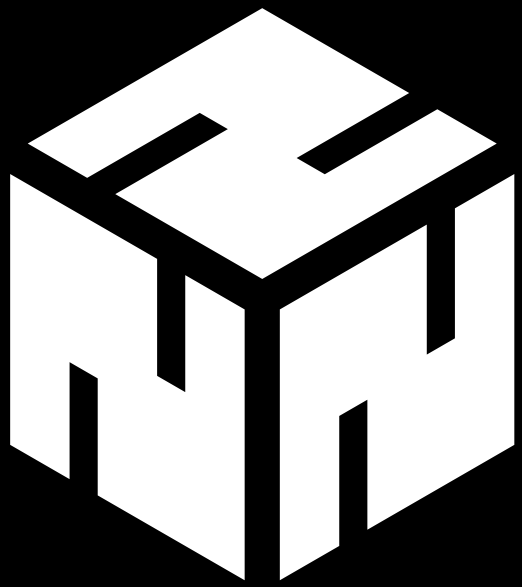


Services absichern

Common Practice

Aleksander Paravac
watz@nerd2nerd.org

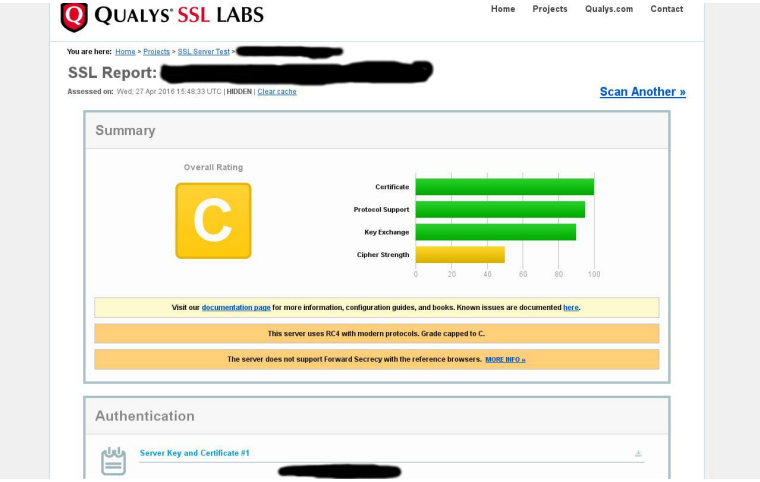
#gpn16

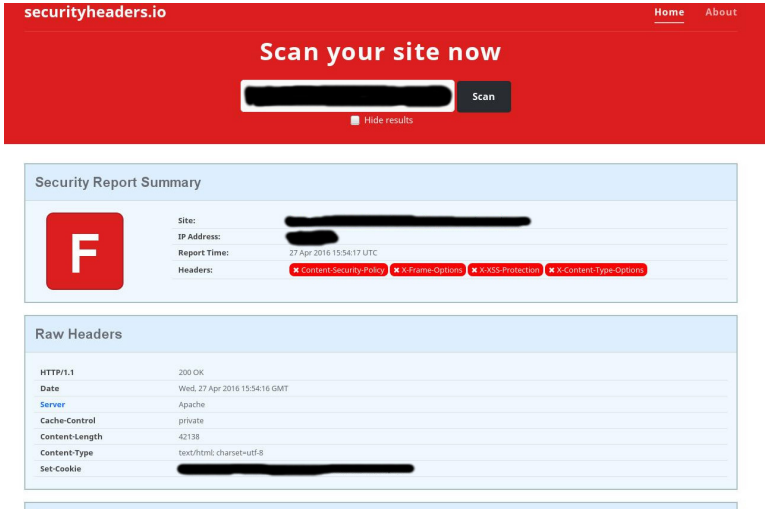
Übersicht

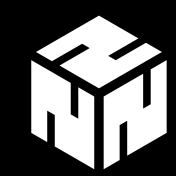
Übersicht

Motivation

Es gibt solche Webseiten ...

