

Computationally Secure Two-Round Authenticated Message Exchange

Henning Schnoor

Klaas Ole Kürtz, Thomas Wilke

Institut für Informatik, Christian-Albrechts-Universität zu Kiel

20. März 2009

Authentifizierung

Grundlegendes Ziel

Kommunikationspartner wollen „überzeugt sein“, miteinander zu sprechen.

Authentifizierung

Grundlegendes Ziel

Kommunikationspartner wollen „überzeugt sein“, miteinander zu sprechen.

Erweiterungen

- ▶ Authentifizierter Nachrichtenaustausch
- ▶ Authentifizierter Schlüsselaustausch

Authentifizierung

Grundlegendes Ziel

Kommunikationspartner wollen „überzeugt sein“, miteinander zu sprechen.

Erweiterungen

- ▶ Authentifizierter Nachrichtenaustausch
- ▶ Authentifizierter Schlüsselaustausch

Üblicherweise:

- ▶ Challenge-Response-Verfahren
- ▶ interaktiv, mindestens 3 Runden

Unser Fall: Webservices

2-Runden-Protokoll

- ▶ C → S: Request
- ▶ S → C: Antwort

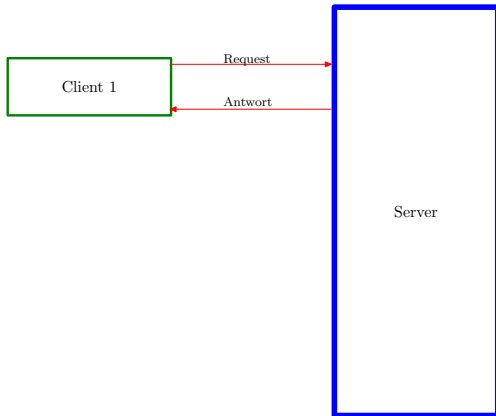
Client schickt in jeder Session
genau eine Nachricht

Unser Fall: Webservices

2-Runden-Protokoll

- ▶ C → S: **Request**
- ▶ S → C: **Antwort**

Client schickt in jeder Session
genau eine Nachricht



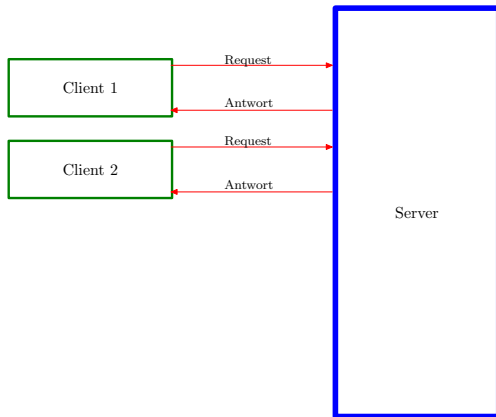
Unser Fall: Webservices

2-Runden-Protokoll

▶ C → S: **Request**

▶ S → C: **Antwort**

Client schickt in jeder Session
genau eine Nachricht



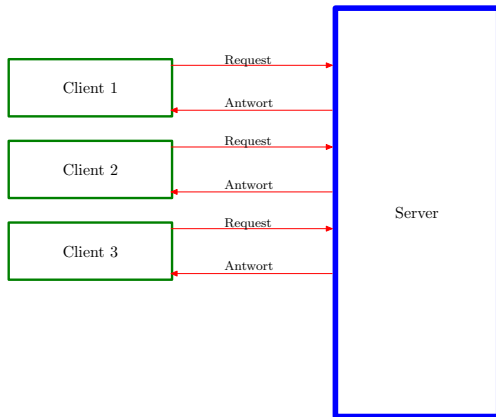
Unser Fall: Webservices

2-Runden-Protokoll

▶ C → S: **Request**

▶ S → C: **Antwort**

Client schickt in jeder Session
genau eine Nachricht

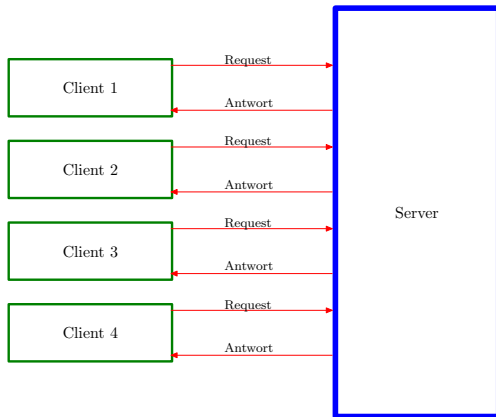


Unser Fall: Webservices

2-Runden-Protokoll

- ▶ C → S: **Request**
- ▶ S → C: **Antwort**

Client schickt in jeder Session
genau eine Nachricht



Nicht nur Authentifizierung, auch Nachrichtenaustausch.

In diesem Vortrag

Unser Beitrag

- ▶ Formales Modell für Sicherheitsanforderungen
- ▶ Protokoll
- ▶ Sicherheitsbeweis für Protokoll

Anwendungsszenario

Situation

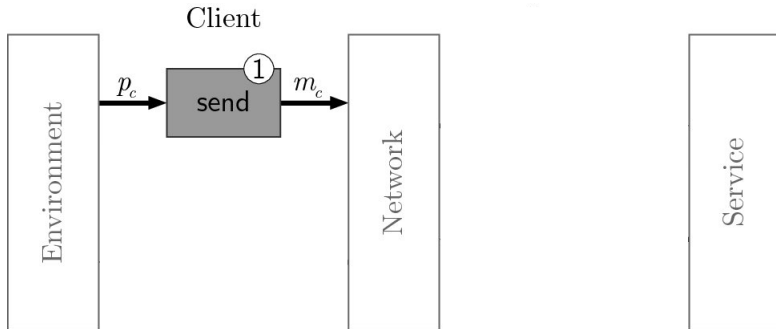
1. **Environment** schickt Anfragen an **Service**
2. **Service** schickt Antwort an **Environment**



Anwendungsszenario

Situation

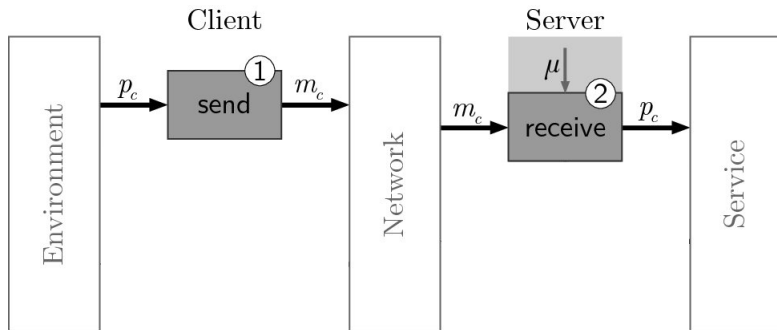
1. **Environment** schickt Anfragen an **Service**
2. **Service** schickt Antwort an **Environment**



