

MODERN WEB APPLICATION DEFENSES AGAINST DANGEROUS NETWORK ATTACKS

Philippe De Ryck

SecAppDev 2017

 <https://www.websec.be>

 @PhilippeDeRyck

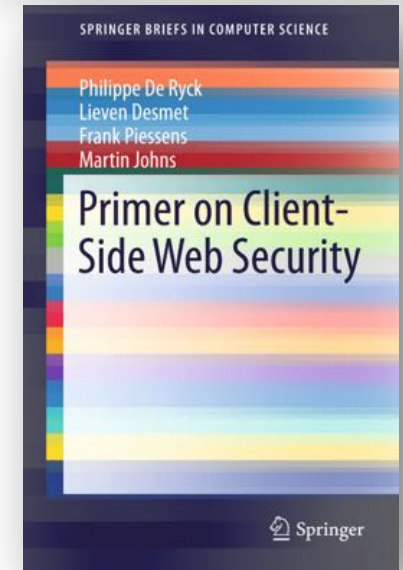
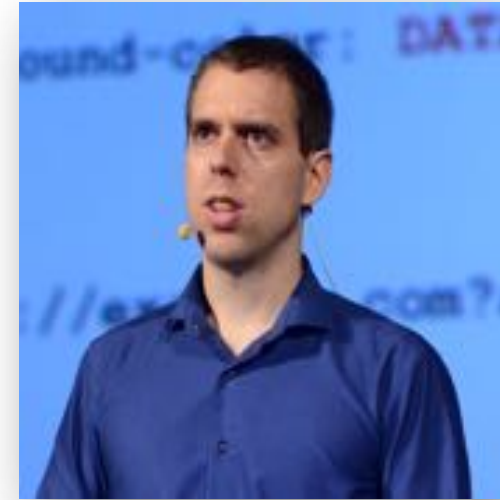
SETUP OF THE HANDS-ON SESSION

- I have prepared a minimal amount of slides
 - Explain key concepts and new security policies
 - Just enough to make sure you know what you're doing during the hands-on part
- Hands-on part runs entirely within a VirtualBox VM
 - All software is pre-installed
 - We run our exercises on the SuperBug training application
- If you have any questions / remarks / discussions ... just call me over
 - Feel free to take the lab as far as you want to

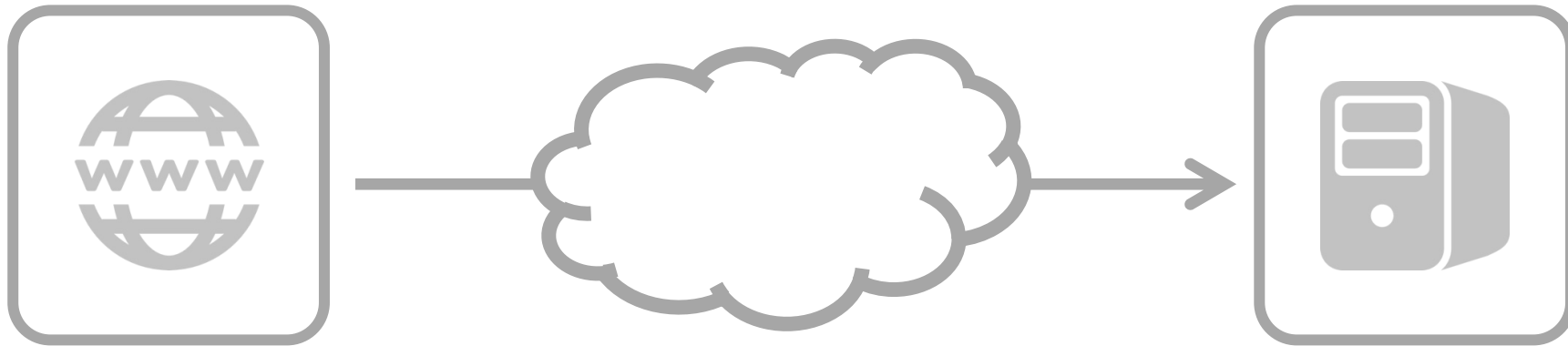
ABOUT ME – PHILIPPE DE RYCK

- My goal is to help you build secure web applications
 - Hosted and customized in-house training
 - Specialized security assessments of critical systems
 - Threat landscape analysis and prioritization of security efforts
 - More information and resources on <https://www.websec.be>

- My security expertise is broad, with a focus on Web Security
 - PhD in client-side web security
 - Main author of the *Primer on client-side web security*



WHY SECURE THE COMMUNICATION CHANNEL?



**SSL was introduced in 1995 by Netscape
to secure online transactions**

THANKS TO SNOWDEN, HTTPS IS FINALLY TAKEN SERIOUSLY



Google uses HTTPS as a ranking signal

Firefox warns about unsafe form submissions

Since November 2016, more than 50% of pages are visited over HTTPS

AWESOME SERVICES HELP IMPROVE HTTPS DEPLOYMENTS

Let's Encrypt is a **free, automated, and open** Certificate Authority.

OBSERVATORY by mozilla

Scan Summary

A

| | |
|----------------|----------------------------|
| Host: | websec.be |
| Scan ID #: | 2757926 |
| Test Time: | November 29, 2016 10:47 AM |
| Test Duration: | 5 seconds |
| Score: | 95/100 |
| Tests Passed: | 10/11 |

SSL Report: www.websec.be (52.58.139.189)

Assessed on: Sun, 04 Dec 2016 15:13:28 UTC | [Hide](#) | [Clear cache](#)

[Scan Another »](#)

Summary

Overall Rating

A+



Visit our [documentation page](#) for more information, configuration guides, and books. Known issues are documented [here](#).

HTTP Strict Transport Security (HSTS) with long duration deployed on this server. [MORE INFO »](#)

