Anonymes Kommunizieren mit Mixminion

Seminar „Peer-to-Peer Netzwerke“

Claudius Korzen

Institut für Informatik
Albert-Ludwigs-Universität, Freiburg

SS 2009

28. Juli 2009

Überblick

Überblick

Motivation

- Üblicherweise: Schutz des **Inhaltes** von E-Mails vor Unbefugten
  - Methoden: Symmetrische/Asymmetrische Verschlüsselungsverfahren
- Immer öfter: Geheimhaltung von **Absender** und/oder **Empfänger** einer Nachricht
  - U.a. motiviert durch Einführung der Vorratsdatenspeicherung
- Hierzu: Verwendung von *Remailern*
  - Entpersonalisierung der Nachrichten
  - Nachrichten können keinen Identitäten mehr zugeordnet werden
  - Basieren auf Mix-Netzwerken nach David Chaum ([3])

Überblick

Definition eines Sitzungsmodells

- *Alice* möchte *Bob* eine Nachricht *N* schicken
- Dabei sollen die Schutzziele
  - Vertraulichkeit
  - Integrität
  - Anonymität
  - Authentizität

  eingehalten werden.
- Angreifer *Eve* möchte Schutzziele gefährden
- Eve kann
  - gesamtes Netzwerk beobachten
  - Netzwerk-Traffic analysieren
  - Pakete abfangen, zurückhalten und manipulieren

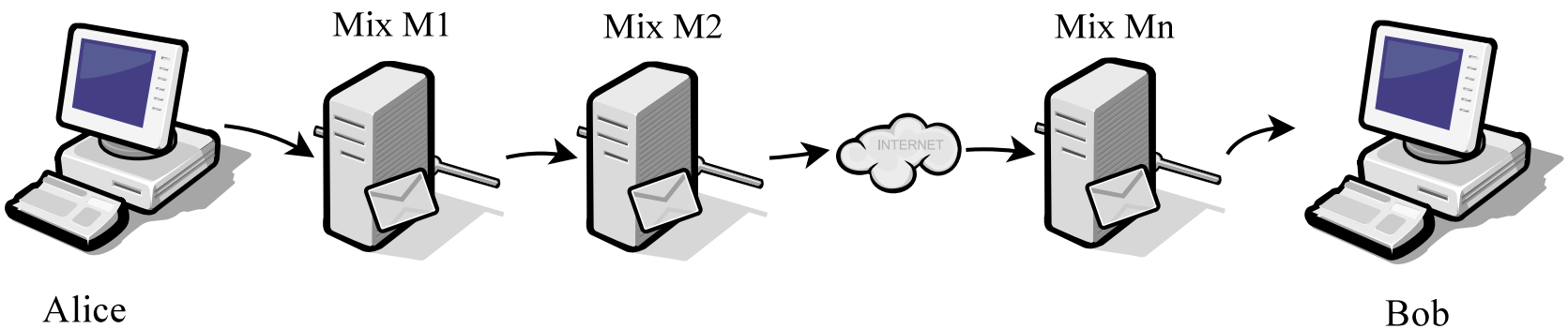
Mix-Netzwerke (1)

- nach *David L. Chaum*
- Basis für die meisten Remailer-Protokolle
- Netzwerk von sog. *Mixes*
  - Server, die von versch. Personen oder Institutionen verwaltet werden

- Mix M_i
- öffentlicher Schlüssel E_{M_i}
- privater Schlüssel D_{M_i}
- leiten Nachrichten weiter

Mix-Netzwerke (2)

- Alice wählt Pfad $M = (M_1, M_2, \ldots, M_n)$ von Mixes aus Netzwerk aus
- Nachricht N wird über gewählten Pfad M verschickt

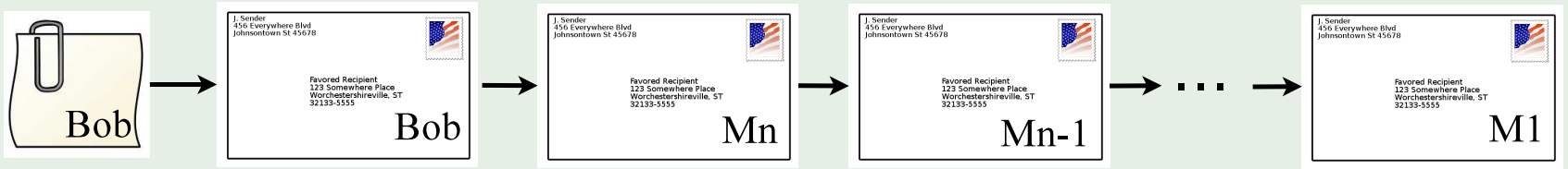


Mix-Netzwerke (3)