Anwendungsszenario

Situation

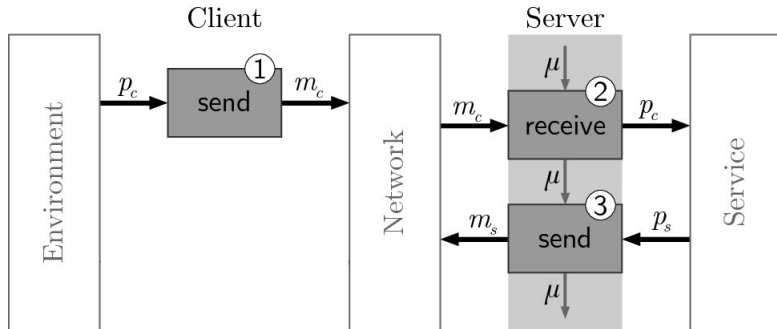
1. **Environment** schickt Anfragen an **Service**
2. **Service** schickt Antwort an **Environment**



Anwendungsszenario

Situation

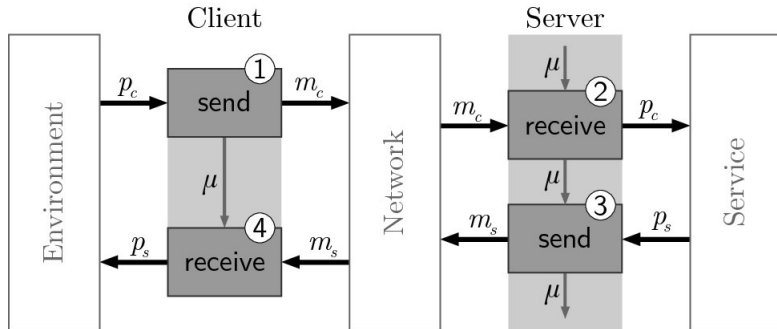
1. **Environment** schickt Anfragen an **Service**
2. **Service** schickt Antwort an **Environment**



Anwendungsszenario

Situation

1. **Environment** schickt Anfragen an **Service**
2. **Service** schickt Antwort an **Environment**



Forderung: Nutzdaten sollen „richtig“ ankommen

Sicherheitsüberlegungen

Mögliche Angriffe auf Kommunikation:

- ▶ Gefälschte Nachrichten
- ▶ Replay-Angriffe

Sicherheitsüberlegungen

Mögliche Angriffe auf Kommunikation:

- ▶ Gefälschte Nachrichten
- ▶ Replay-Angriffe

Gegenmaßnahmen

- ▶ Signaturschema
- ▶ Server speichert Nachrichten

Sicherheitsüberlegungen

Mögliche Angriffe auf Kommunikation:

- ▶ Gefälschte Nachrichten
- ▶ Replay-Angriffe

Gegenmaßnahmen

- ▶ Signaturschema
- ▶ Server speichert Nachrichten

Anforderungen aus Praxis

- ▶ Begrenzter Speicher des Servers?
- ▶ Was passiert bei Datenverlust?

Angreifermodell

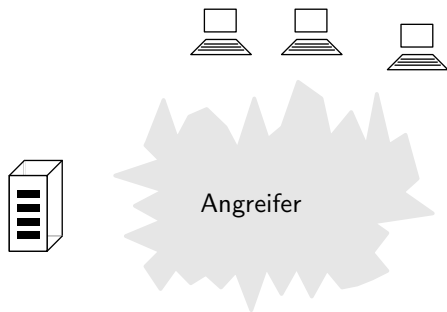
Der Angreifer



Angreifermodell

Der Angreifer

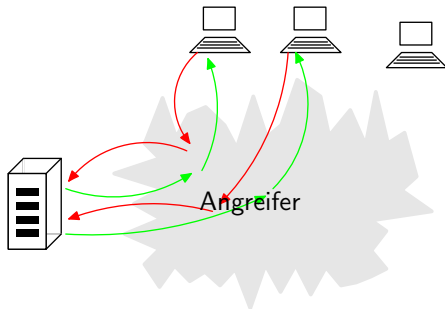
- ▶ Angreifer „ist das Netzwerk“



Angreifermodell

Der Angreifer

- ▶ Angreifer „ist das Netzwerk“
- ▶ Alle Kommunikation läuft über den Angreifer



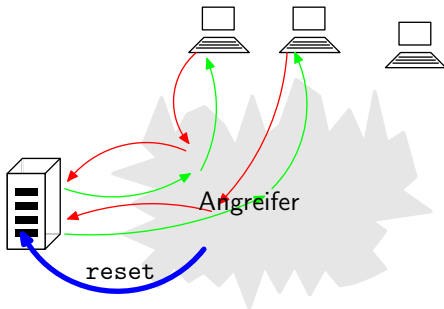
Angreifermodell

Der Angreifer

- ▶ Angreifer „ist das Netzwerk“
- ▶ Alle Kommunikation läuft über den Angreifer

Zusätzlich:

- ▶ Angreifer kann Server-Speicher löschen



Machbar?

Unmöglichkeitsergebnis

Jedes nicht-triviale Protokoll anfällig für Replay-Angriffe:

- ▶ Sei m Nachricht, die Server nach Reset akzeptiert

Machbar?

Unmöglichkeitsergebnis

Jedes nicht-triviale Protokoll anfällig für Replay-Angriffe:

- ▶ Sei m Nachricht, die Server nach Reset akzeptiert
- ▶ Angreifer:
 1. reset
 2. send: m
 3. reset
 4. send: m

Machbar?

Unmöglichkeitsergebnis

Jedes nicht-triviale Protokoll anfällig für Replay-Angriffe:

- ▶ Sei m Nachricht, die Server nach Reset akzeptiert
- ▶ Angreifer:
 1. reset
 2. send: m
 3. reset
 4. send: m

2 und 4 von Server nicht unterscheidbar!

Machbar?

Unmöglichkeitsergebnis

Jedes nicht-triviale Protokoll anfällig für Replay-Angriffe:

- ▶ Sei m Nachricht, die Server nach Reset akzeptiert
- ▶ Angreifer:
 1. reset
 2. send: m
 3. reset
 4. send: m

2 und 4 von Server nicht unterscheidbar!

Server muss Reset „bemerken“ und darauf reagieren können.