<https://letsencrypt.org/>

<https://www.ssllabs.com/sslltest/>

<https://observatory.mozilla.org/>

COMMON THREATS TO THE COMMUNICATION CHANNEL



Eavesdropping on the network



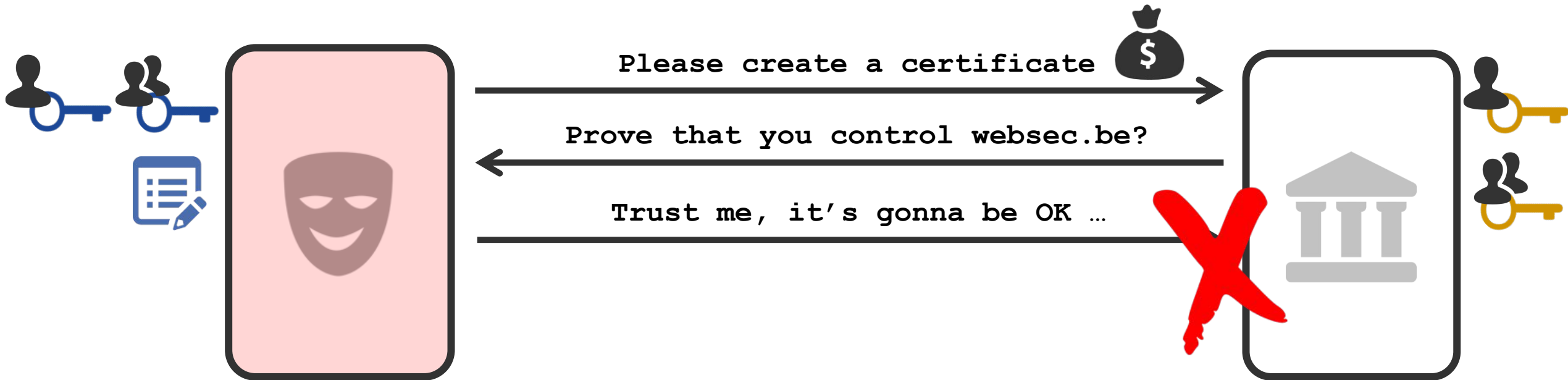
COMMON THREATS TO THE COMMUNICATION CHANNEL

Impersonating a legitimate server



GETTING A CERTIFICATE FROM A CERTIFICATE AUTHORITY

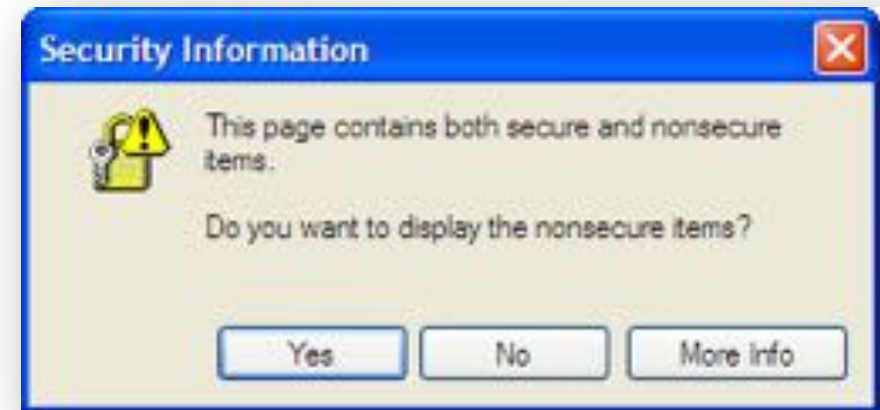
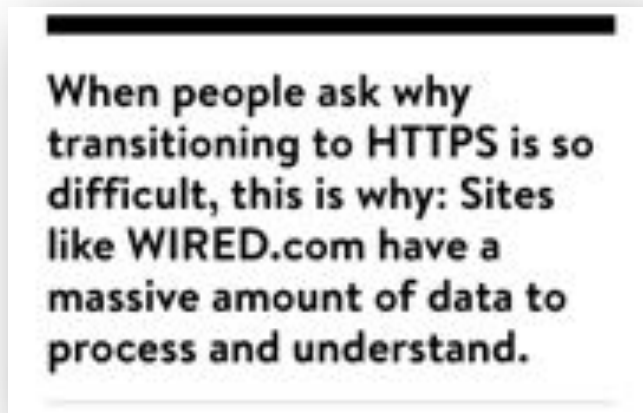
Impersonating a legitimate server with a valid certificate



MOVING FROM HTTP TO HTTPS

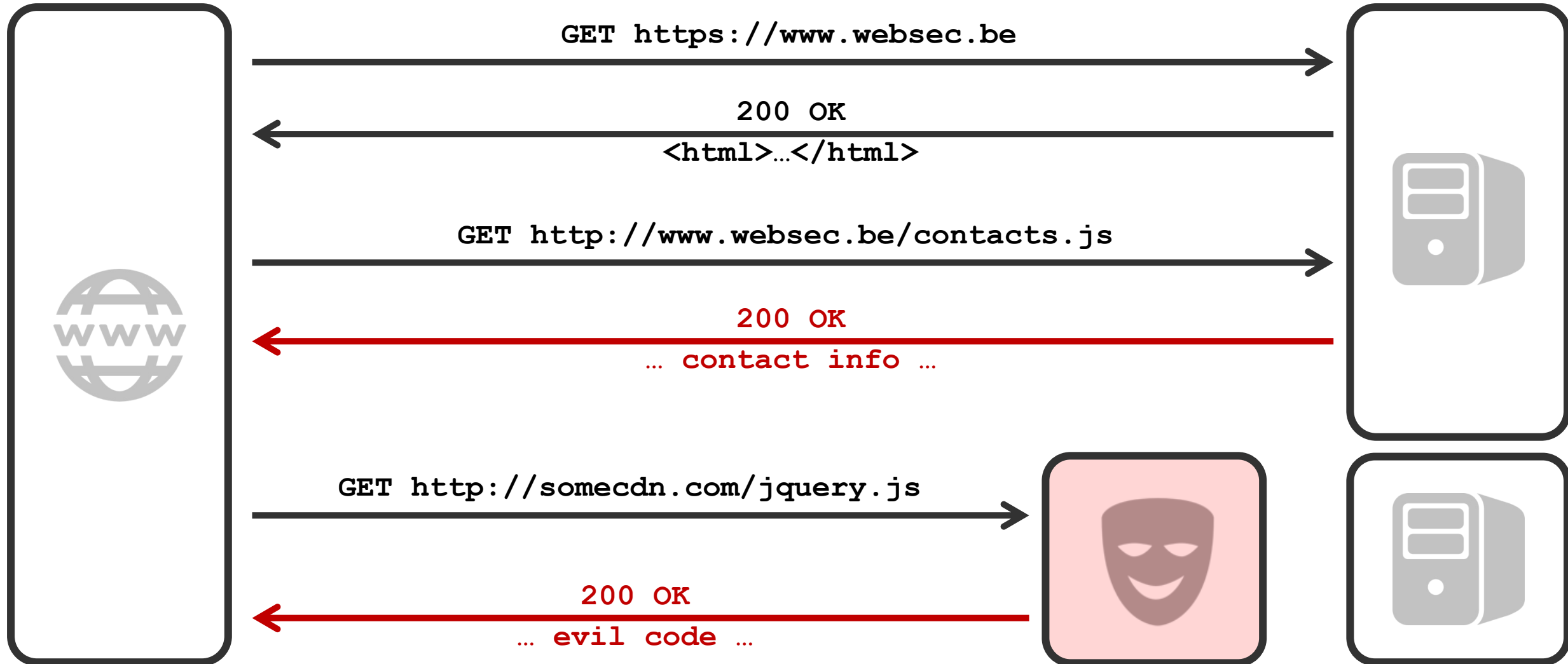
HOW DO YOU GET MIXED CONTENT SITUATIONS?

