

SIT / SRN PROJEKTPRÄSENTATION

Vortragende:

David Seemann

Jochen Schwander

Marcel Math

Phil-Patrick Kwiotek

GLIEDERUNG

- Herangehensweise
- Registrierung
- Login
- Sicherheitsvorkehrungen

—— Registrierung ——

Desktop
Client

Benutzername
Desktop-Passwort
Web-Passwort



encrypt
AES



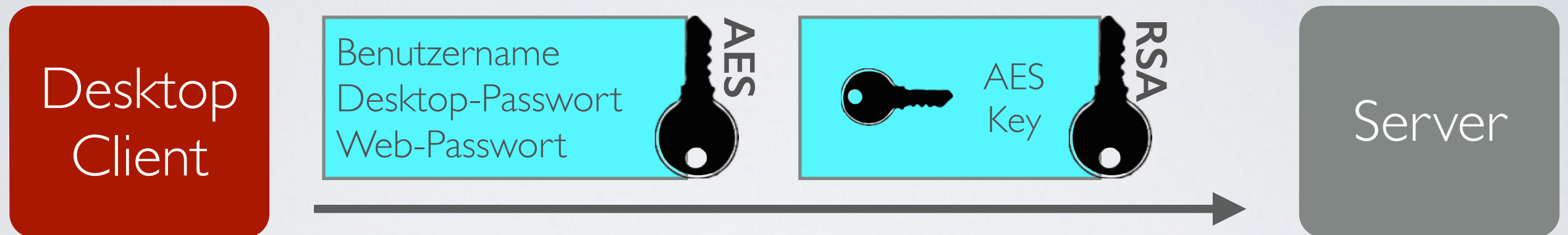
AES
Key



encrypt
RSA

- Der Desktop Client wählt Username und Passwörter und verschlüsselt diese mit AES
- Der AES-Schlüssel wird mit RSA verschlüsselt

— Registrierung —



- Desktop Client sendet Passwörter & Benutzername mit Public-Key verschlüsselt an Server

— Registrierung —



- Server entschlüsselt Benutzername und Passwörter
- Server prüft ob Benutzername noch verfügbar ist

—— Registrierung ——

Server

Salt

$h(\text{Web-Passwort} + \text{Salt})$

$h(\text{Desktop-Passwort} + \text{Salt})$

- Server generiert Salt
- Server erstellt Salted-Hash von Passwörtern

— Registrierung —



- Server legt Benutzer in Datenbank

—— Login ——

Desktop
Client

a, g, p

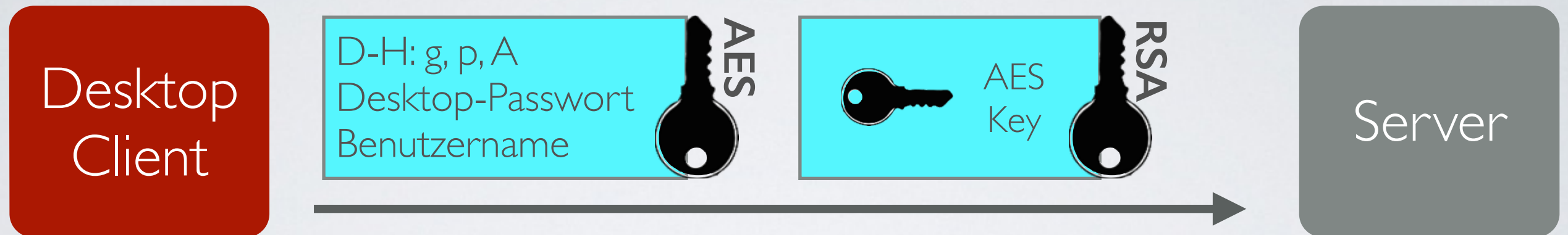
$A = g^a \bmod p$

Desktop-Passwort

Benutzername

- Der Desktop Client beginnt Diffie-Hellman-Schlüsselaustausch

—— Login ——



- Desktop Client sendet D-H-Parameter, Desktop-Passwort & Benutzername mit AES verschlüsselt und den mit dem Public-Key verschlüsselten AES-Key