Uhren

Sicherheit durch Zeitstempel

Nehmen an: Alle Teilnehmer haben lokale Uhr

Uhren

Sicherheit durch Zeitstempel

Nehmen an: Alle Teilnehmer haben lokale Uhr

- ▶ Nicht synchronisiert

Uhren

Sicherheit durch Zeitstempel

Nehmen an: Alle Teilnehmer haben lokale Uhr

- ▶ Nicht synchronisiert
- ▶ Angreifer kontrolliert Uhren

Uhren

Sicherheit durch Zeitstempel

Nehmen an: Alle Teilnehmer haben lokale Uhr

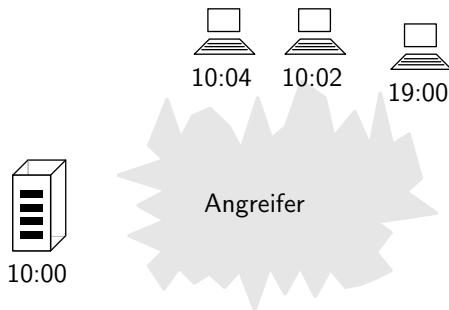
- ▶ Nicht synchronisiert
- ▶ Angreifer kontrolliert Uhren
- ▶ Minimalforderung: Uhren sind **monoton**

Uhren

Sicherheit durch Zeitstempel

Nehmen an: Alle Teilnehmer haben lokale Uhr

- ▶ Nicht synchronisiert
- ▶ Angreifer kontrolliert Uhren
- ▶ Minimalforderung: Uhren sind **monoton**

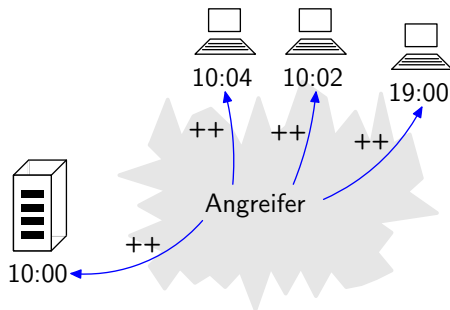


Uhren

Sicherheit durch Zeitstempel

Nehmen an: Alle Teilnehmer haben lokale Uhr

- ▶ Nicht synchronisiert
- ▶ Angreifer kontrolliert Uhren
- ▶ Minimalforderung: Uhren sind **monoton**



Modellierung und Sicherheitsanforderung

„Experiment“: Lassen Angreifer und Protokoll laufen.

- ▶ Feststellen, welche Nachrichten ausgetauscht wurden
- ▶ Abhängig davon: Angreifer erfolgreich?

Protokoll

„**trace**“: Protokoll aller relevanten Ereignisse

Experiment: Ablauf

1. Schlüsselerzeugung für Signaturschema

Experiment: Ablauf

1. Schlüsselerzeugung für Signaturschema
2. Initialisierung:
 - ▶ alle Uhren auf 0
 - ▶ interne Information ε

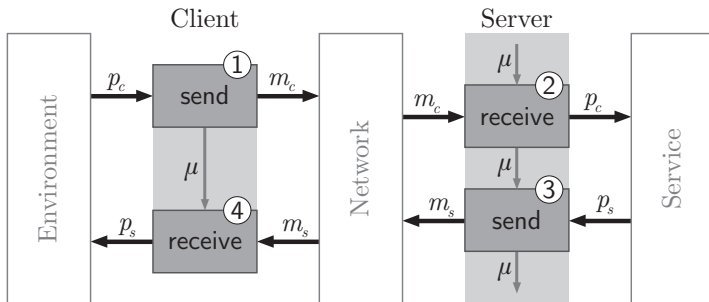
Experiment: Ablauf

1. Schlüsselerzeugung für Signaturschema
2. Initialisierung:
 - ▶ alle Uhren auf 0
 - ▶ interne Information ε
3. Angreifer: Jeder Schritt:
 - ▶ Uhren vorstellen
 - ▶ Korruption von Teilnehmer (Angreifer erhält privaten Schlüssel)
 - ▶ Start einer Client-Session
 - ▶ Zustellung von Nachricht an Client oder Server
 - ▶ Server-Reset

„Syntax“ der Algorithmen

4 Schritte:

1. **Client** send
2. **Server** receive
3. **Server** send
4. **Client** receive



„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Client: Sendeschritt

- Eingabe:
- ▶ s Server an den Nachricht geschickt werden soll
 - ▶ p_c payload-Anfrage
 - ▶ t_c lokale Uhrzeit

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Client: Sendeschritt

- Eingabe:
- ▶ s Server an den Nachricht geschickt werden soll
 - ▶ p_c payload-Anfrage
 - ▶ t_c lokale Uhrzeit
- Ausgabe:
- ▶ m_c Zu sendende Nachricht

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Client: Sendeschritt

- Eingabe:
- ▶ s Server an den Nachricht geschickt werden soll
 - ▶ p_c payload-Anfrage
 - ▶ t_c lokale Uhrzeit

Ausgabe: ▶ m_c Zu sendende Nachricht

trace-Eintrag: ▶ $(c, \text{send}, s, p_c, t_c, m_c)$

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Server: Empfangsschritt

- Eingabe:
- ▶ m_c Eingehende Nachricht
 - ▶ t_s lokale Uhrzeit

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Server: Empfangsschritt

- Eingabe:
- ▶ m_c Eingehende Nachricht
 - ▶ t_s lokale Uhrzeit

- Ausgabe:
- ▶ c (vermutete) Identität des Absenders
 - ▶ p_c extrahierte Payload
 - ▶ $\delta \in \{A, R\}$ Entscheidung Akzeptanz

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Server: Empfangsschritt

Eingabe: ▶ m_c Eingehende Nachricht
 ▶ t_s lokale Uhrzeit

Ausgabe: ▶ c (vermutete) Identität des Absenders
 ▶ p_c extrahierte Payload
 ▶ $\delta \in \{A, R\}$ Entscheidung Akzeptanz

trace-Eintrag: ▶ $(s, \text{receive}, m_c, t_s, c, p_c, \delta)$

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Server: Sendeschritt

- Eingabe:
- ▶ p_s payload-Antwort
 - ▶ t_s lokale Uhrzeit

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Server: Sendeschritt

Eingabe: ▶ p_s payload-Antwort
 ▶ t_s lokale Uhrzeit

Ausgabe: ▶ m_s Zu sendende Nachricht
 ▶ c Empfänger

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Server: Sendeschritt

Eingabe: ▶ p_s payload-Antwort
 ▶ t_s lokale Uhrzeit

Ausgabe: ▶ m_s Zu sendende Nachricht
 ▶ c Empfänger

trace-Eintrag: ▶ $(s, \text{send}, p_s.t_s, m_s, c)$

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Client: Empfangsschritt

- Eingabe:
- ▶ t_c lokale Uhrzeit
 - ▶ m_s Eingehende Nachricht

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Client: Empfangsschritt

- Eingabe:
- ▶ t_c lokale Uhrzeit
 - ▶ m_s Eingehende Nachricht

- Ausgabe:
- ▶ p_s extrahierte Payload
 - ▶ $\delta \in \{A, R\}$ Entscheidung Akzeptanz

„Syntax“ der Algorithmen

Ein/Ausgabe und trace-Protokoll

Client: Empfangsschritt

- Eingabe:
- ▶ t_c lokale Uhrzeit
 - ▶ m_s Eingehende Nachricht

- Ausgabe:
- ▶ p_s extrahierte Payload
 - ▶ $\delta \in \{A, R\}$ Entscheidung Akzeptanz

trace-Eintrag: ▶ $(c, \text{receive}, t_c, m_s, p_s, \delta)$

Korrekttheitsdefinition

Trace enthält

- ▶ Welche Nutzdaten *sollten* geschickt werden?
- ▶ Welche Nutzdaten sind *angekommen*?
- ▶ Wer hat wann welche Nachricht von wem akzeptiert?
- ▶ ...