- Embedding HTTP content in an HTTPS page is considered to be insecure
 - The HTTPS page is loaded over a secure channel, but the HTTP content is not
- This is common when you move from HTTP to HTTPS
 - Your own resources might still be loaded over HTTP
 - External content is often loaded over HTTP as well



https://help.salesforce.com/HTViewSolution?id=000005615&language=en_US
<https://www.wired.com/2016/05/wired-first-big-https-rollout-snap/>

WHY MIXED CONTENT IS CONSIDERED TO BE A PROBLEM



BROWSERS FINALLY FOLLOWING IE'S LEAD

- Internet Explorer warned about mixed content since version 7
 - Today, most desktop browsers block active and warn about passive mixed content
 - Mobile browsers were a bit slower, but have caught up by now

```
▲ Mixed Content: The page at 'https://distrinet.cs.kuleuven.be/' was loaded over (index):1
HTTPS, but requested an insecure image 'http://example.org/insecure.png'. This content
should also be served over HTTPS.
```

```
✖ ▶ Mixed Content: The page at 'https://distrinet.cs.kuleuven.be/' was loaded over VM628:1
HTTPS, but requested an insecure script 'http://example.org/insecure.js'. This request has
been blocked; the content must be served over HTTPS.
```

- Two explicit types of mixed content
 - Passive mixed content does not interact with the page (images, audio, video)
 - Active mixed content has the capability of interacting with the page

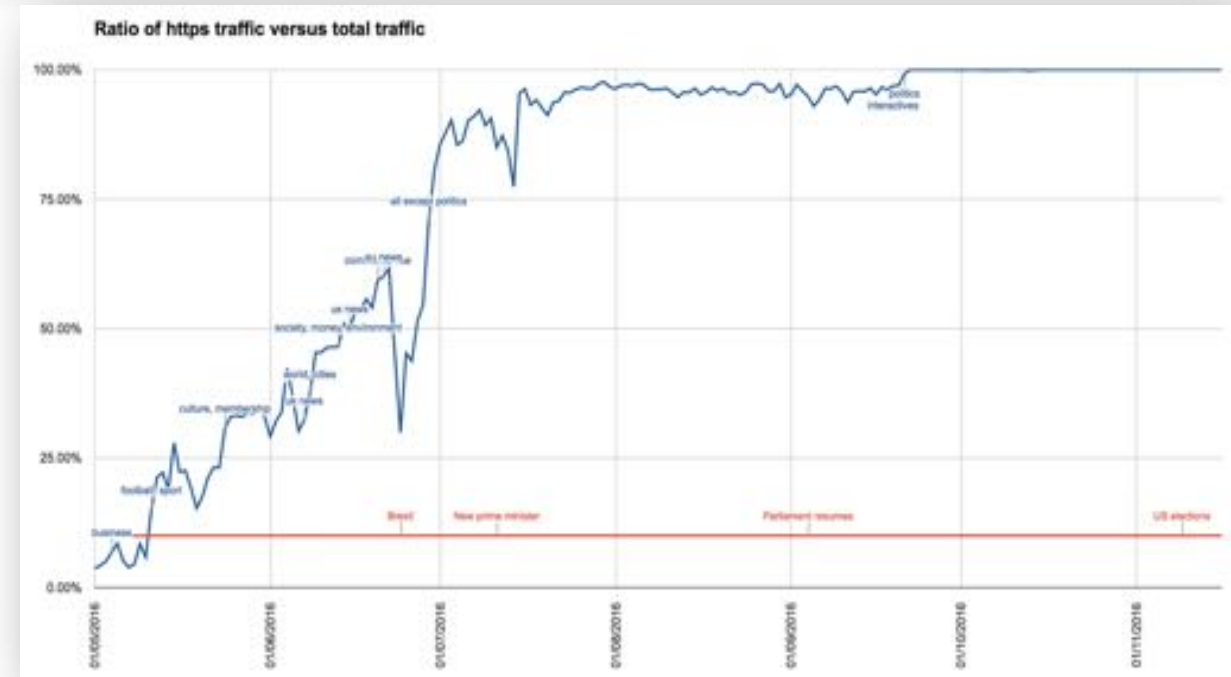
SO HOW DID OTHERS DEAL WITH MIXED CONTENT?

HOW WIRED COMPLETELY ENCRYPTED ITSELF

The Guardian has moved to HTTPS

Discover why and how the Guardian has moved to HTTPS, the secure version of the web protocol that helps to protect user privacy

- Use CSP to detect HTTP content
 - Detect HTTP resources up front
 - Update them before switching
- Gradually move to HTTPS
 - Carefully monitor the process
 - Adapt where necessary



<https://www.wired.com/2016/09/wired-completely-encrypted/>

<https://www.theguardian.com/info/developer-blog/2016/nov/29/the-guardian-has-moved-to-https>

DETECTING MIXED CONTENT WITH CONTENT SECURITY POLICY

- Content Security Policy (CSP) allows you to restrict content on a page
 - Whitelists define where content can be loaded from, everything else is blocked
 - You can instruct the browser to send a report about a violation
 - In report-only mode, the browser will send a report but will not block content

```
Content-Security-Policy-Report-Only:
```

```
default-src https: 'unsafe-inline' 'unsafe-eval';
```

```
report-uri https://yoursite.report-uri.io/default/csp/reportOnly
```

- A report-only CSP policy is ideal to find mixed content on your pages
 - Reports will contain information about the violation
 - You can track these, and fix them before even moving to HTTPS