- Alice verschlüsselt Nachricht N mit öffentlichen Schlüsseln der Mixes aus M
- zusätzlich: Routing-Informationen R_i $(i \in [1, n+1])$
- Verschlüsselung nach "Zwiebel-Technik", beginnend mit E_{M_n}:

$$N' = (R_1, E_{M_1}(R_2, E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$$

Beispiel



Mix-Netzwerke (4)

Ablauf des Versendens einer Nachricht

- Alice sendet N' an M_1
- M_1 kann mit D_{M_1} erste Verschlüsselungsschicht lösen
  - erhält R_2 und $N'' = E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$
- R_2 enthält Adresse von M_2
- M_1 sendet N'' an M_2
- ...
- M_n leitet letztendlich $E_{Bob}(N)$ an Bob weiter

Mix-Netzwerke (4)

Ablauf des Versendens einer Nachricht

- Alice sendet N' an M_1
- M_1 kann mit D_{M_1} erste Verschlüsselungsschicht lösen
  - erhält R_2 und $N'' = E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$
- R_2 enthält Adresse von M_2
- M_1 sendet N'' an M_2
- ...
- M_n leitet letztendlich $E_{Bob}(N)$ an Bob weiter

Mix-Netzwerke (4)

Ablauf des Versendens einer Nachricht

- Alice sendet N' an M_1
- M_1 kann mit D_{M_1} erste Verschlüsselungsschicht lösen
  - erhält R_2 und $N'' = E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$
- R_2 enthält Adresse von M_2
- M_1 sendet N'' an M_2
- ...
- M_n leitet letztendlich $E_{Bob}(N)$ an Bob weiter

Mix-Netzwerke (4)

Ablauf des Versendens einer Nachricht

- Alice sendet N' an M_1
- M_1 kann mit D_{M_1} erste Verschlüsselungsschicht lösen
  - erhält R_2 und $N'' = E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$
- R_2 enthält Adresse von M_2
- M_1 sendet N'' an M_2
- ...
- M_n leitet letztendlich $E_{Bob}(N)$ an Bob weiter

Mix-Netzwerke (4)

Ablauf des Versendens einer Nachricht

- Alice sendet N' an M_1
- M_1 kann mit D_{M_1} erste Verschlüsselungsschicht lösen
  - erhält R_2 und $N'' = E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$
- R_2 enthält Adresse von M_2
- M_1 sendet N'' an M_2
- ...
- M_n leitet letztendlich $E_{Bob}(N)$ an Bob weiter

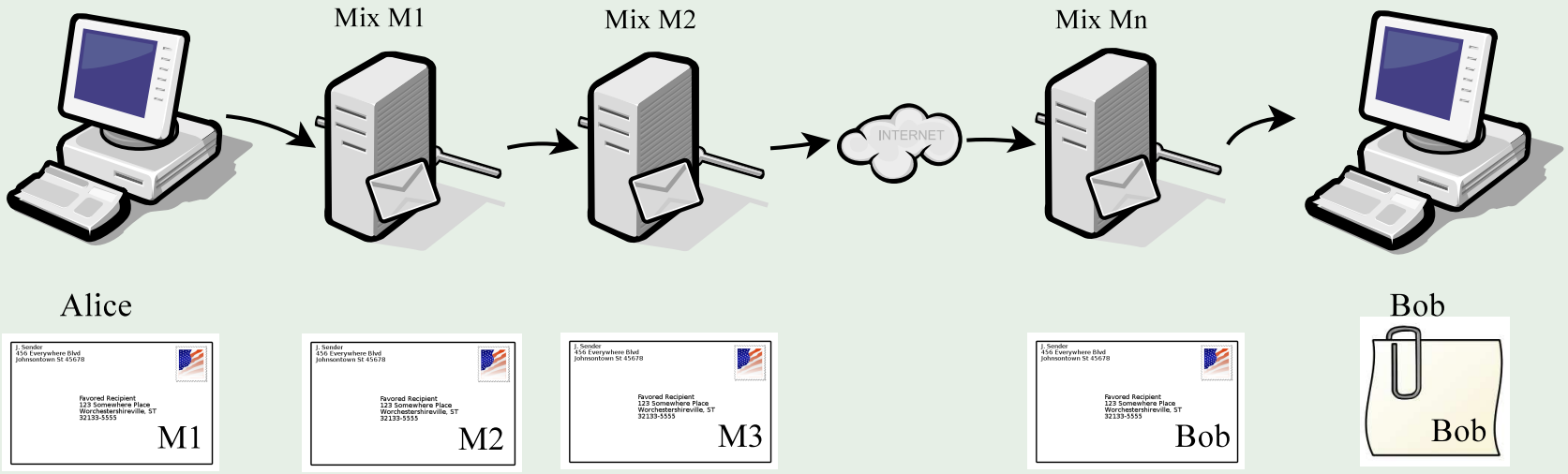
Mix-Netzwerke (4)