Korrekttheitsdefinition

Trace enthält

- ▶ Welche Nutzdaten *sollten* geschickt werden?
- ▶ Welche Nutzdaten sind *angekommen*?
- ▶ Wer hat wann welche Nachricht von wem akzeptiert?
- ▶ ...

Korrekttheitsdefinition

Wenn Angreifer „gutartig“, sollen alle Daten „wie erwartet“ zugestellt und interpretiert werden.

Korrekttheitsdefinition

Trace enthält

- ▶ Welche Nutzdaten *sollten* geschickt werden?
- ▶ Welche Nutzdaten sind *angekommen*?
- ▶ Wer hat wann welche Nachricht von wem akzeptiert?
- ▶ ...

Korrekttheitsdefinition

Wenn Angreifer „gutartig“, sollen alle Daten „wie erwartet“ zugestellt und interpretiert werden.

Formal: Eigenschaft der trace

Experiment: Reset

Reset

Angreifer kann Server-Speicher löschen:

`reset(s)` : Setze Speicher von s auf ϵ

Experiment: Reset

Reset

Angreifer kann Server-Speicher löschen:

`reset(s)` : Setze Speicher von s auf ϵ

Erhalten bleiben:

- ▶ Zugriff auf PKI
- ▶ Uhr wird nicht auf Null gestellt

Server wird von Reset informiert, d.h., kann darauf reagieren.

Angriffsdefinition

Sicherheit

Angreifer ist **erfolgreich**, falls im Experimentablauf

Server Akzeptiert Payload p_c von Client n -mal, Client hat p_c weniger als n -mal geschickt

Angriffsdefinition

Sicherheit

Angreifer ist **erfolgreich**, falls im Experimentablauf

Server Akzeptiert Payload p_c von Client n -mal, Client hat p_c weniger als n -mal geschickt

Client Akzeptiert Nachricht von Server, die *nicht* Antwort auf seine Anfrage ist

Formal: Eigenschaft der trace

Sicherheitsdefinition

Definition

Protokoll ist **sicher**, falls für jeden Polynomialzeit-Angreifer Erfolgswahrscheinlichkeit vernachlässigbar.

Protokollelemente

Verteidigung gegen ...

Nachrichtenfälschung Signaturschema

Replay-Angriffe Timestamps und „erlaubte“ Nachrichtenintervalle

Protokollelemente

Verteidigung gegen ...

Nachrichtenfälschung Signaturschema

Replay-Angriffe Timestamps und „erlaubte“ Nachrichtenintervalle

Angriff gegen Server:

1. reset
2. Sende m
3. reset
4. Sende m

Protokollelemente

Verteidigung gegen ...

Nachrichtenfälschung Signaturschema

Replay-Angriffe Timestamps und „erlaubte“ Nachrichtenintervalle

Angriff gegen Server:

1. reset
2. Sende m
3. reset
4. Sende m

Gegenmaßnahme

Mit Timestamps:

- ▶ Nachrichten haben Zeitstempel
- ▶ Alle Zeitstempel vor einem reset werden danach ungültig

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From: } c, \text{ To: } s, \text{ MsgID: } r, \text{ Timestamp: } t, \text{ Body: } x \}_{sk_c}$

$S \rightarrow S: \{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } y \}_{sk_s}$

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From}:c, \text{To}:s, \text{MsgID}:r, \text{Timestamp}:t, \text{Body}:x \}_{sk_c}$

$S \rightarrow S: \{ \text{From}:s, \text{To}:c, \text{RelTo}:r, \text{Body}:y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From}:c, \text{To}:s, \text{MsgID}:r, \text{Timestamp}:t, \text{Body}:x \}_{sk_c}$

$S \rightarrow S: \{ \text{From}:s, \text{To}:c, \text{RelTo}:r, \text{Body}:y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From}:c, \text{To}:s, \text{MsgID}:r, \text{Timestamp}:t, \text{Body}:x \}_{sk_c}$

$S \rightarrow S: \{ \text{From}:s, \text{To}:c, \text{RelTo}:r, \text{Body}:y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur
2. Falls MsgID in Liste L , lehne ab.
3. Akzeptiere Nachrichten mit Zeitstempel
 - ▶ $t_{\min} \leq \text{Timestamp} < t_{\text{aktuell}} + t_{\text{toleranz}}$

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From}:c, \text{To}:s, \text{MsgID}:r, \text{Timestamp}:t, \text{Body}:x \}_{sk_c}$

$S \rightarrow S: \{ \text{From}:s, \text{To}:c, \text{RelTo}:r, \text{Body}:y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur
2. Falls MsgID in Liste L , lehne ab.
3. Akzeptiere Nachrichten mit Zeitstempel
 - ▶ $t_{\min} \leq \text{Timestamp} < t_{\text{aktuell}} + t_{\text{toleranz}}$
4. Schreibe (t, r) Liste L

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From: } c, \text{ To: } s, \text{ MsgID: } r, \text{ Timestamp: } t, \text{ Body: } x \}_{sk_c}$
 $S \rightarrow S: \{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur
2. Falls MsgID in Liste L , lehne ab.
3. Akzeptiere Nachrichten mit Zeitstempel
 - ▶ $t_{\min} \leq \text{Timestamp} < t_{\text{aktuell}} + t_{\text{toleranz}}$
4. Schreibe (t, r) Liste L
5. Antwort: $\{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } p \}$

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From: } c, \text{ To: } s, \text{ MsgID: } r, \text{ Timestamp: } t, \text{ Body: } x \}_{sk_c}$
 $S \rightarrow S: \{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur
 2. Falls MsgID in Liste L , lehne ab.
 3. Akzeptiere Nachrichten mit Zeitstempel
 - ▶ $t_{\min} \leq \text{Timestamp} < t_{\text{aktuell}} + t_{\text{toleranz}}$
 4. Schreibe (t, r) Liste L
 5. Antwort: $\{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } p \}$
- ▶ Um Speicherplatz freizugeben:
1. Erhöhe t_{\min}
 2. Lösche Einträge mit Zeitstempel $< t_{\min}$ aus L

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From: } c, \text{ To: } s, \text{ MsgID: } r, \text{ Timestamp: } t, \text{ Body: } x \}_{sk_c}$
 $S \rightarrow S: \{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur
2. Falls MsgID in Liste L , lehne ab.
3. Akzeptiere Nachrichten mit Zeitstempel
 - ▶ $t_{\min} \leq \text{Timestamp} < t_{\text{aktuell}} + t_{\text{toleranz}}$
4. Schreibe (t, r) Liste L
5. Antwort: $\{ \text{From: } s, \text{ To: } c, \text{ RelTo: } r, \text{ Body: } p \}$
 - ▶ Um Speicherplatz freizugeben:
 1. Erhöhe t_{\min}
 2. Lösche Einträge mit Zeitstempel $< t_{\min}$ aus L

Server-Reset : Setze t_{\min} auf $t_{\text{aktuell}} + t_{\text{toleranz}}$.

