UDP-/ICMP-Erweiterung für fwtest

Semesterarbeit Wintersemester 2005/06

Beat Strasser
Betreuerin: Diana Senn

Information Security

ETH Zürich

7. Februar 2006

Januar 2005

- DA Gerry Zaugg: *fwtest*-0.6

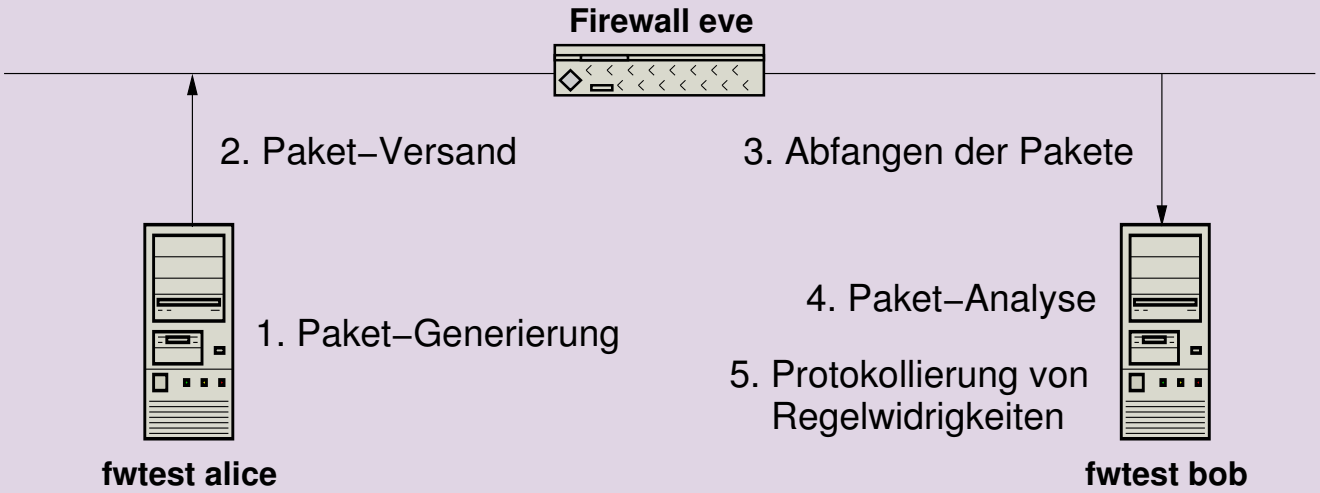
WS 05/06

- SA Adrian Schüpbach: Grössere Umstrukturierung, NAT-Tauglichkeit
- SA Beat Strasser: Erweiterung für UDP und ICMP
- Februar 2006: *fwtest*-1.0

Inhalt

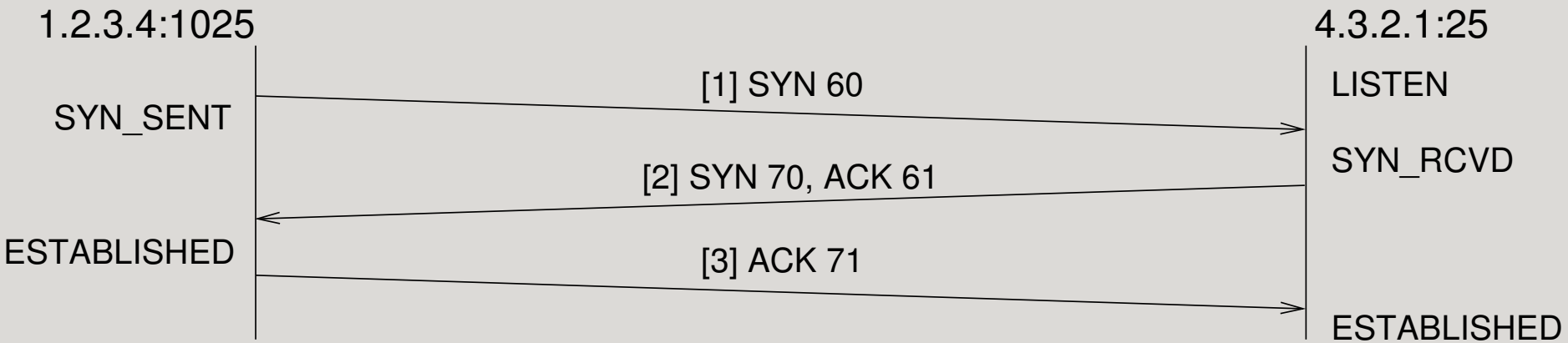
- Anhand geeigneter Testfälle eine Firewall analysieren, welche eine bestimmte Sicherheits-Policy implementiert.
- *fwtest* kann solche Testfälle auf eine Firewall anwenden.

