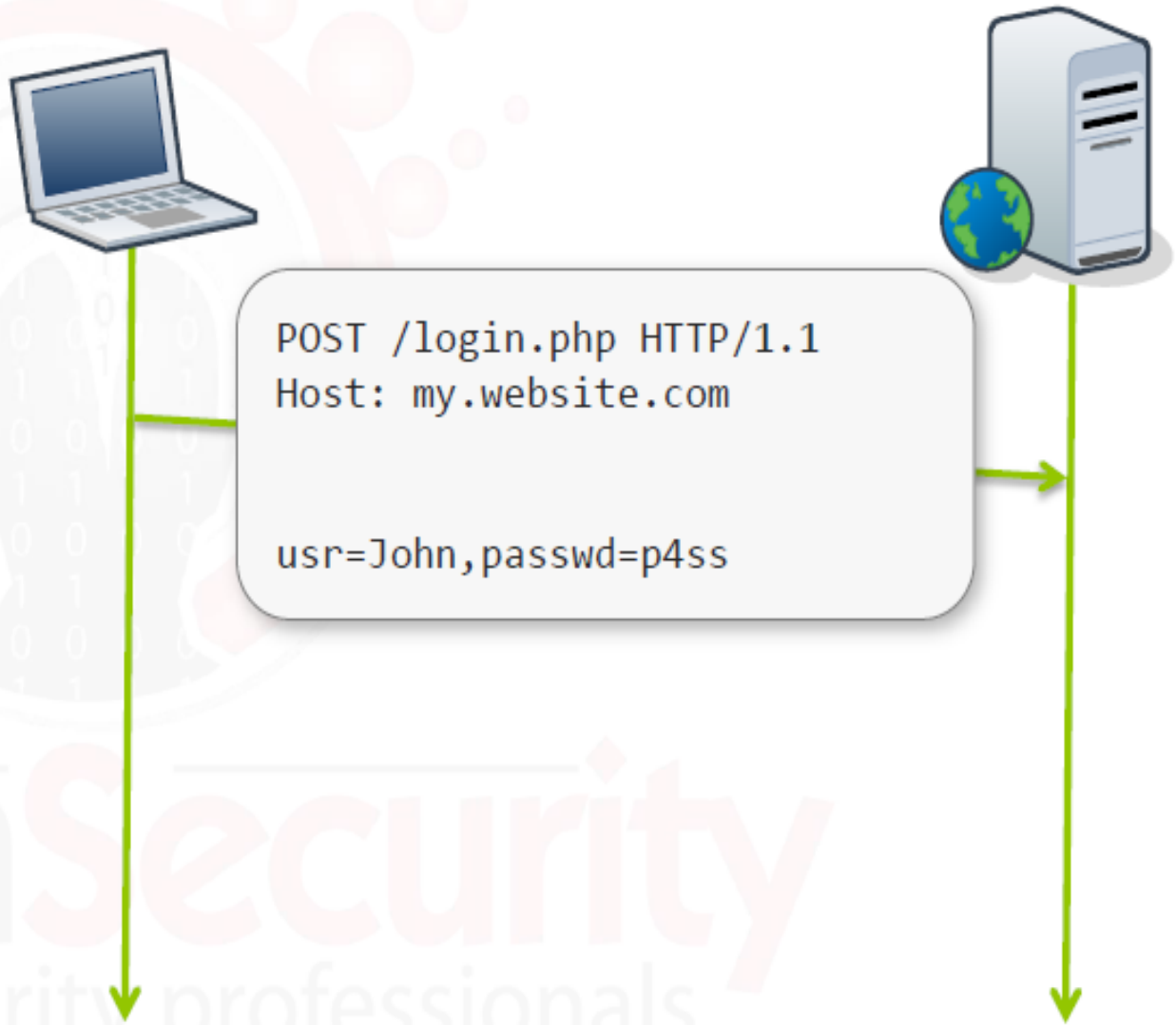
فهم ال Session

- عند الانتقال من صفحة إلى أخرى في موقع معين فإن بروتوكول الـ HTTP لا يمكنه معرفة أن تلك الصفحات قد تم تصفحها من قبل نفس الشخص أم لا
- حيث أن الـ HTTP لا يوفر لنا آلية لعمل ذلك التواصل (بين المستخدم و الخادم) ، فإذا ما طلب المستخدم صفحة من الخادم فإن الخادم يقوم بإعطائه ما أراد و ينتهي عند ذلك فلا يعرف إن كان هو نفس المستخدم أو ليس هو
- لأجل ذلك تم إنشاء تقنية ال Cookies كما ذكرنا سابقا و ال Session للحفاظ على الترابط بين المستخدم و الخادم
- حيث يتم تخزين ال Session عند الخادم على عكس ال Cookie يتم تخزينها عند المتصفح المستخدم
- حيث ان لكل مستخدم Session Id/Token
- وقت انتهاء ال Session اسرع من انتهاء ال Cookie

فهم ال Session (تكمله ...)