Ablauf des Versendens einer Nachricht

- Alice sendet N' an M_1
- M_1 kann mit D_{M_1} erste Verschlüsselungsschicht lösen
  - erhält R_2 und $N'' = E_{M_2}(\ldots(R_n, E_{M_n}(R_{n+1}, E_{Bob}(N)))\ldots)$
- R_2 enthält Adresse von M_2
- M_1 sendet N'' an M_2
- ...
- M_n leitet letztendlich $E_{Bob}(N)$ an Bob weiter

Mix-Netzwerke (5)

Beispiel



- Jeder Mix M_i kennt nur seinen unmittelbaren Vorgänger M_{i-1} und Nachfolger M_{i+1}
- Bob kennt nur M_n als Absender der Nachricht
- keine Kontrolle durch zentrale Instanz

Überblick

Zeitstrahl


1993
∼1994
1995
2007
Nymserver
Typ-0-Remailer
Cypherpunk
Typ-1-Remailer
Mixmaster
Typ-2-Remailer
Mixminion
Typ-3-Remailer

Cypherpunk

- Wortspiel aus “Cipher”, “Cyber” und “Punk”
- *Cypherpunks*: Gruppe, die sich für Kryptographie und Schutz der Privatsphäre einsetzen

- Sehr ähnlich zu Mix-Netzwerken
  - Kaum Änderungen an Funktionsweise von Mixes
  - Deshalb: Relativ anfällig gegenüber bestimmten Angriffen

Angriffe gegen Cypherpunk

- Traffic-Analyse möglich, da:
  - keine Aufteilung der Nachricht in einheitliche Paketgrößen
  - Nachrichten durch Mixes unmittelbar weiter versendet werden
- Replay-Angriffe möglich, da:
  - Eve beliebige Nachrichten abfangen und wieder einspielen kann
  - Nachrichten-Duplikate von Mixen nicht erkannt werden

Mixmaster

- baut auf Cypherpunk auf, beseitigt aber dessen Schwachstellen
- Mixes mit Puffern
  - Bei Ankunft einer Nachricht bei Mix, wird diese zunächst in Puffer gelegt
- Nachrichten werden erst verschickt, wenn Puffer gefüllt ist
- Weiterleitung in zufälliger Reihenfolge
- Nachrichten werden in gleich große Pakete (20 kB) aufgeteilt
- Mixes überprüfen Integrität der Daten anhand von Signaturen

Angriffe gegen Mixmaster

- Allgemein sehr sicher
- Replay-Angriffe nicht mehr möglich
  - Integritäts-Überprüfung
- Traffic-Analyse erschwert
  - einheitliche Paketgrößen
  - Versendezeitpunkte von Nachrichten hängen vom Zufall ab
- Theoretisch *Flooding-Attacken* möglich: Eve
  - fängt Nachricht von Alice ab
  - überflutet Mixes mit eigenen Nachrichten, bis Puffer gefüllt sind
  - gibt danach Alice's Nachricht wieder frei
- Dadurch Verfolgen der Nachricht möglich (da Eve Empfänger von ihren Nachrichten kennt)