Test-Szenario—Der Weg eines Testpakets



Beispiel eines TCP-Verbindungsaufbaus

```
#id expect time     srcip   dstip   srcprt dstprt flags seq ack
 1   OK     12:00:00 1.2.3.4 4.3.2.1 1025   25     S     60  -
 2   NOK    12:00:01 4.3.2.1 1.2.3.4 25     1025   SA    70  61
 3   NOK    12:00:02 1.2.3.4 4.3.2.1 1025   25     A     61  71
```



Aufgabenstellung

Ziel ist die Unterstützung von UDP- und ICMP-Testpaketen in *fwtest*.

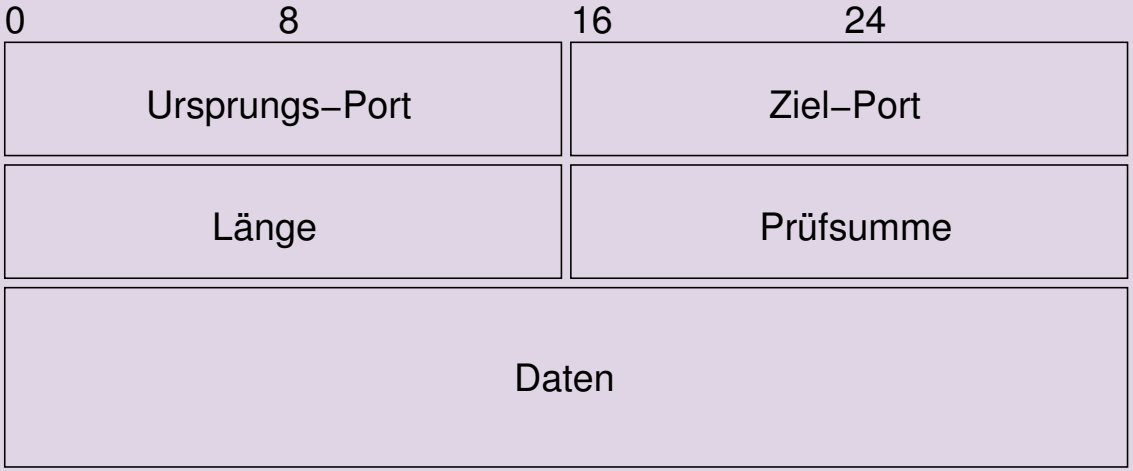
Teilaufgaben:

- Neues Format für Testpaket-Spezifikationen.
  - Design
  - Parser
- Generieren der UDP-/ICMP-Pakete.
- Abfangen/Analyse der Pakete.
- Testen von *fwtest*.

User Datagram Protocol (UDP)

- Protokoll in der Transport-Schicht (wie TCP).
- Verbindungslose, minimale Daten-Übertragung ohne Flusssteuerung und nicht auf Zuverlässigkeit ausgelegt.
- Geeignet für zeitkritische Anwendungen, wobei Verluste einzelner Pakete keine grosse Rolle spielen (Audio/Video Streaming, Voice over IP, DNS).

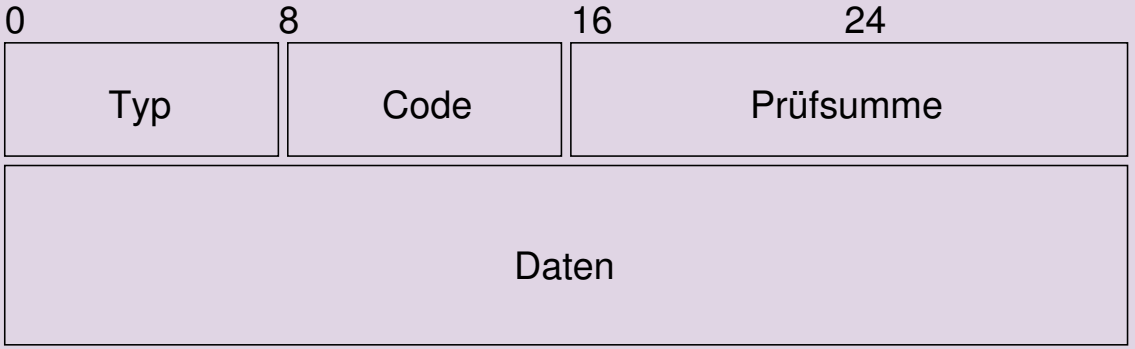
Aufbau UDP-Datagramm



Internet Control Message Protocol (ICMP)

- Bestandteil von IP, aber basiert auf IP-Paketen. Dient zum Austausch von Fehler- und Informationsmeldungen.
- Viele verschiedene Paket-Typen; nur 5 Typen implementiert in *fwtest*.