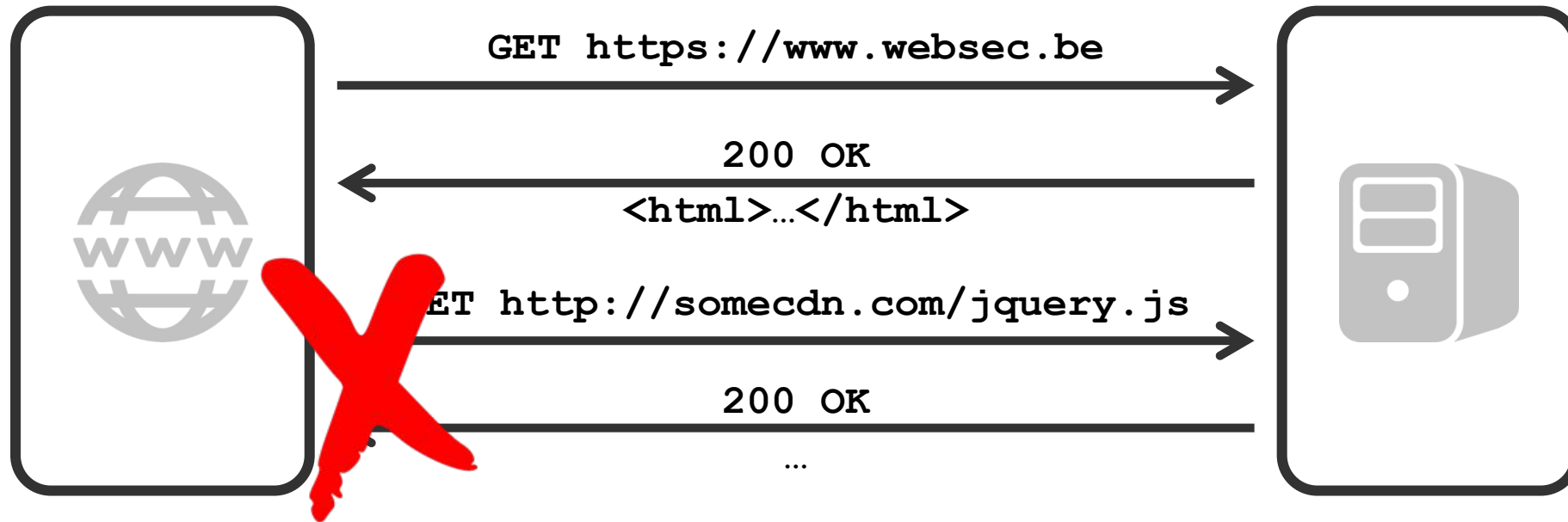
DETECTING MIXED CONTENT WITH CONTENT SECURITY POLICY

| | |
|---------------------|---|
| Document URI | "http://localhost:8080/ViewCSPReports" |
| Blocked URI | "http://ajax.googleapis.com" |
| Referrer | "http://localhost:8080/Home" |
| Violated Directive | "default-src https: 'unsafe-inline' 'unsafe-eval'" |
| Effective Directive | |
| Original Policy | "default-src https: 'unsafe-inline' 'unsafe-eval'; report-uri https://localhost:8443/cspreport" |

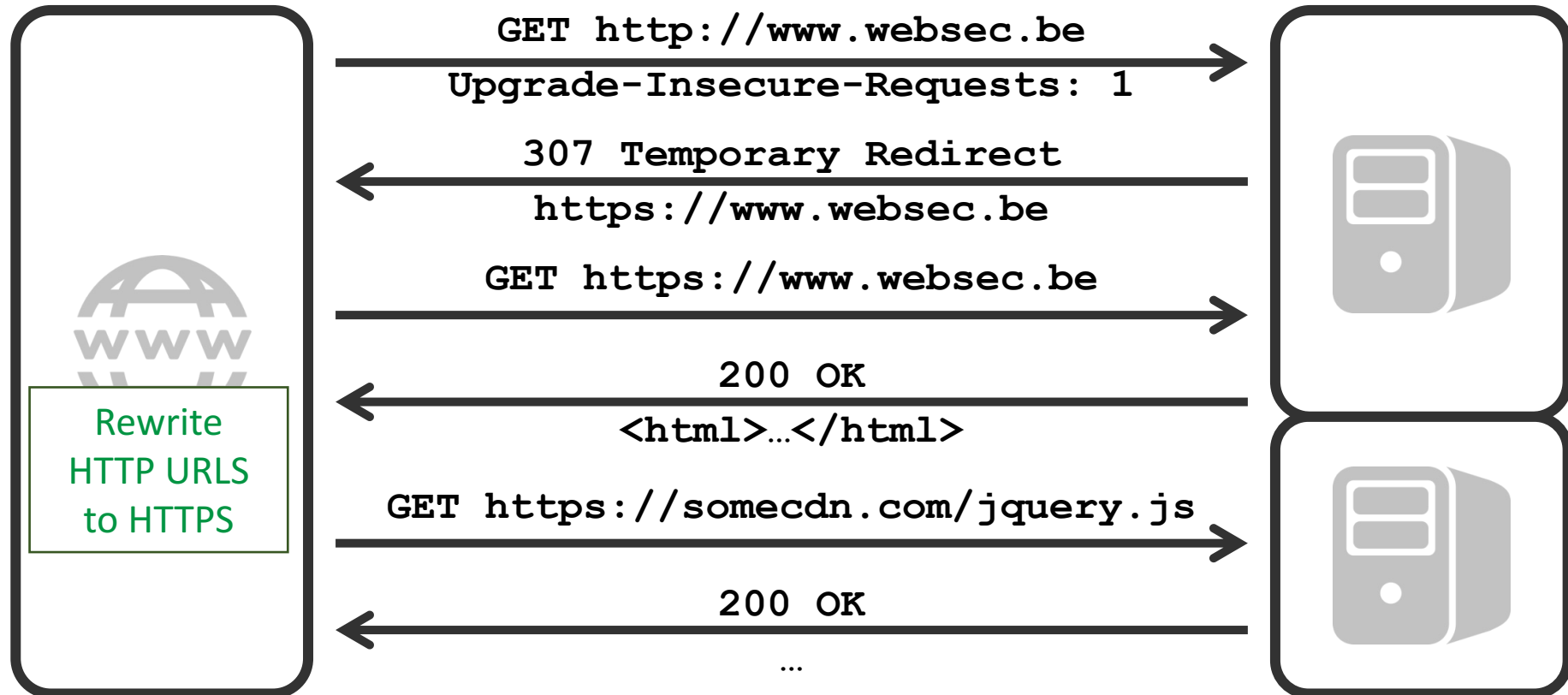
AUTOMATICALLY UPGRADING INSECURE REQUESTS

- Modern browsers can automatically translate HTTP to HTTPS
 - Can be controlled by the **upgrade-insecure-requests** CSP directive
 - Once enabled, all HTTP resources on the page will be loaded over HTTPS
 - Links within the same domain will be treated as HTTPS too
- This mechanism is intended to make the move to HTTPS easier
 - It's not always feasible to update legacy content to avoid mixed content problems
 - It's not a silver bullet, just a tool that makes it possible to secure legacy content
 - Only useful if you are sure that all included HTTP resources are available over HTTPS
- Compatible browsers announce support for the upgrade mechanism
 - When supported, you can redirect to HTTPS and upgrade insecure requests

AUTOMATICALLY UPGRADING INSECURE REQUESTS



AUTOMATICALLY UPGRADING INSECURE REQUESTS



BROWSER SUPPORT FOR UPGRADE-INSECURE-REQUESTS



BLOCKING MIXED CONTENT ONCE AND FOR ALL

- CSP helps you prevent mixed content once your transition is complete
 - The **block-all-mixed-content** directive prevents the loading of HTTP content
 - Applies to everything, including passive mixed content

```
Content-Security-Policy: block-all-mixed-content
```