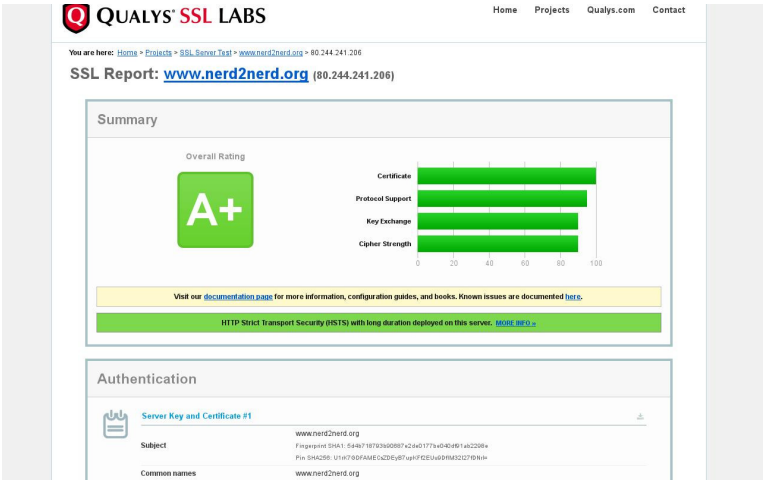


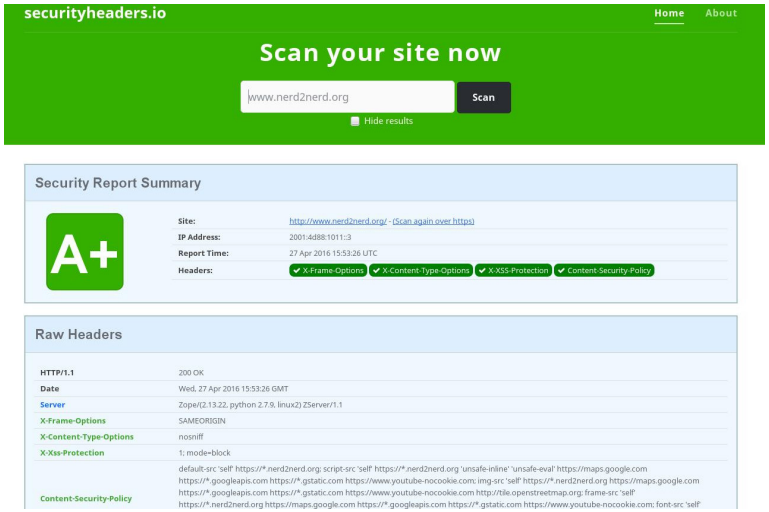




Motivation

…und solche Webseiten





Übersicht

Übersicht

HTTP

Extra Header für mehr Sicherheit

- **X-Frame-Options**
  Bietet Schutz gegen Clickjacking Attacken
- **X-Content-Type-Options**
  Verhindert MIME-Type sniffing
- **X-Xss-Protection**
  Aktiviert den in modernen Browsern eingebauten XSS Schutz
- **X-Content-Type-Options**
  Regelt woher Objekte (CSS, Bilder, Skripte, etc.) nachgeladen werden dürfen

Apache 2.4

```
Header always set X-Frame-Options "SAMEORIGIN"
Header always set X-Content-Type-Options "nosniff"
Header always set X-Xss-Protection "1; mode=block"
```

X-Frame-Options

- DENY
- SAMEORIGIN
- ALLOW-FROM https://example.com

X-Xss-Protection

- 0 # Aus
- 1; mode=block

HTTP

Content-Security-Policy

```
Header always set Content-Security-Policy \
"default-src 'self' https://*.nerd2nerd.org; \
script-src 'self' https://*.nerd2nerd.org '\
unsafe-inline' 'unsafe-eval; \
img-src 'self' https://*.nerd2nerd.org; \
frame-src 'self' https://*.nerd2nerd.org; \
font-src 'self' \
style-src 'self'\
object-src 'none'"
```

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

Content-Security-Policy

Abbildung: Debug der Optionen[1] für Content-Security-Policy.

[1] Liste der Optionen bei Scott Helme

Weiterführender Talk

More Secure Websites

17:30h, ZKM-Vortragssaal

Ein Vortrag von **Ives Laaf** auf der GPN16.

> *Sichere Webanwendungen und was man zusätzlich tun kann. Überblick über die aktuellen Versionen von HTTP-Headern und Tools zum Testen und erstellen.*

Beast

LogJam

Poodle

HTTPS

HTTP-Header, die zweite. SSL only!