Angriffe gegen Mixmaster

- Allgemein sehr sicher
- Replay-Angriffe nicht mehr möglich
  - Integritäts-Überprüfung
- Traffic-Analyse erschwert
  - einheitliche Paketgrößen
  - Versendezeitpunkte von Nachrichten hängen vom Zufall ab
- Theoretisch *Flooding-Attacken* möglich: Eve
  - fängt Nachricht von Alice ab
  - überflutet Mixes mit eigenen Nachrichten, bis Puffer gefüllt sind
  - gibt danach Alice's Nachricht wieder frei
- Dadurch Verfolgen der Nachricht möglich (da Eve Empfänger von ihren Nachrichten kennt)

Überblick

Mixminion

- entwickelt von George Danezis, Nick Mathewson und Roger Dingledine
- Ziel: Entwicklung eines Remailers, der
  - den letzten Forschungsergebnissen entspricht
  - einen hohen Grad an Anonymität bietet
  - einfach zu bedienen ist
- Dabei: Implementierung von z.T. neuartigen Funktionen und Methoden:
  - Antworten auf anonyme Nachrichten
  - Verschlüsselte Verbindungen zwischen Mixes:
    - TLS (statt SMTP)
  - Einführung eines Verzeichnisservers

Mixminion

Antworten auf anonyme Nachrichten

- Bob möchte auf die Nachricht von Alice antworten
- Er kennt aber weder E-Mailadresse noch Identität von Alice
- Lösung: *single-use reply blocks* (SURBs)
  - wird von Absender der Nachricht erstellt, auf die geantwortet werden soll (hier: Alice)
  - nur einmal verwendbar
  - enthält E-Mail-Adresse von Alice und Pfad durch Mixe (beides verschlüsselt)
- Bob antwortet Alice, indem er seine Nachricht an SURB anhängt

Mixminion

Drei verschiedene Nachrichtentypen (1)

- Durch Möglichkeit, Antworten zu verschicken, 2 neue Nachrichtentypen:
  - **normale Nachrichten**: Alice bleibt anonym
  - **direkte Antworten**: Bob bleibt anonym
  - **anonymisierte Antworten**: Alice *und* Bob bleiben anonym
- Nachrichtentypen müssen ununterscheidbar bleiben
- Nachrichten bestehen aus *Header* und *Payload*

Header

- primärer Header
- sekundärer Header (verschlüsselt)