Das Protokoll

Nachrichtenaustausch

$C \rightarrow S: \{ \text{From}:c, \text{To}:s, \text{MsgID}:r, \text{Timestamp}:t, \text{Body}:x \}_{sk_c}$
 $S \rightarrow S: \{ \text{From}:s, \text{To}:c, \text{RelTo}:r, \text{Body}:y \}_{sk_s}$

Server: ▶ Pflegt Liste aus Paaren (Timestamp, MsgID)

1. Prüfe From, Signatur
2. Falls MsgID in Liste L , lehne ab.
3. Akzeptiere Nachrichten mit Zeitstempel
 - ▶ $t_{\min} \leq \text{Timestamp} < t_{\text{aktuell}} + t_{\text{toleranz}}$
4. Schreibe (t, r) Liste L
5. Antwort: $\{ \text{From}: s, \text{To}: c, \text{RelTo}: r, \text{Body}: p \}$
 - ▶ Um Speicherplatz freizugeben:
 1. Erhöhe t_{\min}
 2. Lösche Einträge mit Zeitstempel $< t_{\min}$ aus L

Server-Reset : Setze t_{\min} auf $t_{\text{aktuell}} + t_{\text{toleranz}}$.

Client: ▶ Prüfe Signatur, From:, To:, RelTo:

„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

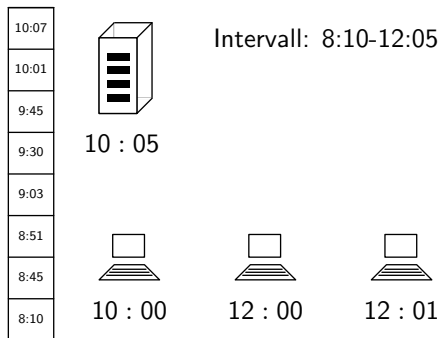
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



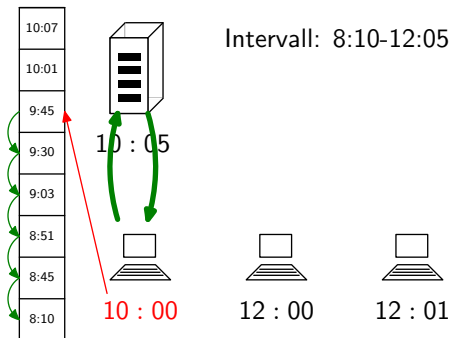
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



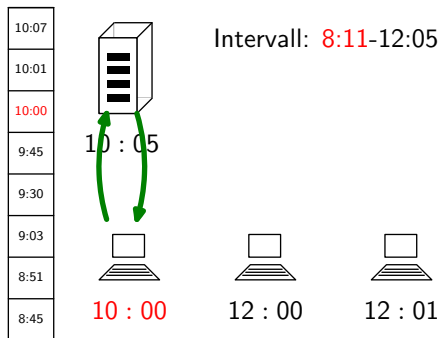
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



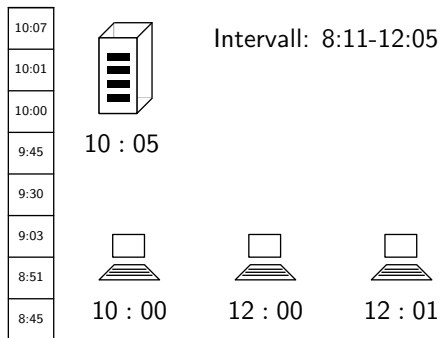
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



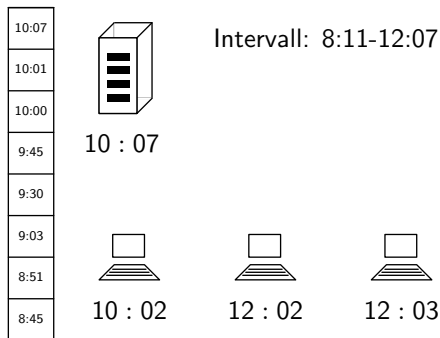
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



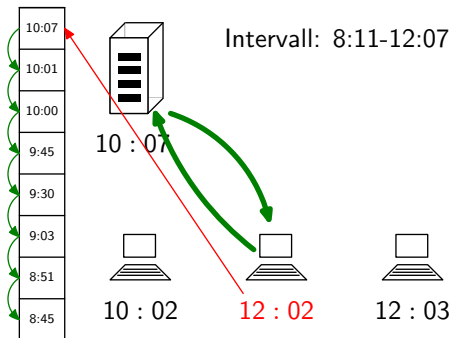
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



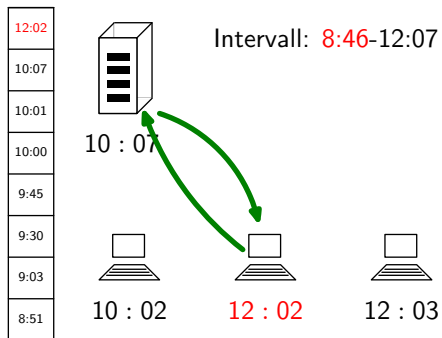
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



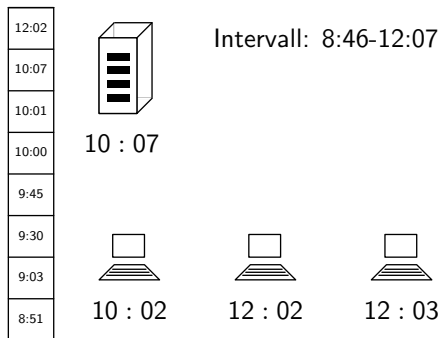
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



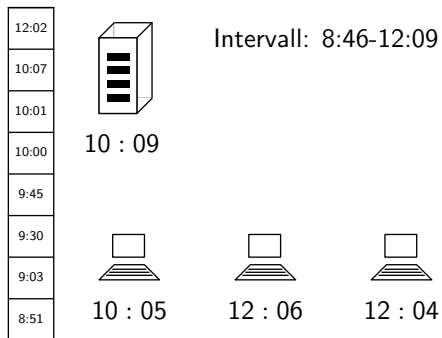
„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

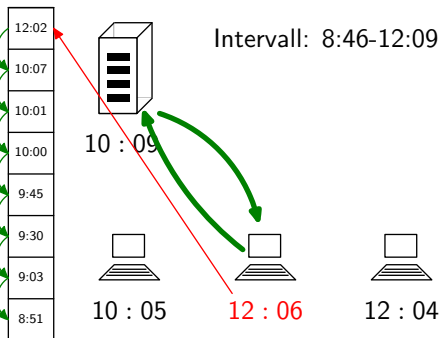
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

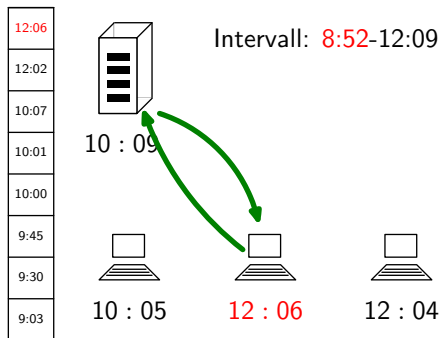
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