- This is called *strict mode* for mixed content
 - Prevents the user from potentially overriding this setting
 - Avoids any UI indicator that mixed content is being blocked
 - Also applies to nested browsing contexts

BEST PRACTICES

- Do not underestimate the move to HTTPS
 - Start by reading up on potential problems from experience reports
 - The move to HTTPS not only impacts your content, but also your SEO
 - Use CSP to detect mixed content problems before making the transition
- Gradually tackle the transition to HTTPS and learn as you go
 - Start loading external content over HTTPS before making the move yourself
 - Update all references to `http://` with `https://` or the relative `//`
 - If your user base does not use IE, use the `upgrade-insecure-requests` directive
- Blocking mixed content can be useful to avoid future mixed content problems
 - Only use `block-all-mixed-content` when all resources are loaded over HTTPS



Hands-on Lab Part 1

PROS / CONS OF THE LATEST HTTPS SECURITY POLICIES

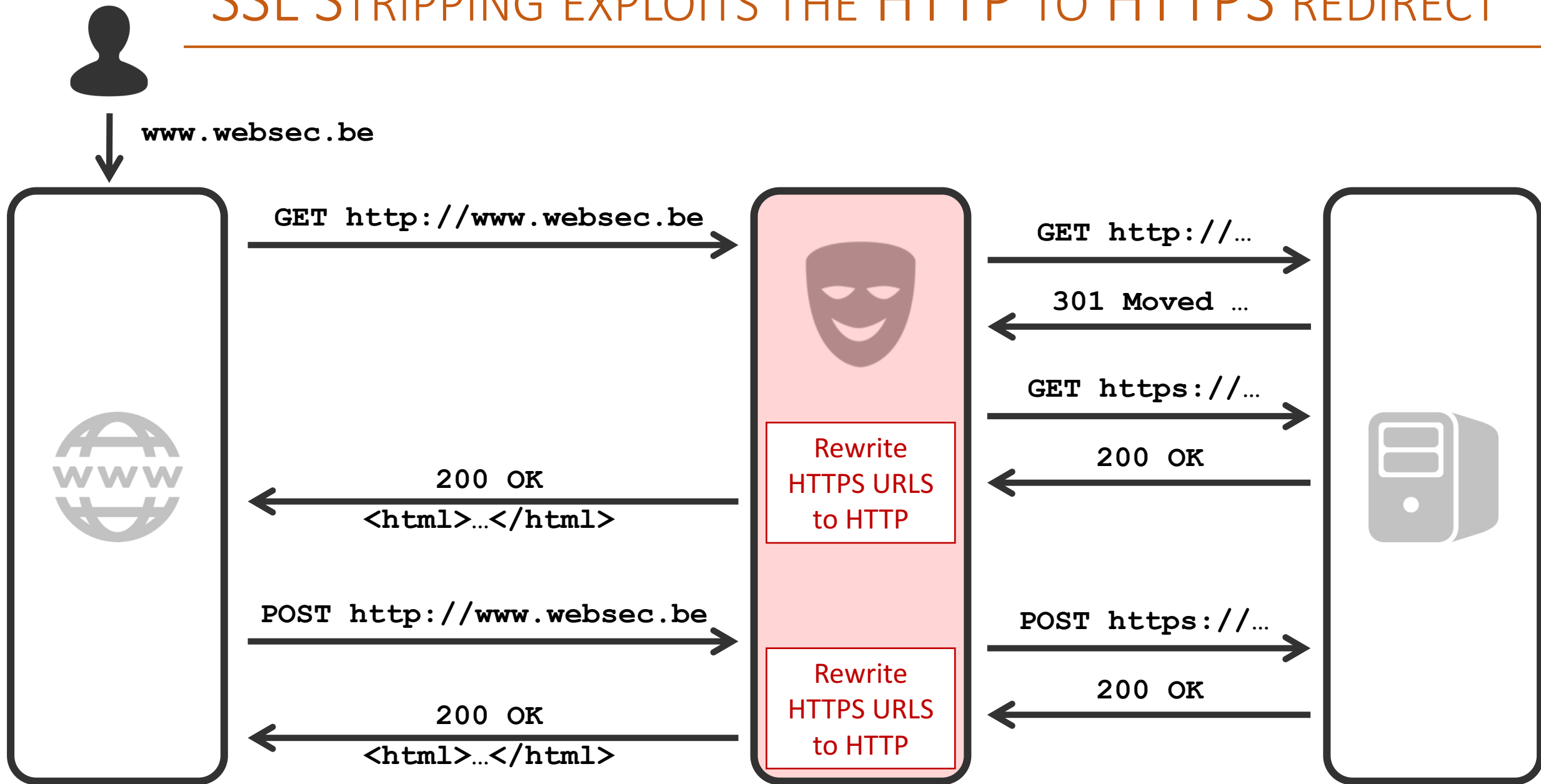
HTTP WEAKENS HTTPS SITES

95% of HTTPS servers vulnerable to trivial MITM attacks

It would be extremely difficult for the attacker to obtain a valid certificate for a domain he does not control, and using an invalid certificate would cause the victim's browser to display an appropriate warning message. Consequently, man-in-the-middle attacks against HTTPS services are hard to pull off, and often not very successful. However, there are plenty of realistic opportunities to use the unencrypted HTTP protocol to attack most HTTPS websites.

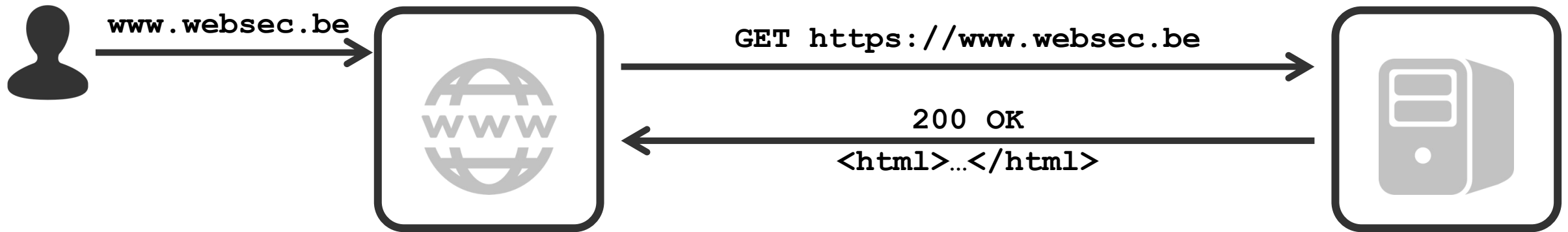
Encrypted communications are an essential requirement for banks and other financial websites, but HTTPS alone is not sufficient to defend these sites against man-in-the-middle attacks. Astonishingly, many banking websites lurk amongst the 95% of HTTPS servers that lack a simple feature that renders them still vulnerable to pharming and man-in-the-middle attacks. This missing feature is HTTP Strict Transport Security (HSTS), and only 1 in 20 secure servers currently make use of it, even though it is supported by practically all modern browsers.