Payload

- enthält Teile der Nachricht

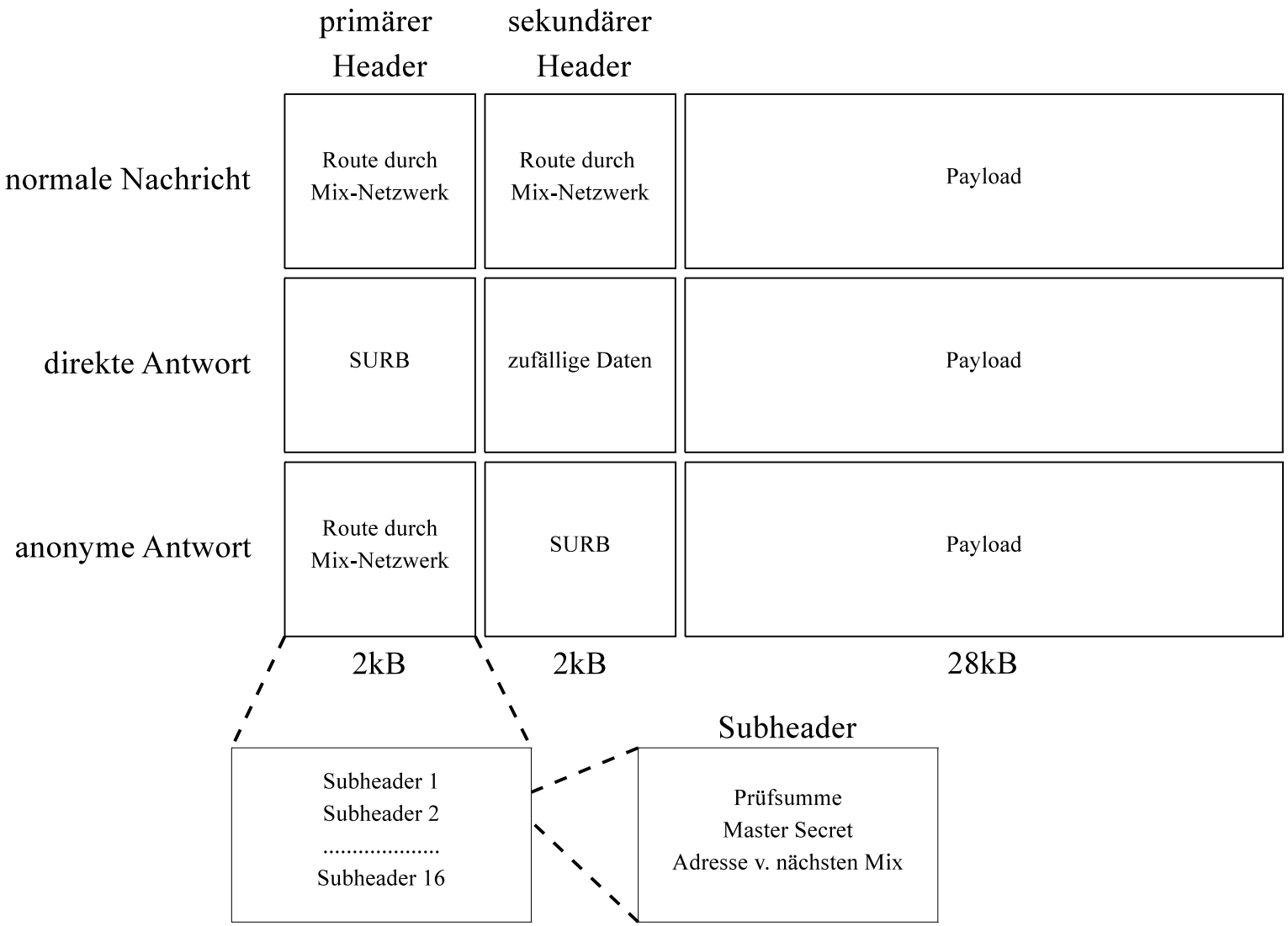
Mixminion

Drei verschiedene Nachrichtentypen (2)

- Remailerpfad wird in zwei Teile aufgeteilt
  - Primärer Header für ersten Teil des Pfades
  - Sekundärer Header für zweiten Teil des Pfades
- Für jeden Mix im Pfad enthalten Header jeweils einen Subheader
- Pfad enthält max. 32 Mixes
  - Primärer Header mit max. 16 Subheader
  - Sekundärer Header mit max. 16 Subheader
- Subheader enthalten
  - Prüfumme über Rest des Headers
  - *Master-Secret*
  - Adresse des nächsten Mixes

Mixminion

Drei verschiedene Nachrichtentypen (3)



