

5. Übungszettel

Abgabe bis Dienstag, 17. Mai 2022 – 16:00 Uhr

Besprechung: Dienstag, 17. Mai 2022

Abgabe in festen Gruppen (Namen + Matrikelnummern angeben)

Abgabe via Artemis: <https://alpro.besec.uni-bonn.de>

Projekt 1 (2 + 16 + 2 Punkte)

- a) Erstellen Sie zwei RSA-Schlüsselpaare und jeweils selbst signierte Zertifikate.
- b) Entwickeln Sie in C / C++ / Rust / Go (keine Skriptsprachen) eine Client-/Server-Anwendung, die folgende funktionale Eigenschaften erfüllt:
 - 1) Client und Server kommunizieren TLS-verschlüsselt über TCP und verwenden jeweils einen der Schlüssel.
 - 2) Nach dem erfolgreichen Verbindungsaufbau senden sich Client und Server jeweils den SHA256-Fingerprint des öffentlichen Schlüssels des Kommunikationspartners und geben den empfangenen Fingerprint hexadezimal kodiert auf der Standardausgabe aus.
 - 3) Anschließend wird die Verbindung beendet, der Server wartet dann auf die nächste Verbindung, der Client wird beendet.
- c) Erstellen Sie ein Makefile für Ihr Projekt. Geben Sie alle Quelldateien in Ihrer Abgabe mit ab.

Hinweise (nur für C und OpenSSL):

- 1) Sie benötigen von OpenSSL nur die folgenden Headerdateien:
`<openssl/ssl.h>`
`<openssl/err.h>`
`<openssl/x509.h>`
- 2) Aktivieren Sie die Abfrage des Zertifikats der Gegenseite mit:
`SSL_CTX_set_verify(ctx, SSL_VERIFY_PEER, verify_callback);`
(Die Callbackfunktion „verify_callback“ muss dabei immer den Wert 1 zurückliefern.)
- 3) Die Digest-Länge von SHA256 ist 32 Bytes. Reservieren Sie für die hexadezimale Darstellung 65 Zeichen, initialisieren Sie diese bereits mit ‘\0’:
`char hexfp[65];`
`memset(hexfp, '\0', 65);`
- 4) Je nach System benötigen Sie die libssl und libcrypto:
`gcc -l ssl -l crypto ...`