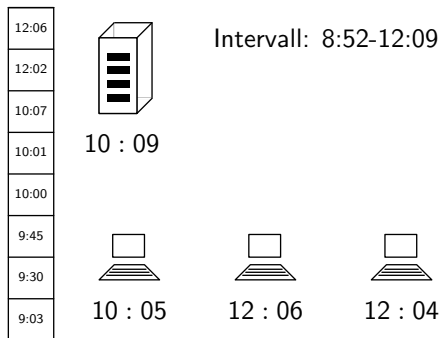
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

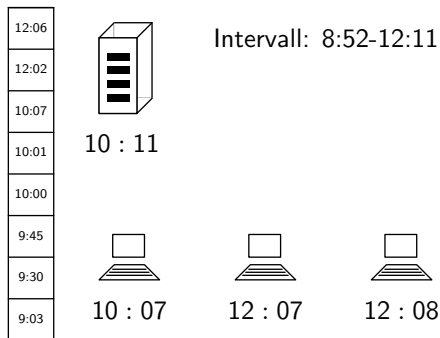
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

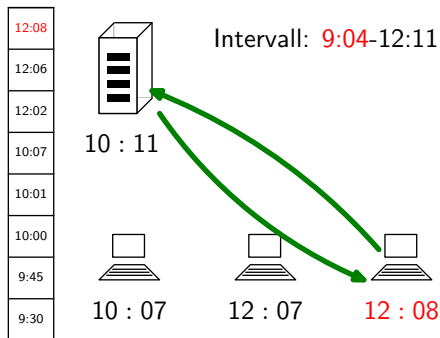
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

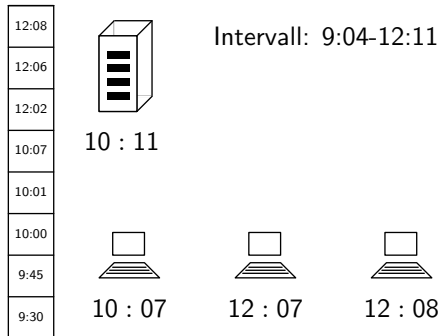
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

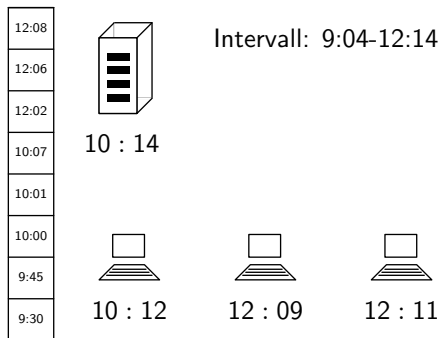
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

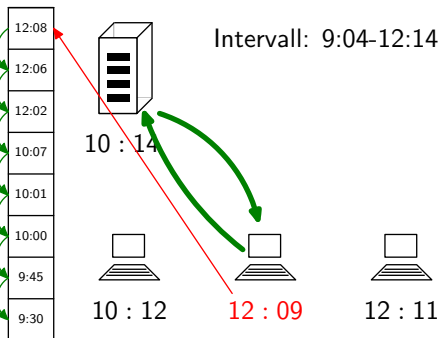
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

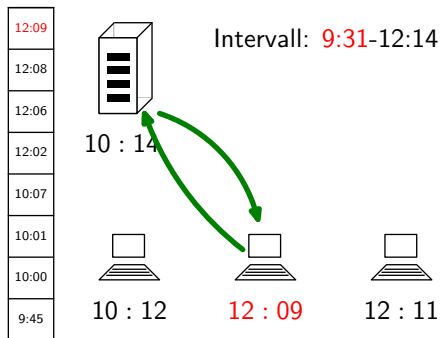
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

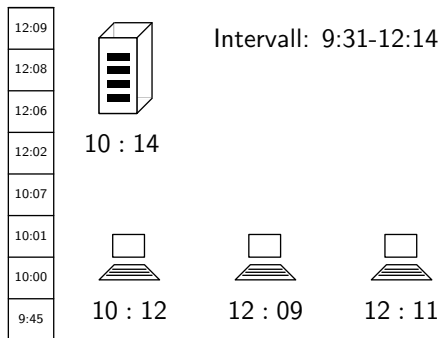
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

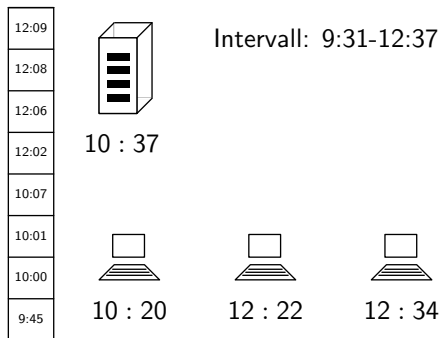
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

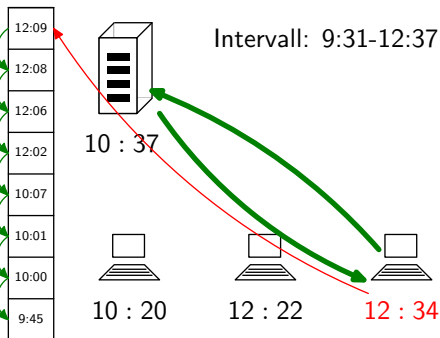
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

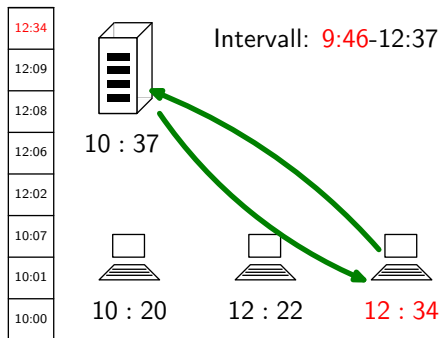
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

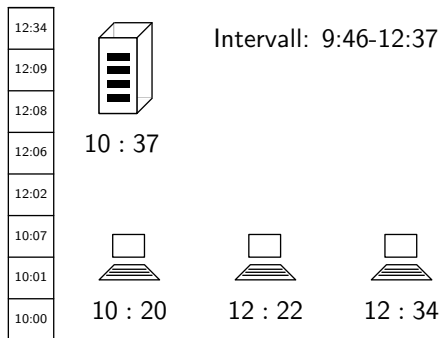
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

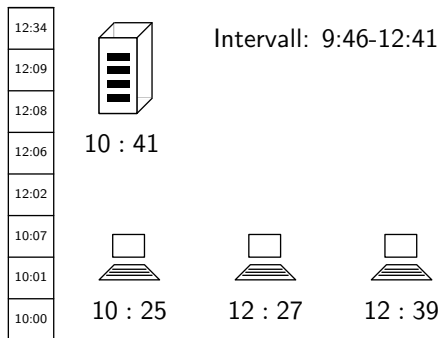
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

