VSDB

Verschlüsselung

DES
Diffie-Hellman

Daniel Dimtrijevic Thomas Traxler

14. März 2014

5AHITT

Inhaltsverzeichnis

1 Aufgabenstellung

Kommunikation [12Pkt] Programmieren Sie eine Kommunikationsschnittstelle zwischen zwei Programmen (Sockets; Übertragung von Strings). Implementieren Sie dabei eine unsichere (plainText) und eine sichere (secure-connection) Übertragung.

Bei der secure-connection sollen Sie eine hybride Übertragung nachbilden. D.h. generieren Sie auf einer Seite einen privaten sowie einen öffentlichen Schlüssel, die zur Sessionkey Generierung verwendet werden. Übertragen Sie den öffentlichen Schlüssel auf die andere Seite, wo ein gemeinsamer Schlüssel für eine synchrone Verschlüsselung erzeugt wird. Der gemeinsame Schlüssel wird mit dem öffentlichen Schlüssel verschlüsselt und übertragen. Die andere Seite kann mit Hilfe des privaten Schlüssels die Nachricht entschlüsseln und erhält den gemeinsamen Schlüssel.

Sniffer [4Pkt] Schreiben Sie ein Sniffer-Programm (Bsp. mithilfe der jpcap-Library http://jpcap.sourceforge.net oder jNetPcap-Library http://jnetpcap.com/), welches die plainText-Übertragung abfangen und in einer Datei speichern kann. Versuchen Sie mit diesem Sniffer ebenfalls die secure-connection anzuzeigen.

Info Gruppengröße: 2 Mitglieder Punkte: 16

Erzeugen von Schlüsseln: 4 Punkte Verschlüsselte Übertragung: 4 Punkte Entschlüsseln der Nachricht: 4 Punkte Sniffer: 4 Punkte

2 Designüberlegungen

Die Verschlüsselung sollte im besten Fall mithilfe des Dekorator-Patterns erfolgen, da die einzige Anforderung verschlüsselte Kommunikation und kein anderes Anwendungsgebiet ist, können wir ohne jegliches schlechtes gewissen und auch hierauf einschränken, dies bedeutet wir erweitern schlicht BufferedReader und BufferedWriter welche bereits alle nötigen Methoden zur Kommunikation bieten und erweitern diese lediglich um die Funktion der Verschlüsselung, die Schlüsselerstellung wird in eine eigene Klasse ausgelagert und der Sniffer bildet ein eigenes Programm für sich.

3 Arbeitsaufteilung

Name	Arbeitssegment	Time Estimated	Time Spent
Thomas Traxler	Chat	1h	0.5h
Daniel Dimitrijevic	En-/Decrypt	2h	2h
Thomas Traxler	En-/Decrypt	1h	1h
Walter Klaus	Diffie-Hellman	1h	2h
Daniel Dimitrijevic	Sniffing	1h	2h
Gesamt		6h	7.5h

4 Arbeitsdurchführung

Beim Sniffer habe ich mir die Vorlagen von Jnetpcap angeschaut und diese zur Kreation meines Sniffers herangenommen. Im der ersten Phase des Sniffers suche ich mir die Liste an Adaptern heraus die auf die der Sniffer sniffen kann. Das hat am Anfang zu Problemen bei mir geführt da ich noch von den Vorjahren einige Loopback adapter hatte die ich erst deaktivieren musste. Nachdem ich dieses Problem gelöst hatte ging ich die Aufgabe der Packetausgabe an. Da hatte ich für die normale Ausgabe keine Problem, bis ich bemerkte das ich Limitieren musste welche Packete mitgesnifft werden müssen. Nach einer weile Recherche habe ich auch dieses Problem gelöst. Danach war die Ausgabe der Packete nur noch ein kleines Problem. Es kostet mich ein wenig extra Zeit herauszufinden mit welchem Befehl ich die Payload eines Packetes ausgeben kann und es in Hexadezimal auch anzuzeigen. Danach war nur noch die Anzahl der Durchläufe einzugeben und den Sniffer zuschließen.
Die Verschlüsselung erfolgt durch erweiterte Reader und Writer. Diese EncryptedReader und Writer beinhalten alle funktionen normaler Reader und Writer, die Kommunikation ist daher nicht weiter beeinflusst, zusätzlich benötigen sie jedoch im Konstruktor ein Bytearray als Key und bieten die möglichkeit der verschlüsselten Übertragung mit den Methoden ReadLineEnrypted und WriteLineEncrypted, welche alle übergebenen bzw. auszugebenden Nachrichten DES ver- bzw. entschlüsselt, hierbei wird immer der jeweilig übergebene Schlüssel verwendet. Für die Zukunft wäre es denkbar auch weitere Methoden des Readers und Wirters verschlüsselt zur Verfügung zu stellen oder aber auch die Verschlüsselungsmethode variabel zu machen. Die Verschlüsselung selbst wurde mit der Klasse javax.crypto.Cipher durchgeführt welche eine leichte Umsetzung hierfür bietet.

Die Schlüsselerstellung erfolgt mittels des DiffieHellmann Verfahrens, die hierzu benötigten Ausgangswerte können mittels eines Properties Objekts übergeben werden, ist dieses Leer oder unvollständig werden Defaultwerte verwendet. Die für dieses Verfahren notwendigen und verwendeten Algorithmen sind allgemein verfügbar und nahezu überall einsehbar.

5 Quellen

http://jnetpcap.com/