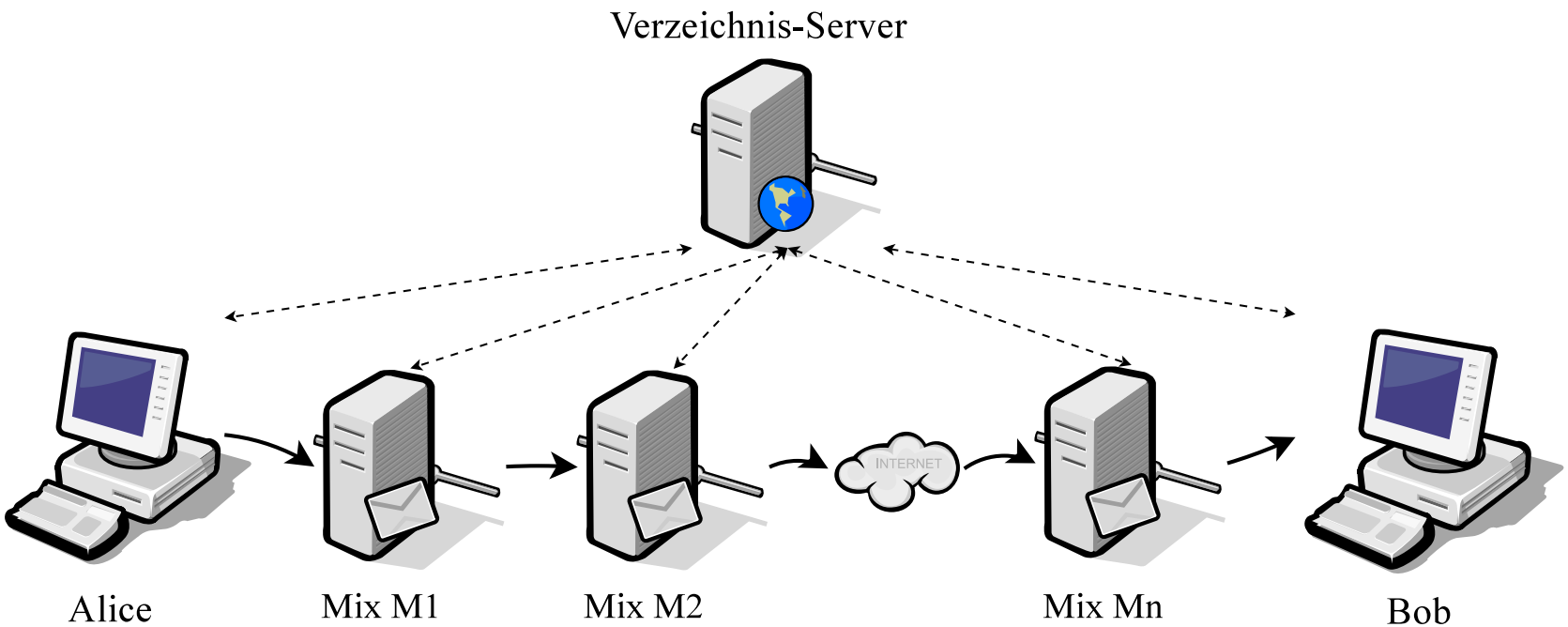
Mixminion

Einführung eines Verzeichnisservers (1)

- Gruppe von redundanten Servern
- Speichern Informationen wie Status und verwendete Schlüssel von Mixes
- regelmäßige Aktualisierung und Synchronisierung notwendig
  - Veraltete/unterschiedliche Informationen gefährden Anonymität
- Jeder Mix registriert sich beim Verzeichnisserver und sendet regelmäßig seine aktuellen Daten
- Benutzer können Informationen von Verzeichnisservern abrufen

Mixminion

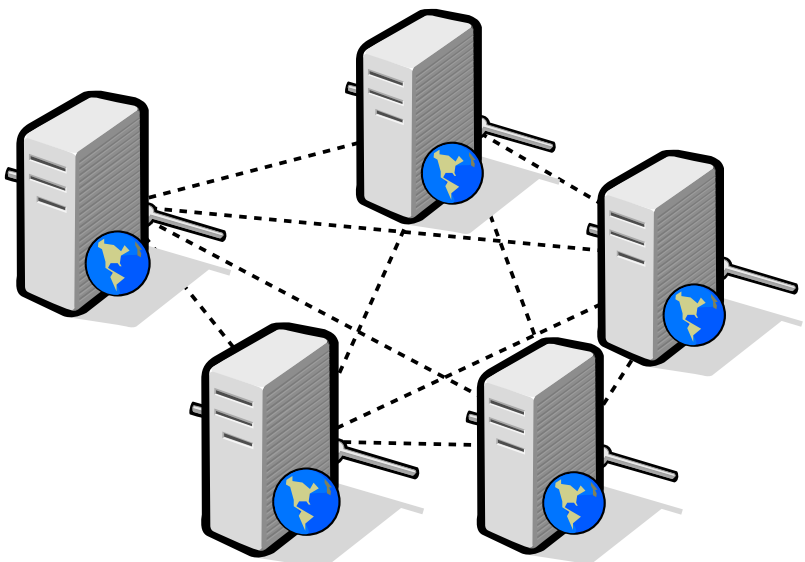
Einführung eines Verzeichnisservers (2)



Mixminion

Einführung eines Verzeichnisservers (3)

- Verzeichnisserver verifizieren sich gegenseitig
  - Jeder Server muss Glaubwürdigkeit der anderen Server bestätigen
  - Beruht auf der Annahme, dass nicht alle Server manipuliert sind
  - Im Zweifelsfall reicht 1 vertrauenswürdiger Server, um Anonymität zu gewährleisten



