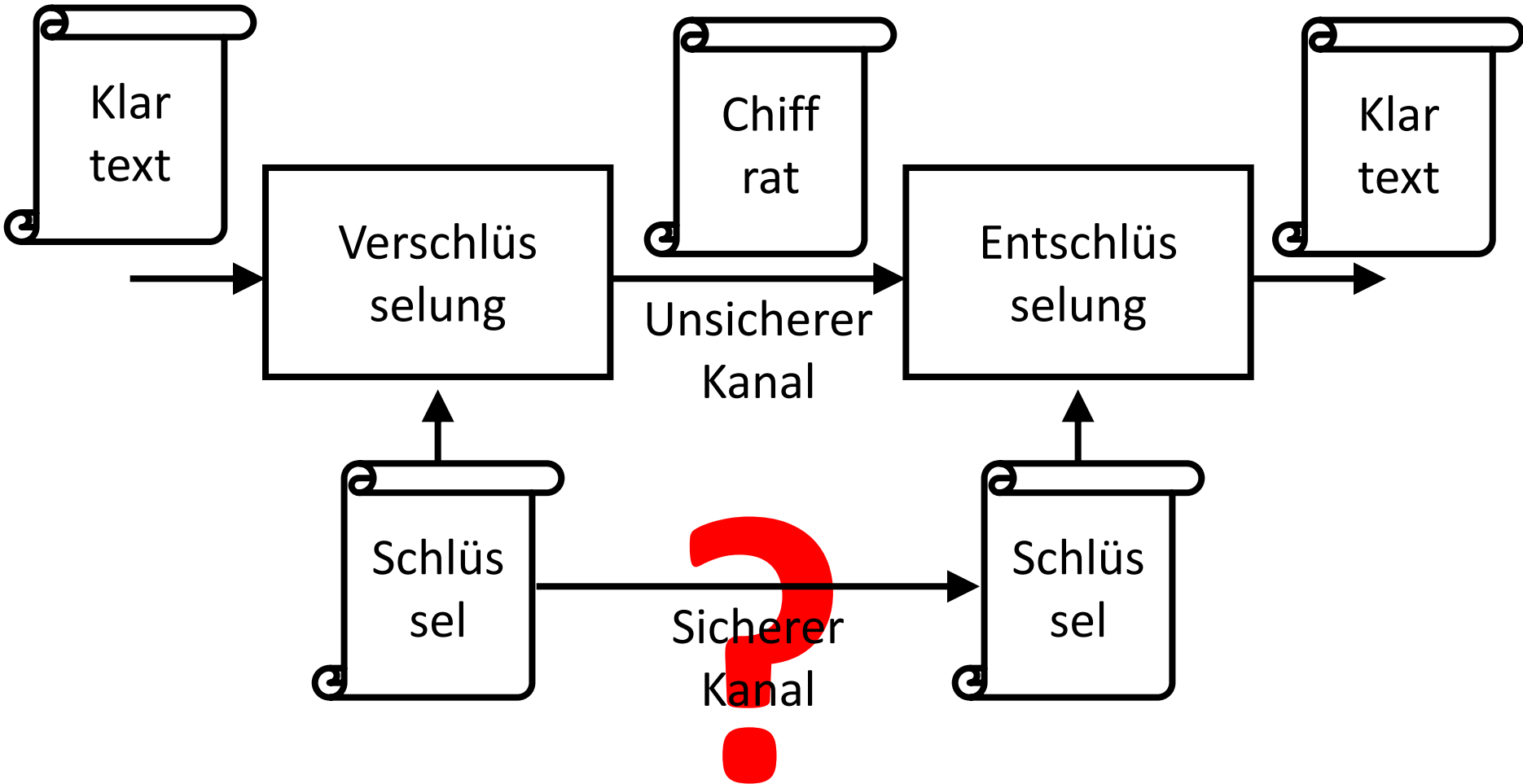


Betriebssysteme und Sicherheit

– Sicherheit

Schlüsselaustausch

Schlüsselaustausch



Problem

- Bevor man Verschlüsselung oder Signaturverifikation verwenden kann, muss man einen Schlüssel austauschen
 - Symmetrische Verschlüsselung: vertraulich
 - Asymmetrische Verschlüsselung: unverändert
- Wie kann man diesen Schlüsselaustausch mittels eines off-line, a priori Kanals ermöglichen?