—— Login ——

Server

Private Key  AES Key  D-H: g, p, A
Desktop-Password
Benutzername

- Server entschlüsselt D-H-Parameter, Desktop-Password & Benutzername

—— Login ——

überprüfe:

$$\mathbf{x} \stackrel{?}{=} \mathbf{h}(\mathbf{Desktop-Passwort} + \mathbf{Salt})$$

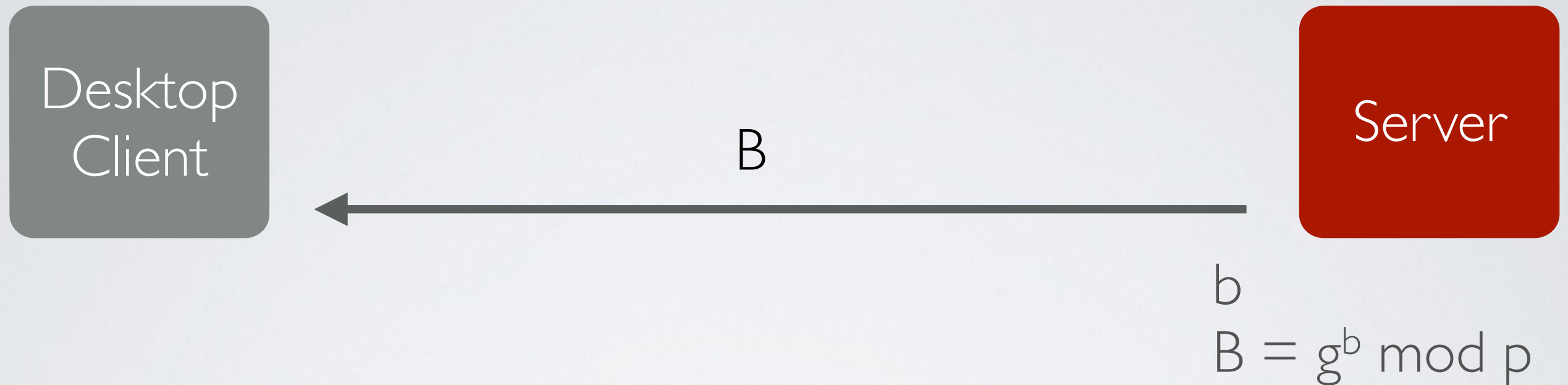
Server

DB

$$\mathbf{h}(\mathbf{Desktop-Passwort} + \mathbf{Salt}) = \mathbf{x} \quad \mathbf{Salt}$$

- Server holt sich Benutzerdaten aus der Datenbank und hinterlegt diese für später.
- Server überprüft ob Hash von Desktop-Passwort dem Datenbankeintrag für diesen User übereinstimmt

—— Login ——



- Server setzt Diffie-Hellman-Schlüsselaustausch fort und sendet Desktop Client sein B

—— Login ——



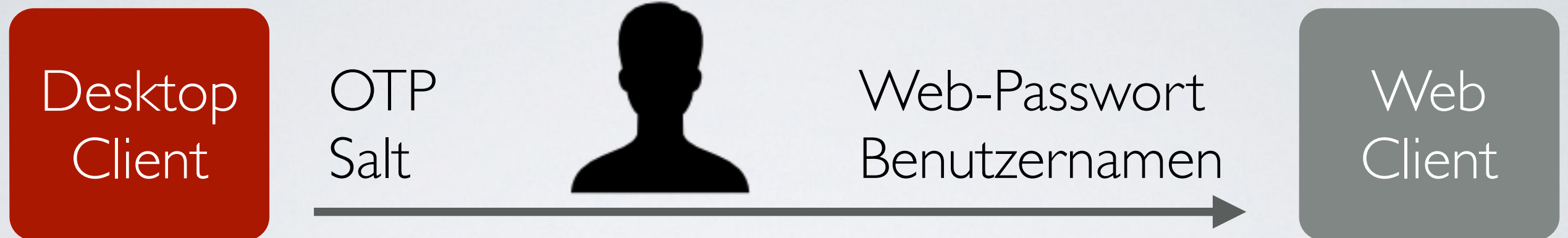
- Server & Desktop Client berechnen K , der geheime Schlüssel
- Server & Desktop Client starten AES-verschlüsselten Stream.
 K dient dabei als geheimer Schlüssel.