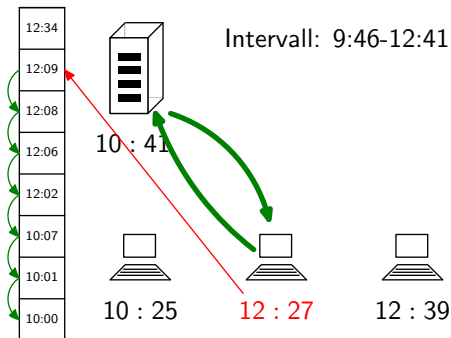
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

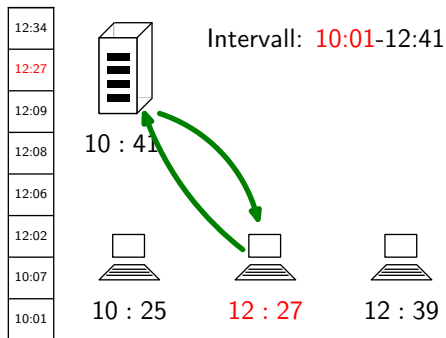
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

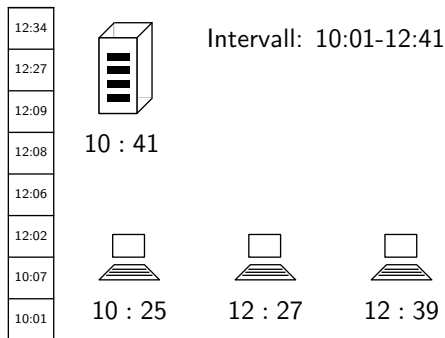
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

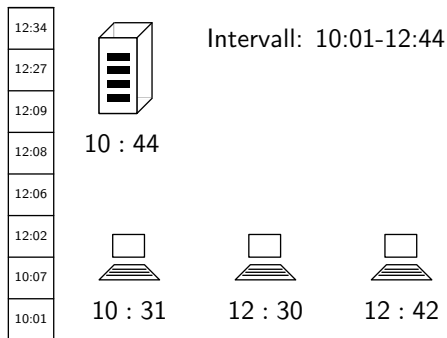
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

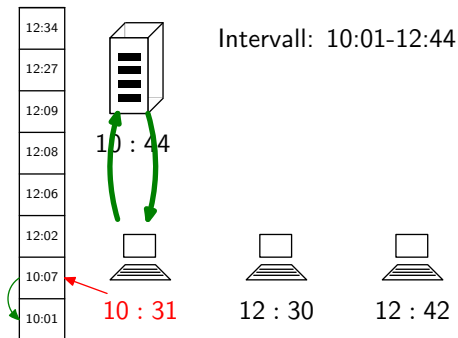
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

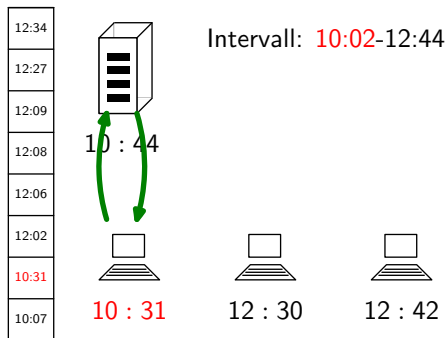
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

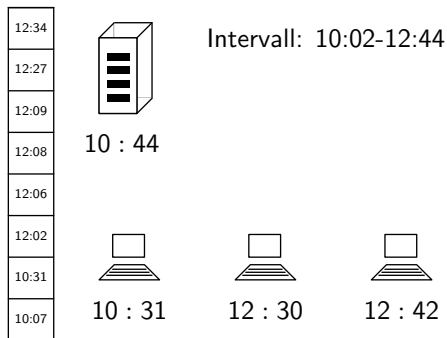
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

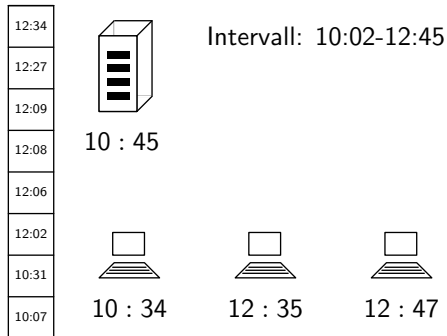
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

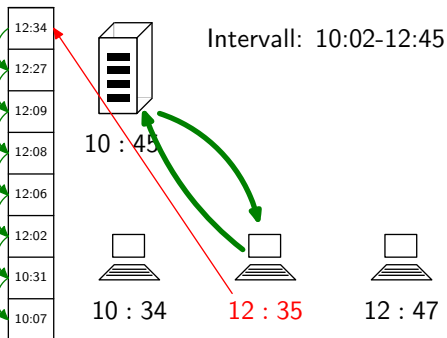
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

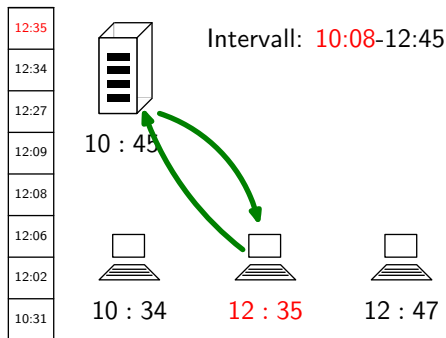
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

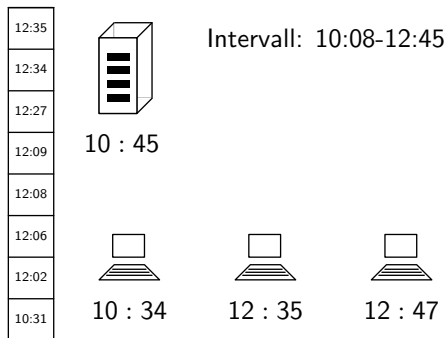
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

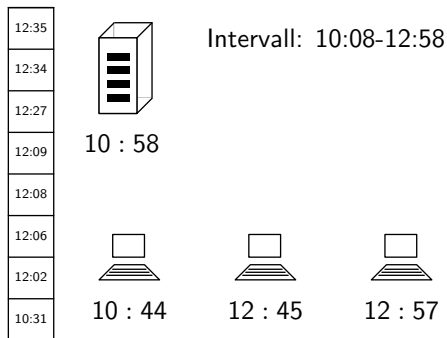
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

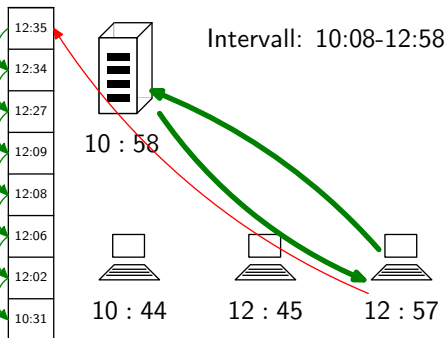
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