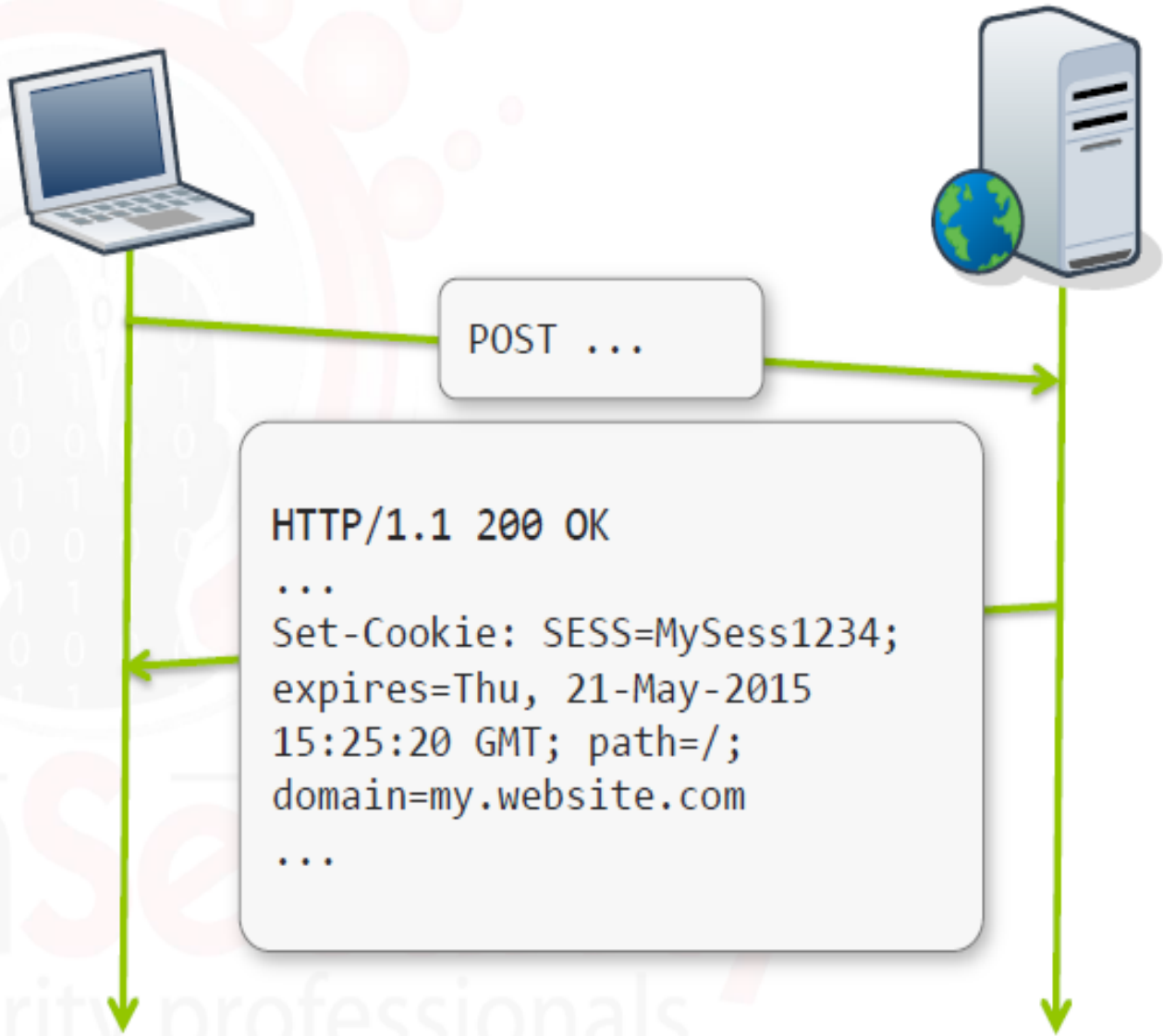


The client uses a login form to POST the user's credentials.