Kerberos

- Abgeleitet vom Needham-Schroeder Protokoll
- Viele Versionen (Fehlerkorrekturen)
- Erstes Schlüsselaustauschprotokoll
- Verwendet nur symmetrische Verschlüsselung
- ABER: verwendet vertrauenswürdige dritte Partei, die mit allen Parteien X einen gemeinsamen, symmetrischen Schlüssel $E_X()$ hat

Kerberos

B

A

S

r, A, B



$E_A(r, B, K, L), E_B(K, A, L)$



$E_B(K, A, L), E_K(A, T)$



$E_K(T+1)$



Diffie-Hellman

- Erstes Public Key Protokoll (Ursprung aller Public Key Forschung)
- Ermöglicht den Austausch eines symmetrischen Schlüssels
- Reduziert die Anzahl notwendiger symmetrischer Schlüssel von $O(n^2)$ zu $O(n)$ öffentlicher Schlüssel
- Unmodifiziert nur gegen passive Angreifer sicher

DH-Keys

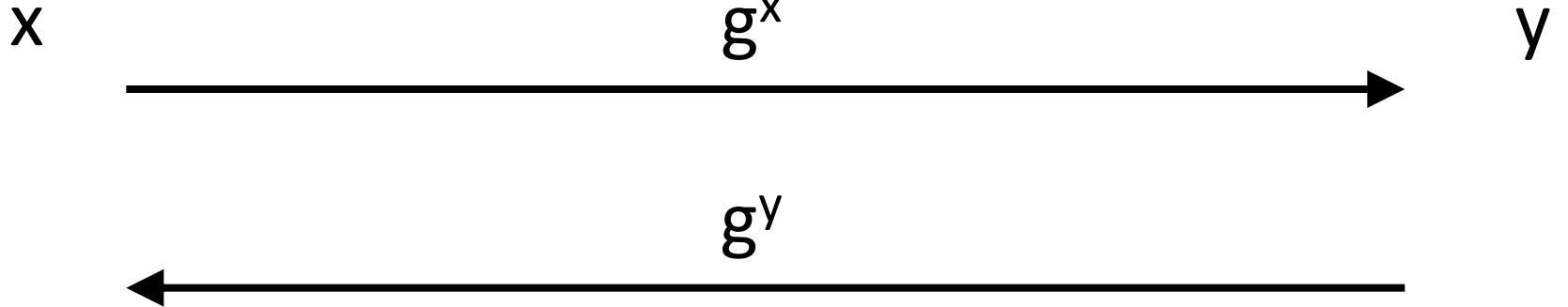
Private key:

x

Public key:

g^x

DH-Protokoll

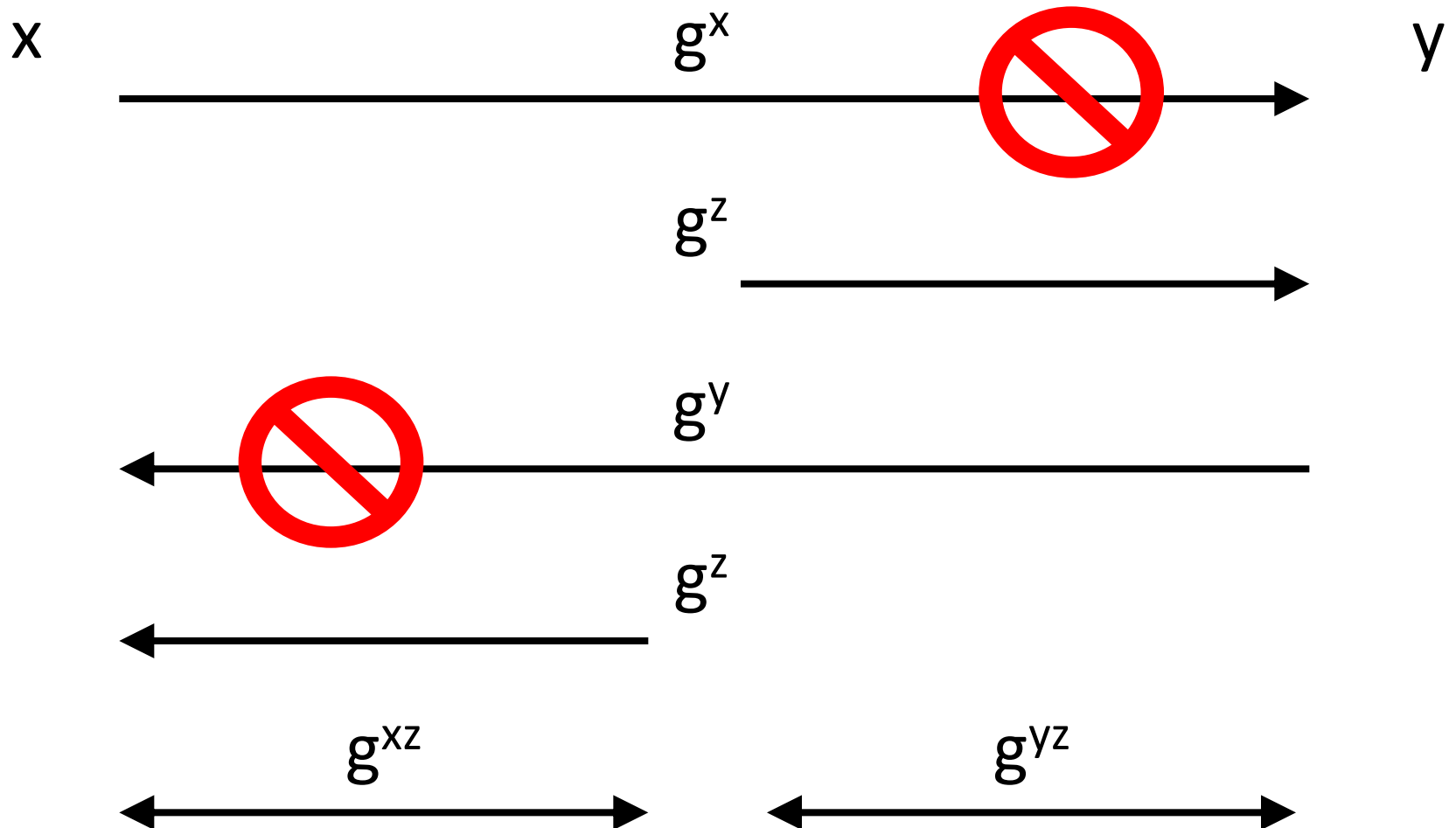


Symmetric Key: g^{xy}

Wieso ist DH sicher ?

- Mitleesen von g^x, g^y
 - Schwer x zu berechnen (Diskreter Logarithmus)
 - Schwer y zu berechnen
 - Gegeben g^x, g^y ist es schwer g^{xy} zu berechnen
- ⇒ „Computational Diffie-Hellman Problem“
- ⇒ Sicherer Schlüsselaustausch

Man-in-the-Middle Attack



SSL

- Entwickelt von Netscape zum sicheren Web surfen
- Zwei (Unter-) Protokollschichten auf Session Ebene
 - Verschlüssele TCP Verkehr
 - Schlüsselaustausch
- Verwendet Zertifikate für den Schlüsselaustausch