SSL STRIPPING EXPLOITS THE HTTP TO HTTPS REDIRECT









STRICT TRANSPORT SECURITY AGAINST SSL STRIPPING

- Strict Transport Security converts all HTTP requests to HTTPS



- Modern browsers support HTTP Strict Transport Security (HSTS)
 - HTTP response header to enable Strict Transport Security
 - When enabled, the browser will not send an HTTP request anymore

| | | | | | | |
|------------------|---|---|---|---|---|---|
| |  |  |  |  |  |  |
| From version ... | 4 | 4 | 7 | 11 | 4.4.4 | 7.1 |

HSTS CAN BE ENABLED WITH A SIMPLE ONE-LINER

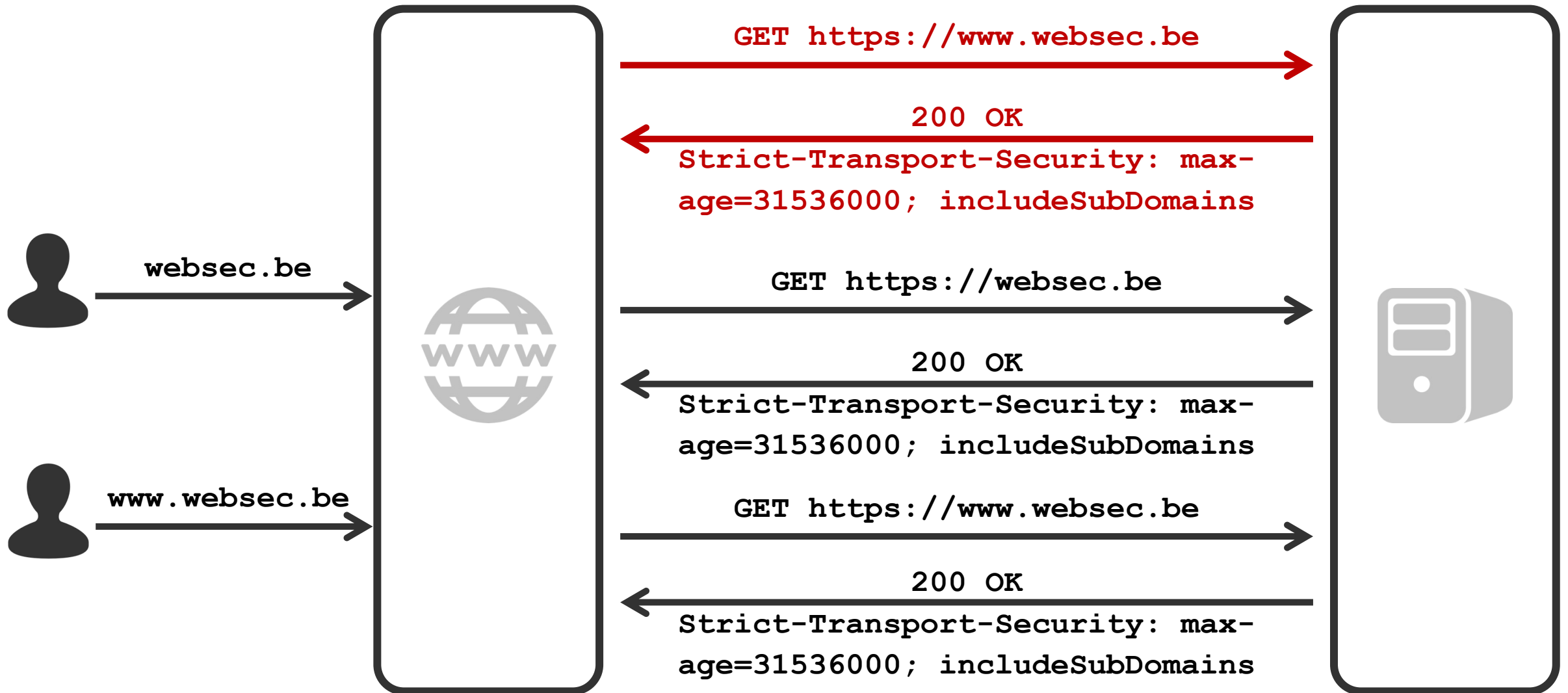
- The policy is controlled by the **Strict-Transport-Security** header
 - **max-age** specifies how long the policy should be enforced in seconds
 - Make sure this is long enough to cover two subsequent visits
 - If necessary, the policy can be disabled by setting **max-age** to 0

```
Strict-Transport-Security: max-age=31536000
```

- The policy can be extended to automatically include subdomains
 - This behavior is controlled by the **includeSubDomains** flag
 - Before enabling this, carefully analyze the services you are running on your domain

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

BUT HOW DO YOU MAKE THE FIRST CONNECTION OVER HTTPS?



PRELOADING HSTS INTO THE BROWSER

Enter a domain for the HSTS preload list:

Check status and eligibility

Enter a domain for the HSTS preload list:

Check status and eligibility

Status: websec.be is not preloaded.

Eligibility: In order for websec.be to be eligible for preloading, the errors below must be resolved:

✘ Error: No includeSubDomains directive

The header must contain the 'includeSubDomains' directive.

✘ Error: No preload directive

The header must contain the 'preload' directive.

Information

This form is used to submit domains for inclusion in Chrome's [HTTP Strict Transport Security](#) list that are hardcoded into Chrome as being HTTPS only.

Most major browsers (Chrome, [Firefox](#), Opera, Safari, [IE 11 and Edge](#)) also have HSTS support. See the [HSTS compatibility matrix](#).

Submission Requirements

If a site sends the `preload` directive in an HSTS header, it is considered for preloading. Requests submitted via the form on this site.

In order to be accepted to the HSTS preload list through this form, your site must satisfy the following requirements:

1. Serve a valid **certificate**.
2. **Redirect** from HTTP to HTTPS on the same host.
3. Serve all **subdomains** over HTTPS.
 - In particular, you must support HTTPS for the `www` subdomain if a DNS

```
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
```

<https://hstspreload.appspot.com/>

BEST PRACTICES FOR DEPLOYING HSTS

- Correctly configure HSTS on your domain
 - Attach the HSTS header to every response from the domain
 - Set the **max-age** to a sensible value, long enough to cover two subsequent visits
- Works towards preloading
 - Only include the preload flag once you're applying to be put on the list
- In case you're afraid of breaking things, deploy HSTS conservatively
 - Never set the **max-age** longer than the expiration date of the certificate
 - Decrease the **max-age** over time, until a new certificate is installed
 - Enable HSTS on subdomains first, before switching on **includeSubDomains**



Invalid Server Certificate

You attempted to reach www.google.com, but the server presented an invalid certificate.