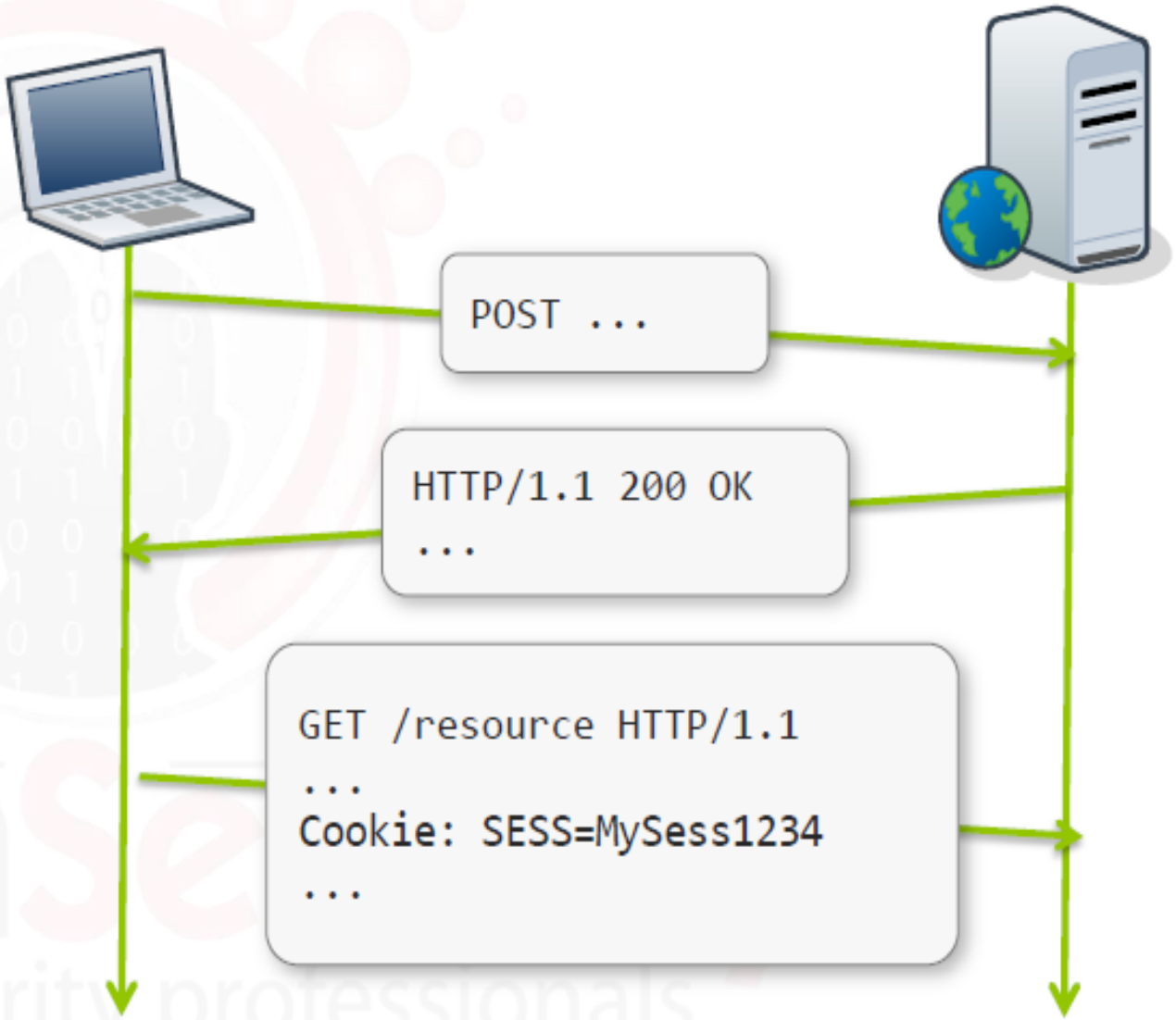


The server sends back a response with a Set-cookie header field.