Zertifikat

Ein Zertifikat besteht aus

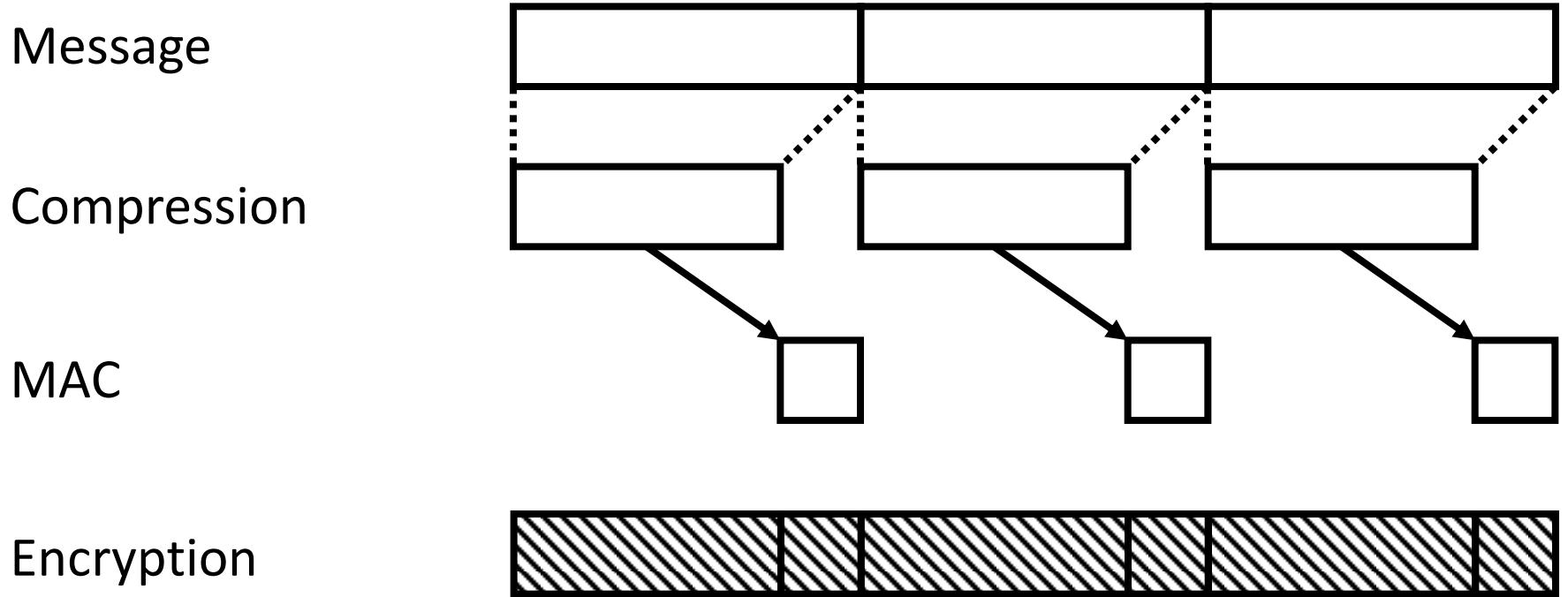
- einer Identität
- einem öffentlichen Schlüssel
- der Identität des Signierers
- einer Signatur

$A, \text{PubKey}_A, CA, S_{CA}(A, \text{PubKey}_A, CA)$

SSL

- Verwendet PKI um „sicher“ über das Internet zu kommunizieren
- SSL (Secure Sockets Layer) / TLS (Transport Layer Security) verschlüsselt Daten auf Session Schicht (d.h. über der Transport Schicht / TCP)
- SSL verschlüsselt einen Strom von Daten und verifiziert die Authentizität des Servers (Defaulteinstellung)

