Aufgabe 2
Netzwerk- und Kryptopraktikum
Sommersemester 2024

Aufgabenstellung

Auf `nkp.mhgb.net:10001` wartet ein Server auf Anfragen. Nach einer erfolgreichen Authentifizierung am Server erhältst du eine Flag.

Teilaufgabe 1

In dieser Aufgabe wirst du mit der elliptischen Kurve Curve25519 arbeiten. `pure25519` ist eine reine Python-Implementierung für diese Kurve, die für die Implementierung der Services verwendet wurde. Als Vorbereitung brauchst du für diese Aufgabe in Python lediglich:

```
pip install pycryptodome
pip install pure25519
```

Auf `nkp.mhgb.net:10002` läuft ein Testserver. Der Source Code für dieses Service ist frei verfügbar, du kannst ihn im Moodle-Kurs herunterladen (→ `testserver.py`). Der Code ist so vollständig, dass du dieses Service auch lokal bei dir betreiben kannst. Allerdings erkennst du, dass das vom Service verwendete Schlüsselpaar aus einem json-File gelesen wird, das dir nicht zur Verfügung steht. Ein Schlüsselpaar kannst du aber auch selbst erstellen und weitere benötigte Daten auch selbst „erfinden“. Dies lässt sich mit

```
from pure25519 import ed25519_oop as ed25519
ed_sign, ed_verif = ed25519.create_keypair()
```

einfach erledigen. Bring den Testserver bei dir lokal zum Laufen.

Du kannst nach dieser Teilaufgabe

- den Testserver bei dir lokal betreiben und
- mit dem `pure25519`-Modul Curve25519-Schlüsselpaare erstellen.

Teilaufgabe 2

Du findest – als Kommentar – im Source Code eine kurze Beschreibung des Authentifizierungsprotokolls, das hier verwendet wird, außerdem eine ausreichend genaue Spezifikation der Protokollnachrichten.

Erstelle dir aus dem Source Code eine Darstellung des Protokolls mit allen Protokollnachrichten.

Du hast nach dieser Teilaufgabe

- einen Überblick über das verwendete Protokoll und
- Kenntnis über das Nachrichtenformat.

Teilaufgabe 3

Weitere Kommentare verraten dir, dass es sich hier um eine Version zum Testen handelt. Dieses Service akzeptiert beliebige Client IDs und beliebige Public Keys und führt für diese das Tendermint-Authentifizierungsprotokoll durch.[1] Du hast eine Kurzbeschreibung des Protokolls und den Source Code des Testservers. Erstelle nun einen Client, der sich erfolgreich gegenüber dem Testserver `nkp.mhgb.net:10002` authentifiziert. Dies sollte nicht allzu schwierig sein, da sich die Nachrichten vom und zum Server offenbar nicht sehr unterscheiden. Wenn dein Client korrekt funktioniert, solltest du das Ergebnis „`handshake successful`“ erhalten.

Du kannst nach dieser Teilaufgabe

- Nachrichten mit dem Testserver austauschen (sowohl lokal als auch mit `nkp.mhgb.net:10002`),
- das Protokoll erfolgreich mit dem Testserver (sowohl lokal als auch mit `nkp.mhgb.net:10002`) abschließen können und
- den im Protokoll vereinbarten Schlüssel korrekt verwenden können.

Du weißt,

- wie die Authentifizierung durchgeführt wird,
- wie ein Session Key zwischen Client und Server erzeugt wird und
- welches Verfahren zur Verschlüsselung verwendet wird.

Teilaufgabe 4

Auf `nkp.mhgb.net:10001` läuft ein ganz ähnlicher Service.

Dieser – so entnimmst du den Kommentaren im vorliegenden Source Code – akzeptiert allerdings nur Verbindungen vom Testserver `nkp.mhgb.net:10002` (d.h. es muss dessen ID verwendet werden), ignoriert mitgelieferte Public Keys und authentifiziert ausschließlich gegen den Public Key des Testservers. Du solltest ansonsten aber genauso wie mit dem Testserver kommunizieren können. Der Source Code für diesen Server steht dir ebenfalls im Moodle-Kurs zur Verfügung (→ `server.py`). Somit kannst du schnell erkennen, wie die Unterschiede genau aussehen. Jedenfalls erkennst du sofort, dass du auch hier keinen direkten Zugriff auf Schlüssel hast.

Du stößt auf den Artikel „Prime, Order Please! Revisiting Small Subgroup and Invalid Curve Attacks on Protocols using Diffie-Hellman“[2] von Cremers und Jackson. In Abschnitt 7.3 geht es offenbar um das hier eingesetzte Protokoll und du liest dort:

> „[. . .] automatically discovers an attack using small subgroups, which allows an attacker to steal the identity of any party willing to complete a handshake with them. That is, if the attacker completes a handshake with A, the attacker can now connect to B using A's identity.“

[1] Verwendet wird hier ein für die Tendermint-Architektur beschriebenes Authentifizierungsprotokoll, welches hier so umgesetzt wird, dass ein Sicherheitsproblem entsteht. Die Services in dieser Challenge haben ansonsten nichts mit Blockchains zu tun.

[2] `https://eprint.iacr.org/2019/526.pdf`

Es sieht so aus, als könntest du dich doch erfolgreich authentifizieren. Du liest weiter:

> „The attack is relatively simple, if an attacker sends a small order point as their ephemeral public key, the resulting challenge is constant, regardless of the other party's contribution. When the handshake completes, the attacker obtains a signature on this constant challenge from the honest participant, which can then be substituted as their own signature in future sessions."

Mehr brauchst du eigentlich gar nicht. Überlege dir jetzt, wie du dich erfolgreich am Server als Testserver authentifizieren kannst.

Nach dieser Teilaufgabe

- kannst du erklären, was eine Small Subgroup Attack ist,
- hast du einen passenden „small order point" in Curve25519 identifiziert,
- kannst du abschätzen, mit welcher Wahrscheinlichkeit dein Angriff erfolgreich ist und
- kannst du damit abschätzen, nach vielen Versuchen du (im Mittel) erfolgreich sein solltest.

Teilaufgabe 5

In einem ersten Schritt kannst du beide Server – Testserver und Server – selbst betreiben, um deinen Angriff zu testen (fehlende Daten selbst erzeugen). So kannst du besser (insb. in den Logs) sehen, was auf den verschiedenen Systemen passiert.

Führe schließlich deinen Angriff auf die Server `nkp.mhgb.net:10001` und `nkp.mhgb.net:10002` durch. Du wirst feststellen, dass die einfachste Idee (das neutrale Element der Gruppe verwenden), nicht zum Ziel führt, weil der Server dies erkennt. Glücklicherweise gibt es in der Curve25519-Gruppe auch andere Elemente mit kleiner Ordnung, die du verwenden kannst.

Nach erfolgreicher Authentifizierung als Testserver bei `nkp.mhgb.net:10001` erhältst du eine Flag. Gib diese im Moodle-Quiz ein, um den zweiten Badge zu erhalten.

Nach dieser Teilaufgabe

- kannst du Vorschläge machen, wie der Angriff auf das Protokoll serverseitig verhindert werden kann.

—

Beachte auch bei dieser Aufgabe, dass es sich um ein kryptographisches Praktikum handelt. Die Aufgabe lässt sich lösen, ohne dabei etwas kaputt zu machen.