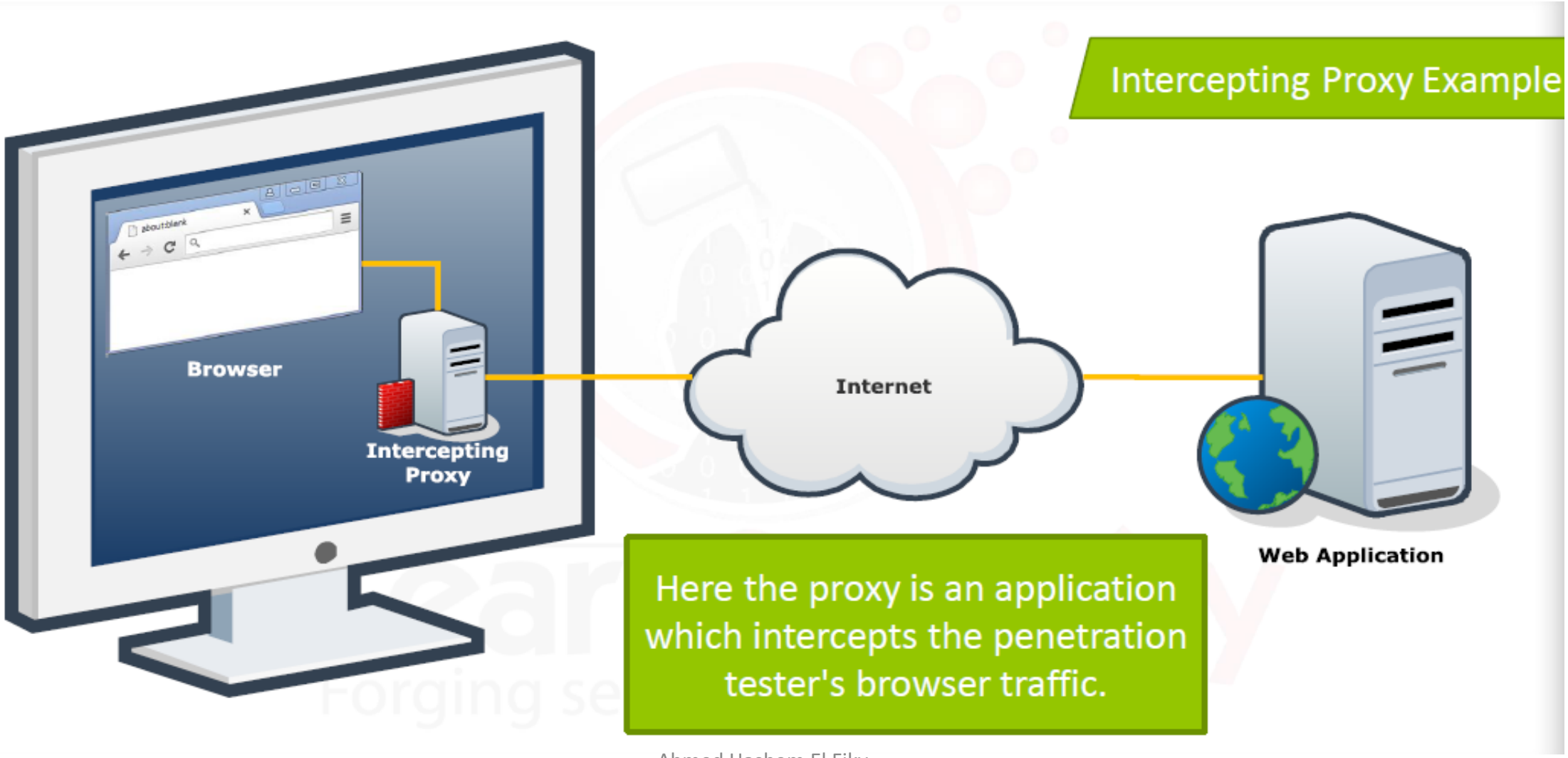
The cookie contains the **session ID**.



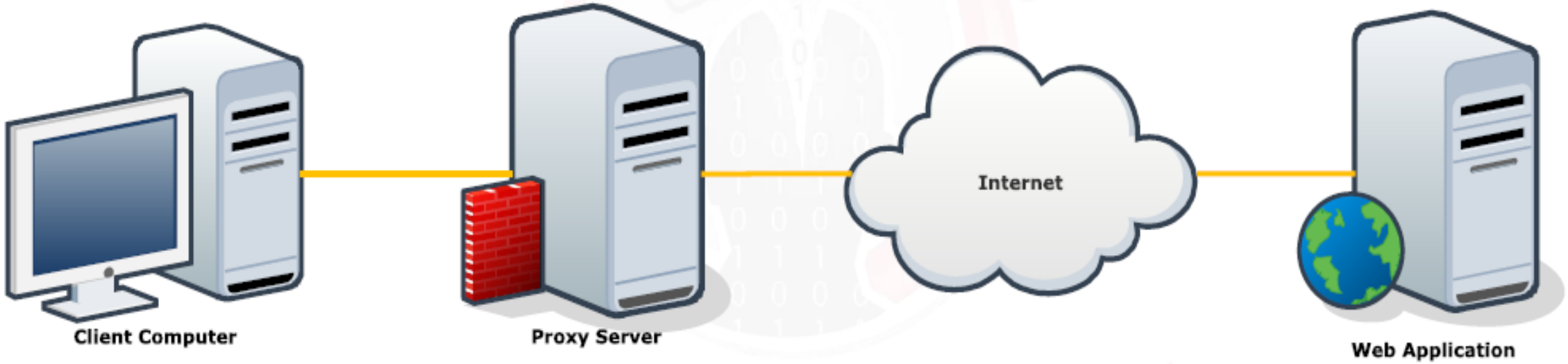
The browser will send back the cookie according to the cookie protocol, thus sending the **session ID**.