—— Login ——



- Server berechnet One Time Password (OTP) und schickt es zusammen mit dem Salt aus der Datenbank an den Desktop Client

—— Login ——



- Desktop Client zeigt Benutzer OTP & Salt an
- Der **Benutzer** fügt OTP & Salt im Web Client ein
- Der **Benutzer** gibt zusätzlich sein Web-Passwort und seinen Benutzernamen ein

—— Login ——



Web
Client

$h (\text{OTP} , h (\text{Web-Passwort} , \text{Salt}))$

Server

- Der Web Client berechnet den Hash aus dem Web-Passwort und dem Salt
- Der Web Client berechnet den Hash davon und dem OTP
- Der Web Client sendet diesen Hash an den Server

—— Login ——

überprüfe:

$h(\textcolor{red}{x} + \text{OTP}) \stackrel{?}{=}$

$h(h(\text{Desktop-Passwort} + \textcolor{red}{Salt}) + \text{OTP})$

Server

DB

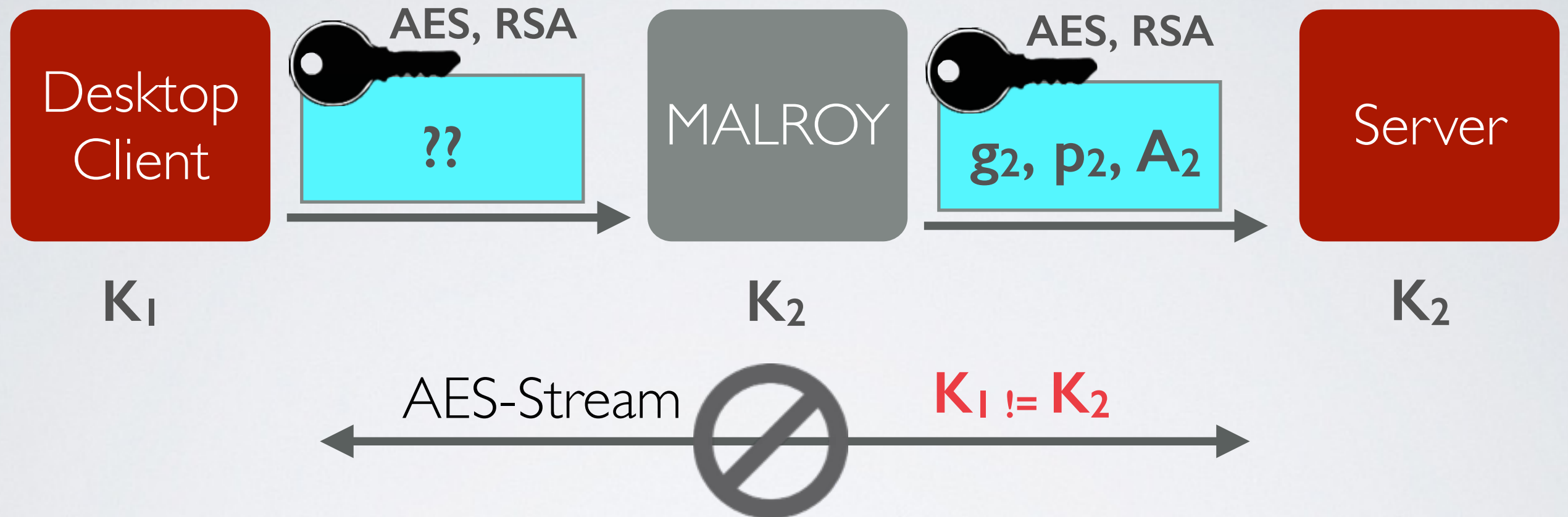
$h(\text{Web-Passwort} + \text{Salt}) = \textcolor{red}{x}$

$\textcolor{red}{Salt}$

- Der Server vergleicht den Hash mit dem Hash aus Datenbank und OTP

Sicherheitsvorkehrungen

Man in the Middle



- Ein **Man in the Middle Angriff** ist nicht möglich, da der Desktop-Client den AES-Stream mit einem K verschlüsselt!

—— Sicherheitsvorkehrungen ——

Brute Force

Brute Force Angriffe sind nicht möglich, da der Server nur je 3 Loginversuche zulässt und danach der User sperrt.

Der User muss danach über Telefon / den Post-Weg neu aktiviert werden.

—— Login ——

Der Benutzer ist vollständig authentifiziert!

nun folgt die Live Demo