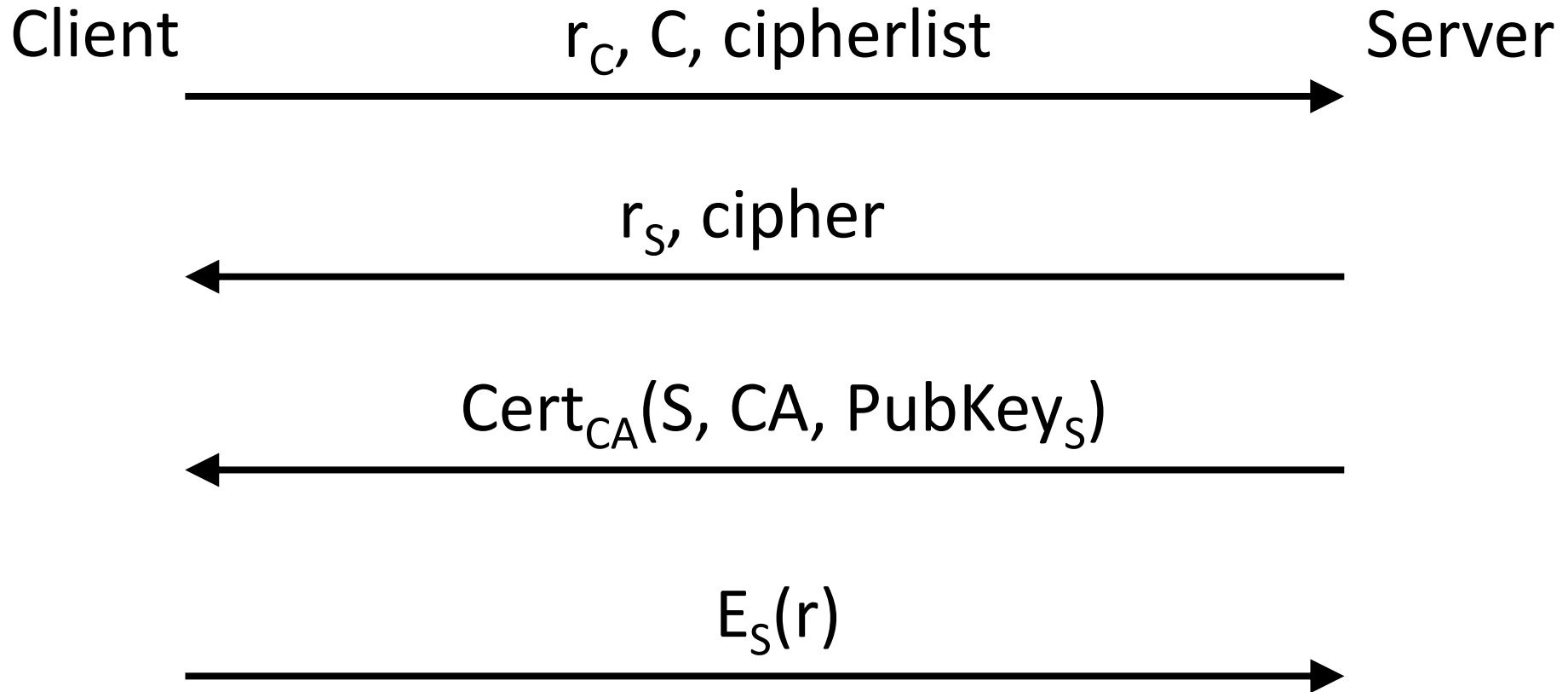
SSL Encryption



SSL

- Verschlüsselung und MAC sind mittels symmetrischer Kryptographie realisiert (aus Geschwindigkeitsgründen)
- Die Schlüssel für Verschlüsselung und MAC werden durch Public Key Cryptography ausgetauscht

SSL Authentication (Default)



Schlüsselberechnung

- Die symmetrischen Schlüssel für Verschlüsselung und MAX werden aus r , r_c und r_s berechnet (mittels padded hash functions)
- Die Identität des Servers ist sein DNS Name (z.B. `www.tu-dresden.de`)
- Eine SSL abgesicherte Verbindung garantiert, dass der Client mit dem Server verbunden ist, dessen Name im Browser angezeigt wird (ohne das jemand die Daten mitlesen kann)

SSL Client Authentication

·
·
·

$\text{Cert}_{CA}(C, CA, \text{PubKey}_C)$



$E_S(r), S_C(r, \dots)$



.... – bisheriger Protokolllauf

Wie greife ich SSL an?

- Auf dem Client
 - Malware (Trojan Horse)
 - Wo endet die SSL Verbindung?
- Chosen plaintext attack
 - Mache (cross-site) Anfragen mit (überwiegend) bekannten URL/cookie Informationen im Browser
 - Beobachte Netzwerkverkehr
 - Vergleiche mit „challenge“ (zu erratendem Klartext)