Mixminion

Ablauf des Versendens einer Nachricht (1)

- Alice holt sich nötige Informationen (Status, Schlüssel, Adresse von Mixes) von Verzeichnisserver
- Verschlüsselung der Nachricht äquivalent zu Mixmaster
- Jeder Mix
  - prüft Integrität der Daten anhand der Prüfsumme im Header
  - speichert Prüfsumme bestimmte Zeit lang, um Replay-Angriffe zu vermeiden
  - baut verschlüsselte Verbindung zum nächsten Mix auf
    - Berechnung von symmetrischen Schlüsseln anhand des Master Secret im Subheader
    - Verifizierung des nächsten Mixes
  - erneuert nach Übertragung der Nachricht symm. Schlüssel und Master Secret

Mixminion
Swap-Operation

- Nachricht wird solange übertragen, bis *Kreuzungspunkt* erreicht wird
- Liegt zwischen ersten und zeiten Teil des Remailerpfades
- Durchführung einer **swap**-Operation:
  - Sekundärer Header wird entschlüsselt
  - Vertauschung von primären und sekundären Header
- Verhindert Tagging-Atacken:
  - Sekundärer Header wurde mit Prüfsumme über Payload verschlüsselt
  - Wenn Payload manipuliert wurde, kann sekundärer Header nicht entschlüsselt werden
  - Nachricht kann nicht weiter versendet werden

Angriffe gegen Mixminion (1)

- Kaum Erfahrungen, da bisher nur im Teststadium
- Entwicklung wurde im Jahre 2007 eingestellt
- Deshalb: Fehler in der Implementierung wahrscheinlich

- In der Theorie gilt Mixminion als das sicherste Remailer-Protokoll
  - Zahlreiche Sicherheitsmechanismen zur Vermeidung von Angriffen

Angriffe gegen Mixminion (2)

Angriff	**Verteidigung**
Attacken gegen Mixes	
Manipulieren eines Mixes	keine Gefahr, solange nicht alle Mixes manipuliert
Wiedereinspielen von Nachrichten	Speichern von Prüfsummen; zusätzl: Keine Entschlüsselung nach Schlüsselrotation mehr möglich
Tagging-Attacke	Swap-Operation
Flooding-Attacke	Mixes mit Puffern erschweren Angriff

Angriffe gegen Mixminion (3)

Angriff	**Verteidigung**
Passive Attacken	
Traffic-Analyse	Einheitliche Paketgrößen; Zufällige Versende-Reihenfolge; Puffer
Attacken gegen Verzeichnisserver	
Manipulieren eines Verzeichnisservers	keine Gefahr, solange nicht alle Server manipuliert sind
Ausnutzen von unterschiedlichem Wissen v. Benutzern	Ständige Synchronisierung und Aktualisierung der Verzeichnisserver

Überblick

Zusammenfassung

- Remailer als anonymisierender Nachrichtenvermittler
- Remailer-Protokolle: Cypherpunk, Mixmaster, **Mixminion**
- viele Sicherheitsmechanismen, um Angriffe zu vermeiden:
  - einheitliche Paketgrößen
  - Puffer; zeitversetzte und zufällige Versendezeitpunkte
  - Swap-Operation
  - ...
- Besonderheit bei Mixminion:
  - Möglichkeit des Antwortens auf anonyme Nachrichten
- Allerdings:
  - Mixmaster beliebter und verbreiteter als Mixminion
  - Testphase von Mixminion $\rightarrow$ geringere Nuterbasis $\rightarrow$ weniger Mixes $\rightarrow$ Gefahr von manipulierten Mixes höher

Vielen Dank für die Aufmerksamkeit

Literaturverzeichnis

George Danezis, Roger Dingledine und Nick Mathewson
Mixminion: Design of a Type III Anonymous Remailer Protocol,
In Proceedings of the 2003 IEEE Symposium on Security and Privacy 2–15, 2003

Jens Kubieziel
Anonym im Netz - Techniken der digitalen Bewegungsfreiheit,
Open Source Press München, 2007

David L. Chaum
Untraceable electronic mail, return addresses, and digital pseudonyms,
Commun. ACM 24(2):84–90, 1981