- **Strict-Transport-Security**
  User Agent dazu zwingen TLS zu benutzen
- **Public-Key-Pins**
  Versucht SSL MiTM Attacken zu verhindern

HTTPS

Strict-Transport-Security

```
Header always set Strict-Transport-Security \
"max-age=31536000; includeSubDomains; preload"
```

HTTPS

Public-Key-Pins

```
Header always set Public-Key-Pins '\
pin-sha256="X3pGTSOuJeEVw989IJ/...="; \
pin-sha256="MHJYVThihUrJcxW6wcq...="; \
pin-sha256="isi41AizREkLvvft0IR...="; \
max-age=10';
```

Debug: `Public-Key-Pins-Report-Only`

Fingerprint über:

```
openssl req -pubkey < yourcert.pem |
openssl pkey -pubin -outform der |
openssl dgst -sha256 -binary | base64
```

Apache 2.4

```
SSLEngine on
SSLProtocol all -SSLv2 -SSLv3
SSLHonorCipherOrder on
SSLCompression off

SSLDHParametersFile /path/to/dh.pem

SSLCipherSuite "ECDH+AESGCM:DH+AESGCM:ECDH+AES256:\
DH+AES256:ECDH+AES128:DH+AES:ECDH+3DES:DH+3DES:\
RSA+AESGCM:RSA+AES:RSA+3DES:!aNULL:!MD5:!DSS"
```

Übersicht

SMTP

Postfix

```
# Enable TLS/SSL
smtpd_use_tls = yes
[..]

# Select strong ciphers
smtpd_tls_loglevel = 0
smtpd_tls_dh1024_param_file =
  /etc/postfix/ssl/dh_2048.pem
smtpd_tls_dh512_param_file =
  /etc/postfix/ssl/dh_512.pem
smtpd_tls_eecdh_grade = strong
smtpd_tls_mandatory_protocols = !SSLv2,!SSLv3
smtpd_tls_mandatory_ciphers = high
tls_ssl_options = NO_COMPRESSION
```

Postfix Cipher List

```
tls_preempt_cipherlist = yes
tls_high_cipherlist = EDH+CAMELLIA:EDH+aRSA:\
EECDH+aRSA+AESGCM: EECDH+aRSA+SHA384: \
EECDH+aRSA+SHA256:EECDH:+CAMELLIA256:+AES256: \
+CAMELLIA128:+AES128:+SSLv3:!aNULL:!eNULL: \
!LOW:!3DES:!MD5:!EXP: !PSK:!DSS:!RC4:!SEED: \
!ECDSA:CAMELLIA256-SHA:AES256-SHA: \
CAMELLIA128-SHA:AES128-AES
```

Postfix und DANE

```
# Enable DANE
smtp_dns_support_level = dnssec
smtp_tls_security_level = dane
smtp_tls_loglevel = 1
```

SMTP

Postfix und SASL Auth

```
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
broken_sasl_auth_clients = yes
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
```

Weitere DNS Geschichten

- DKIM (DomainKey Identified Mail)
- SPF (Sender Policy Framework)

Übersicht

Dovecot SSL

```
ssl = yes
ssl_protocols = !SSLv3 !SSLv2
ssl_dh_parameters_length = 2048
ssl_options = no_compression

ssl_cert = </etc/ssl/private/yourcert.pem
ssl_key = </etc/ssl/private/yourkey.key

ssl_cipher_list = EDH+CAMELLIA:EDH+aRSA:\
EECDH+aRSA+AESGCM:EECDH+aRSA+SHA384:\
EECDH+aRSA+SHA256:EECDH:+CAMELLIA256:\
+AES256:+CAMELLIA128:+AES128:!SSLv2:\
!aNULL:!eNULL:!LOW:!3DES:!MD5:!EXP:!PSK:!DSS:\
!RC4:!SEED:!ECDSA:CAMELLIA256-SHA:AES256-SHA:\
CAMELLIA128-SHA:AES128-AES
```

Übersicht

Ejabber

ejabberd.yml - Client to Server

```
listen:
- port: 5222
  module: ejabberd_c2s
  ciphers: "[...]"
  starttls: true
  starttls_required: true
  certfile: "/etc/ssl/ejabberd/jabber.pem"
  dhfile: "/etc/ssl/ejabberd/jabber.pem.dh"
  tls_compression: false
  protocol_options:
    - "no_sslv3"
    - "no_tlsv1"
    - "cipher_server_preference"
```