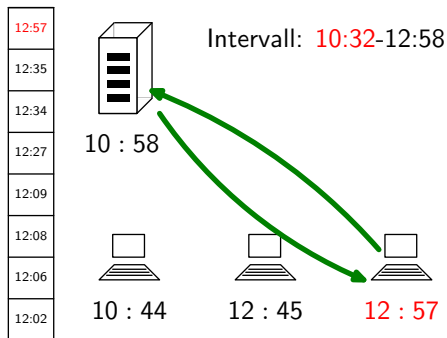
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

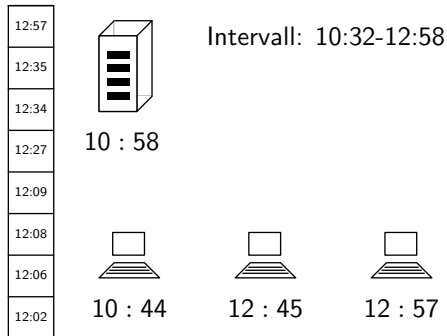
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

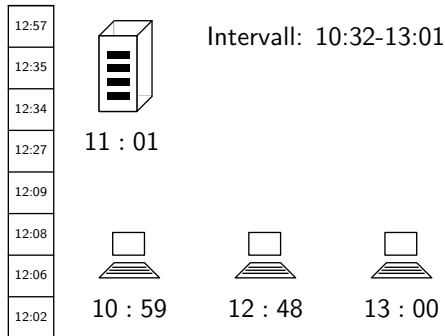
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

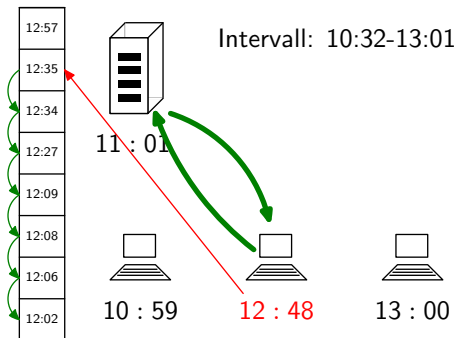
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

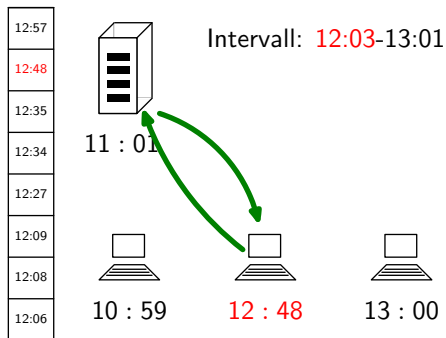
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

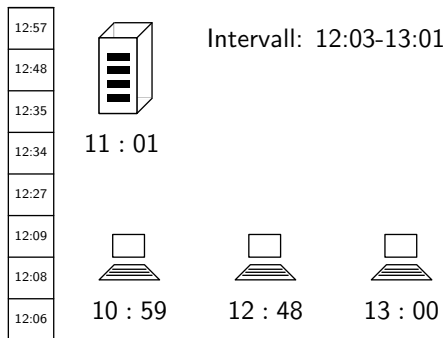
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

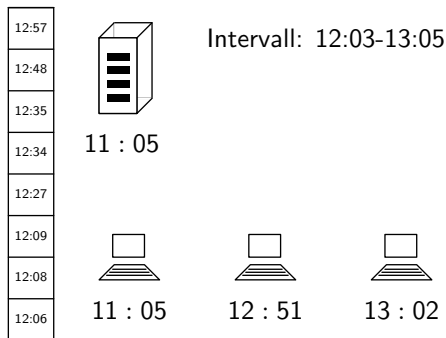
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

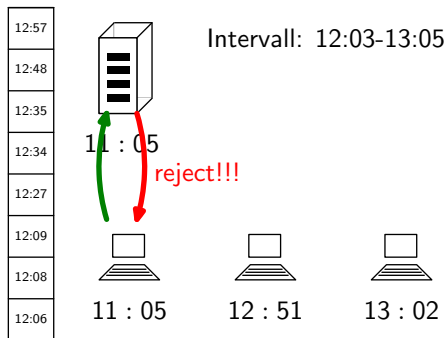
- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich
- ▶ Legitime Anfragen abgewiesen

Beispiel

Server mit 2 Stunden Toleranz und kleiner Liste



„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich
- ▶ Legitime Anfragen abgewiesen

Gegenmaßnahme

- ▶ Bei hoher Toleranz:
Entsprechend lange Liste

„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich
- ▶ Legitime Anfragen abgewiesen

Gegenmaßnahme

- ▶ Bei hoher Toleranz:
Entsprechend lange Liste
- ▶ Optimal: „OK-Intervall“ hat
aktuelle Zeit als Mittelpunkt

„Denial of Service“ Angriff

Server „kapern“

- ▶ Uhren „ans Limit“
- ▶ viele Anfragen

Effekt

- ▶ Akzeptiertes Intervall verschiebt sich
- ▶ Legitime Anfragen abgewiesen

Gegenmaßnahme

- ▶ Bei hoher Toleranz:
Entsprechend lange Liste
- ▶ Optimal: „OK-Intervall“ hat
aktuelle Zeit als Mittelpunkt
- ▶ Minimal: Liste so groß, dass
Intervall nie komplett in Zukunft

Nachrichtentexte?

Bisher:

Protokoll unabhängig von message bodies betrachtet.

Nachrichtentexte?

Bisher:

Protokoll unabhängig von message bodies betrachtet.

Problem:

Informationen in Bodies können sicherheitsrelevant sein!

- ▶ Separate Signaturen für Teile
- ▶ enthaltene Schlüssel

Nachrichtentexte?

Bisher:

Protokoll unabhängig von message bodies betrachtet.

Problem:

Informationen in Bodies können sicherheitsrelevant sein!

- ▶ Separate Signaturen für Teile
- ▶ enthaltene Schlüssel

Forderung:

Inhalte dürfen Sicherheit nicht gefährden:

- ▶ Bodies ohne Wissen des privaten Schlüssels erstellbar
- ▶ Zugriff auf Signatur-Orakel erlaubt.

Ausnahme: Protokollnachrichten

Unser Protokoll ist sicher

Satz

Wenn Signaturschema resistent gegen existential forgery, dann ist Protokoll sicher.

Zusammenfassung

Motivation

- ▶ Authentifizierung mit zwei ausgetauschten Nachrichten
- ▶ Begrenzter Speicher
- ▶ Angreifer: `reset`

Zusammenfassung

Motivation

- ▶ Authentifizierung mit zwei ausgetauschten Nachrichten
- ▶ Begrenzter Speicher
- ▶ Angreifer: `reset`

Beiträge

- ▶ Modell und Sicherheitsbegriff für Szenario
- ▶ Sicheres Protokoll

Zusammenfassung

Motivation

- ▶ Authentifizierung mit zwei ausgetauschten Nachrichten
- ▶ Begrenzter Speicher
- ▶ Angreifer: `reset`

Beiträge

- ▶ Modell und Sicherheitsbegriff für Szenario
- ▶ Sicheres Protokoll

Danke!