أمثلة على Web Application Proxies

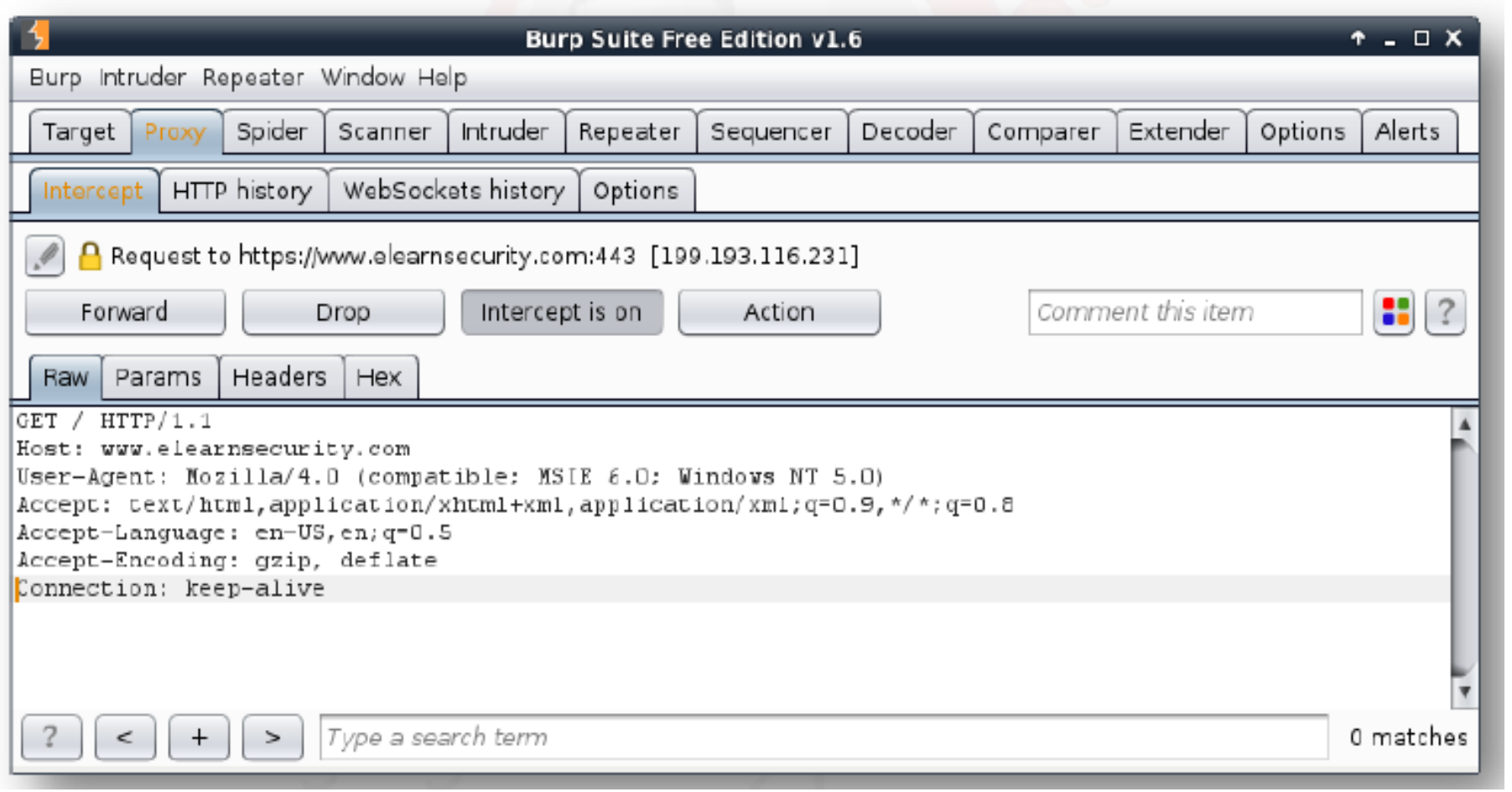


Proxy Server Example



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

الادوات المستخدمة ك Intercept Proxy



- اداة ال BurpSuite
- اداة ال ZAP

تم بحمد الله انتهاء المقدمة