[Back](#)

▼ [help me understand](#)

When you connect to a secure website, the server hosting that site presents your browser with something. This certificate contains identity information, such as the address of the website, which is verified by a third party. By checking that the address in the certificate matches the address of the website, it is possible to verify that you are visiting the website you intended, and not a third party (such as an attacker on your network).

In this case, the server certificate or an intermediate CA certificate presented to your browser is invalid. The malformed certificate contains invalid fields, or is not supported.

Certificate

General Details Certificate Path

Certificate path

- DigNotar Root CA
 - DigNotar Public CA 2025
 - *.google.com

[View Certificate](#)

Certificate status

This certificate is OK.

Learn more about [certificate paths](#)

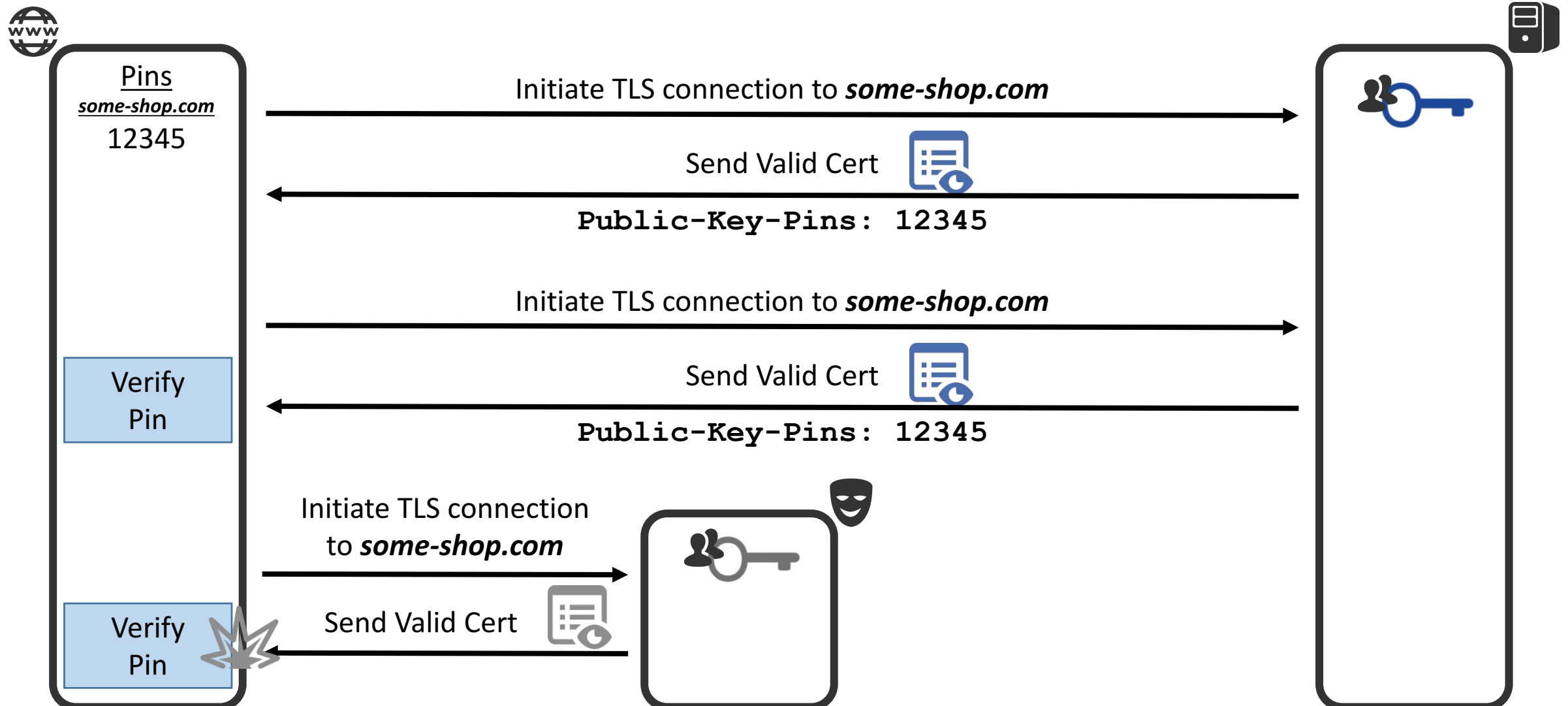
OK

HTTP PUBLIC-KEY PINNING (HPKP)


- HPKP is a server-driven, browser-enforced security policy
 - Instructs the browser to only accept a pinned public key
 - Intended to be used in combination with HSTS
- Pins associate a hostname with a cryptographic identity
 - Can be on certificate level, CA level, ...
 - Trade-off between specificity and resilience

```
Public-Key-Pins: max-age=3000;  
pin-sha256="d6qzRu9zOECb90Uez27xW1tNsj0e1Md7GkYYkVoZWmM=" ;  
pin-sha256="E9CZ9INDbd+2eRQozYqqbQ2yXLVKB9+xcprMF+44U1g="
```

HTTP PUBLIC-KEY PINNING (HPKP)



HPKP IN PRACTICE

| | | | | | | |
|------------------|---|---|---|---|---|---|
| |  |  |  |  |  |  |
| From version ... | 38 | 35 | None | None | 5 | None |

- HPKP has the potential to make sites unreachable
 - A mismatch between the key and the stored pin results in errors
 - New pins can only be set after the old pins are validated
- Precautions taken by the browsers
 - At least two pins are required to enable HPKP
 - Max-age is handled carefully (e.g. to allow gradual key migration)
- Operators should have a pinned but unused backup key