Ejabber

ejabberd.yml - Server to Server

```
-
  port: 5269
  module: ejabberd_s2s_in
  s2s_use_starttls: required_trusted
  s2s_certfile: "/etc/ssl/ejabberd/jabber.pem"
  s2s_dhfile: "/etc/ssl/ejabberd/jabber.pem.dh"
  s2s_protocol_options:
  - "no_sslv3"
  - "no_tlsv1"
  - "cipher_server_preference"
  s2s_ciphers: "[...]"
```

Übersicht

IRC

Unrealircd

```
set {
  ssl {
    certificate yourcert.crt;
    key yourkey.key;
    trusted-ca-file ca_chain.pem;
    renegotiate-bytes "64m";
    renegotiate-time "10h";
    server-cipher-list "[...]";
  };
};
```

Übersicht

Updates - Updates und Updates

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

Forthcoming OpenSSL releases
============================

The OpenSSL project team would like to announce the forthcoming release of OpenSSL versions 1.0.2h, 1.0.1t.

These releases will be made available on 3rd May 2016 between approximately 1200-1500 UTC. They will fix several security defects with maximum severity "high".

Abbildung: OpenSSL Update[1].

[1] Matthew Green auf Twitter

Empfohlene Cipher Liste

```
SSLCipherSuite "EDH+CAMELLIA:EDH+aRSA:\
EECDH+aRSA+AESGCM:EECDH+aRSA+SHA256:EECDH:\
+CAMELLIA128:+AES128:+SSLv3:!aNULL:!eNULL:\
!LOW:!3DES:!MD5:!EXP:!PSK:!DSS:!RC4:!SEED:\
!IDEA:!ECDSA:kEDH:CAMELLIA128-SHA:AES128-SHA"
```

Cipher - Cipher - Cipher

Diff

```
> TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
> TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
> TLS_RSA_WITH_3DES_EDE_CBC_SHA
> TLS_RSA_WITH_AES_128_CBC_SHA256
> TLS_RSA_WITH_AES_128_GCM_SHA256
> TLS_RSA_WITH_AES_256_CBC_SHA
> TLS_RSA_WITH_AES_256_CBC_SHA256
> TLS_RSA_WITH_AES_256_GCM_SHA384
< TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA
< TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA
< TLS_RSA_WITH_CAMELLIA_128_CBC_SHA
```

Nonces

golem.de IT-NEWS FÜR PROFIS

HOME TICKER VIDEO Suchen

TOP-THEMEN: Google I/O Anleitung Auto Security Test Vorgelesen mehr...

SERVICES: PREISVERGLEICH STELLENMARKT TOP-ANGEBOTE ABO

TLS/GCM

Gefahr durch doppelte Nonces

Moderne TLS-Verbindungen nutzen üblicherweise das AES-GCM-Verschlüsselungsverfahren. Das benötigt einen sogenannten Nonce-Wert, der sich nicht wiederholen darf. Ansonsten ist die Sicherheit dahin.

Die AES-Verschlüsselung im sogenannten Galois/Counter-Mode (GCM) hat sich für TLS-Verbindungen weitgehend durchgesetzt. Der Grund dafür ist, dass alle anderen Verschlüsselungsmodi, die von TLS unterstützt werden, in der Vergangenheit zahlreiche Sicherheitslücken hatten. Doch auch GCM ist nicht frei von Problemen. In einem Forschungsprojekt, an dem auch der Autor dieses Artikels beteiligt war, wurden Probleme bei der Generierung des sogenannten Nonce-Wertes untersucht. Eine Reihe von Webservern nutzt sich wiederholende Nonce-Werte, eine gravierende Sicherheitslücke. Betroffen sind unter anderem Webseiten des Kreditkartenkonzerns Visa.

Fremder Javascript-Code auf der dänischen Webseite von Visa - ein Angriff auf die fehlerhafte GCM-Verschlüsselung (Bild: Screenshot)

Artikel:	TLS/GCM Gefahr durch doppelte Nonces
Inhalt:	· Forbidden Attack
Datum:	20.5.2016, 08:05
Autor:	Hanno Böck
Themen:	SSL, Chacha20, Sicherheitslücke, Verschlüsselung, IBM, Server, Technologie, Applikationen, Internet, Security

Abbildung: Forbidden Attack[1]

[1] Golem vom 20.05.2016

Referenzen

Danke an

- https://bettercrypto.org/
- https://www.ssllabs.com/
- https://securityheaders.io/
- Scott Helme