Aufbau ICMP-Datagramm



Echo

- Zum ‘Pingen’ eines Rechners im Netz.
- 2 Typen: Anfrage (Typ 8) und Antwort (Typ 0).
- Enthält Identifikations-/Sequenz-Nummern und Datenblock.

Timestamp

- Simple Uhrensynchronisation.
- 2 Typen: Anfrage (Typ 13) und Antwort (Typ 14).
- Enthält Identifikations-/Sequenz-Nummern und Timestamps.

Bei Fehlermeldungen werden zur Paket-Identifizierung beim Ursprungsrechner mindestens 64bit des originalen IP-Datagramms zurückgeschickt.

Destination unreachable

- Fehlermeldung (Typ 3) bei unerreichbarem Ziel (Rechner, Port, Protokoll...).

Redirect

- Fehlermeldung (Typ 5) informiert über bessere Routen zum Ziel-Netzwerk/-Rechner (für bestimmten ToS).
- Enthält IP des "näheren" Gateways.

Time exceeded

- Fehlermeldung (Typ 11) informiert, wenn TTL des Originalpakets abläuft vor der Ankunft am Zielrechner.

Generieren/Abfangen von UDP-/ICMP-Paketen

fwtest-0.6 benutzt Bibliotheken, die wir auch für UDP und ICMP gut gebrauchen können.

Generieren

- *Libnet*-Bibliothek unterstützt sämtliche von uns gewählten Protokolltypen.
- Prüfsumme wird automatisch berechnet durch *Libnet*.

Abfangen

- Verwendung der *Pcap*-Bibliothek.

Unterstützung mehrerer Protokolle zur gleichen Zeit

- *fwtest*-0.6: Mehrere Protokolle vorgesehen, aber nur ein Protokoll pro Testlauf.
- Trennung der Protokolle nicht realistisch (ICMP-Fehlermeldungen verweisen meist auf Pakete anderer Protokolle).
- Einzelne Paket-Spezifikation muss neu den Protokoll-Typen enthalten.
- Die Abfang-Paketlänge wird auf einen grösseren Wert gesetzt, so dass auch Protokolle mit mehr Daten (als TCP) unterstützt werden.

Protokollunspezifische Werte: ID, Empfangs-Erwartung, Protokoll-Typ, Ursprungs-/Ziel-IP.

Protokollspezifische Werte aller Protokoll-Typen

Protokoll-Typ	*Wert 1*	*Wert 2*	*Wert 3*	*Wert 4*	*Wert 5*
TCP	srcprt	dstport	flags	seqnr	acknr
UDP	srcprt	dstport			
TCPUDP	srcprt	dstport	flags	seqnr	acknr
ICMPecho	type	idnr	seqnr		
ICMPtstamp	idnr	seqnr	otime	rtime	ttime
ICMPunreach	code	origid			
ICMPredir	code	gwip	origid		
ICMPtexc	code	origid			

Referenzierung auf Original-Pakete

- ICMP-Fehlermeldungen müssen “Original”-Pakete zurück senden. Wir verweisen mittels Paket-ID darauf.
- Es wird einfachheitshalber nicht der IP-Header plus 64bit, sondern das ganze IP-Datagramm angehängt.
- Keine Zufallswerte mehr in Paketen.

Parser für neues TP-Spezifikations-Format

Beispiel Testpaket-Spezifikation

```
define(ipa, 172.16.70.3)
define(ipb, 192.168.72.7)
testcase 1 {
  packet 1 { udp send { ipa ipb 12300 domain }
             receive { ipa ipb 12300 domain } }
  packet 2 { icmpunreach send { ipb ipa PORT_UNREACHABLE  1.1 }
                      receive {} }
}
```

- Präprozessor *m4* für ICMP-Konstanten und selber definierte...
- Lexer mit *Flex*, Parser mit *Bison* generiert.

Fazit

- Erfolgreiche Implementation von UDP/ICMP.
- *fwtest*-0.6 gut vorbereitet für zusätzliche Protokolle.
- Vollständige Implementation von ICMP nur schwer möglich (viele verschiedene Untertypen).
- Einiger Mehr-Aufwand für ICMP als für UDP.