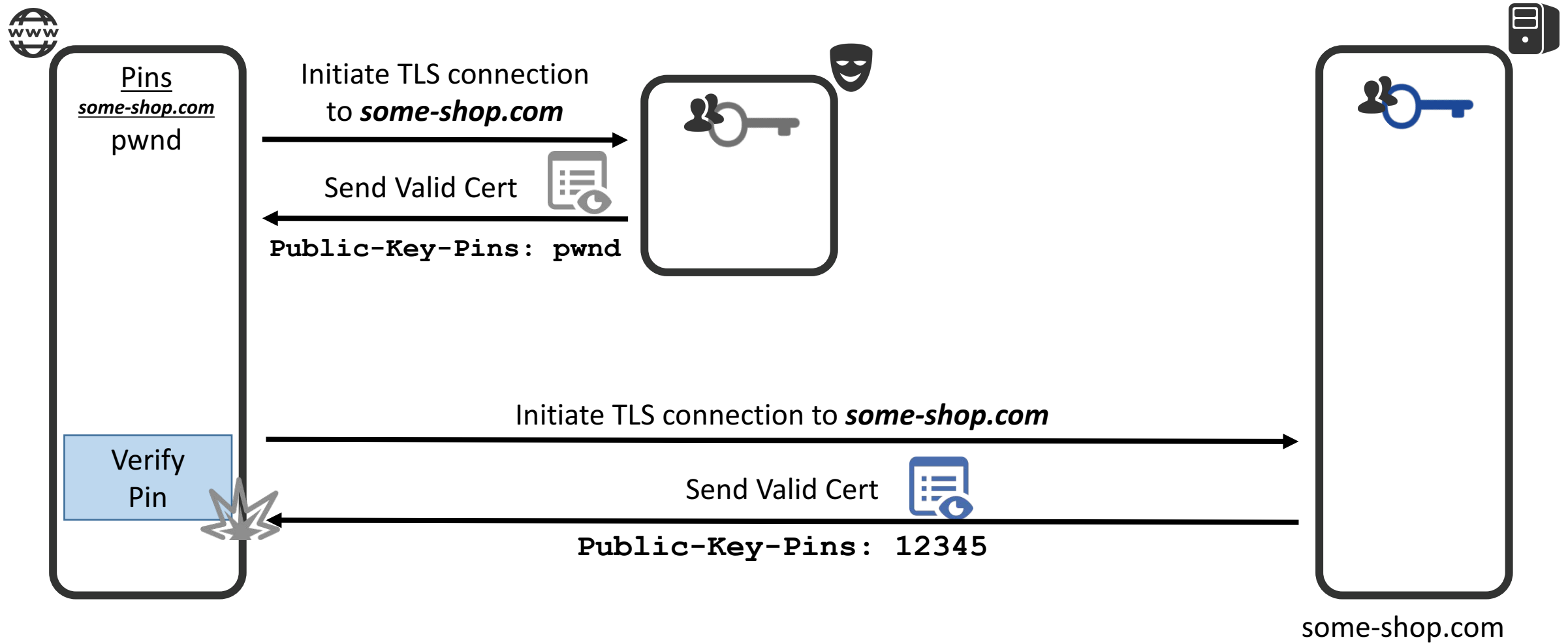
HPKP OFFERS A REPORTING MODE

```
{
  "date-time": "2014-04-06T13:00:50Z",
  "hostname": "www.example.com",
  "port": 443,
  "effective-expiration-date": "2014-05-01T12:40:50Z"
  "include-subdomains": false,
  "served-certificate-chain": [
    "-----BEGIN CERTIFICATE-----\n
    MIIEDCCAuygAwIBAgIDAjppMA0GCSqGSIb3DQEBBQUAMEIxCzAJBgNVBAYTA1VT\n
    ...
    HFa9llF7blcq26KqltyMdmKVvvBulRP/F/A8rLIQjcxz++iPAsbw+zOzlTvjwsto\n
    WHPbqCRiOwYlnQ2pM714A5AuTHhdUDqB1O6gyHA43LL5Z/qHQF1hwFGPa4NrZQU6\n
    yuGnBXj8ytqU0CwIPX4WecigUCAkVDNx\n
    -----END CERTIFICATE-----",
    ...
  ],
  "validated-certificate-chain": [
    "-----BEGIN CERTIFICATE-----\n
    MIIEDCCAuygAwIBAgIDAjppMA0GCSqGSIb3DQEBBQUAMEIxCzAJBgNVBAYTA1VT\n
    ...
    HFa9llF7blcq26KqltyMdmKVvvBulRP/F/A8rLIQjcxz++iPAsbw+zOzlTvjwsto\n
    WHPbqCRiOwYlnQ2pM714A5AuTHhdUDqB1O6gyHA43LL5Z/qHQF1hwFGPa4NrZQU6\n
    yuGnBXj8ytqU0CwIPX4WecigUCAkVDNx\n
    -----END CERTIFICATE-----",
    ...
  ],
  "known-pins": [
    'pin-sha256="d6qzRu9zOECb90Uez27xWltNsjoelMd7GkYYkVoZWmM="',
    "pin-sha256=\"E9CZ9INDbd+2eRQozYqqbQ2yXLVKb9+xcprMF+44Ulg=\""
  ]
}
```

HPKP OFFERS A REPORTING MODE

- HPKP also supports a report-only mode
 - The pins will be checked, but the connection will not be blocked
 - Report will be sent to the end point
 - Great for testing your policy before breaking your entire website
- The header allows the specification of a reporting endpoint
 - When the browser detects a violation, it will send a report
 - Great for actively detecting problems
 - Do not send reports to your pinned domain, should it become blocked ...
- **Report-uri.io** is a freely available service to collect your reports

WHAT CAN GO WRONG WITH HPKP?



DEALING WITH HOSTILE PINNING

- Has been coined as HPKP Suicide or RansomPKP
 - Concerns scenarios where your server is compromised
 - Pins are served to your users, and this cannot be easily undone
- Hostile pinning is a difficult problem to solve
 - Spec suggests that browsers limit the duration of max-age
 - Use complementary solutions like Certificate Transparency
- Browsers have a minimal preload list
 - Contains large and important sites (Google, Mozilla, Twitter, ...)
 - Adding your own sites does not seem to be possible
 - The update process is quite slow too, which is suboptimal for key pins

BEST PRACTICES FOR DEPLOYING HSTS

- Enable in report-only mode to gain insights into attacks
 - Detect rogue keys in use
 - Get insights into the behavior of middleboxes

- Read about the problems that arise when enabling blocking mode
 - Map out all the scenarios up front
 - Start with a small max-age
 - Keep your key material in a safe place, even after your certs have expired



Hands-on Lab Part 2

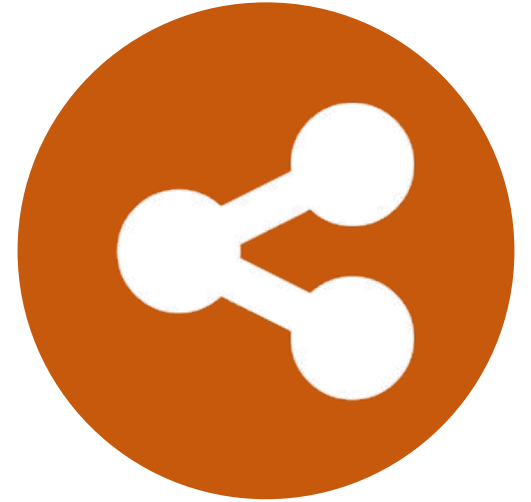
NOW IT'S UP TO YOU ...



Secure



Follow



Share

Web Security Essentials

April 24 – 25, Leuven, Belgium

<https://essentials.websec.be>