Karlsruher Institut für Technologie

Broadcast-Encryption : Validierung

Praxis der Softwareentwicklung WS 2012/13

Team : Amrei Loose, Niklas Baumstark, Christoph Michel, Tobias Gräf, Mohammed Abu Jayyab
Betreuer : Carmen Kempka, Christoph Striecks

INSTITUT FÜR KRYPTOGRAPHIE UND SICHERHEIT / EUROPÄISCHES INSTITUT FÜR SYSTEMSICHERHEIT

Inhaltsübersicht

- Einleitung
- Ablauf
- Statistik
- Frameworks und Tools
- Tests
  - Modultests (Unit Tests)
  - Oberflächentests (UI Tests)
- Testabdeckung
- Fazit

Institut für Kryptographie und Sicherheit
Europäisches Institut für Systemsicherheit

Einleitung

Warum testen wir überhaupt ?

- Sicherstellen, dass unser System
  - einwandfrei funktioniert.
  - seine Spezifikationen und Anforderungen erfüllt.

- Erhöhen von Vertrauen in unserer Software.
- Erleichterung der Erweiterbarkeit.

Ablauf

- Bereits mit Beginn der Implementierungsphase wurde fortwährend getestet.
- Es bestand kein Bedarf für ein Gruppentreffen.
- Es musste lediglich das Dokument aufgefasst werden.
- Deshalb gibt es keine richtige Zeitaufteilung.

Institut für Kryptographie und Sicherheit
Europäisches Institut für Systemsicherheit

Statistik

- 727 Git-Commits.
- ~7400 Lines of Code.
  - 1700 Unit-Tests.
  - 400 C++.
- 140 Java-Dateien
  - 36 Test-Klassen.
- 189 Typen.

Institut für Kryptographie und Sicherheit
Europäisches Institut für Systemsicherheit

Frameworks und Tools

- JUnit: Das Standard-Framework für Java-Unit-Tests.
- Robolectric: Laufzeitumgebung für Client-Tests für Android.
- Mockito: Objekt-Mocking für Logik-Tests.
- EclEmma: Ein Eclipse-Plugin, das die Codeüberdeckung beim Testen eines Programms misst, indem es die Anzahl aller Instruktionen ins Verhältnis zu sämtlichen erreichten Instruktionen setzt.

Tests

- Zwei Kategorien:
  - Unit Tests: Testen die Logik und Kernfunktionalität eines Programms.
  - UI Tests: Beschäftigen sich mit der Benutzeroberfläche und ihre Funktionsweise.

- Positive Tests: Testen das Verhalten bei erwarteten und gültigen Eingaben.
  - Sicherstellen von Korrektheit.
- Negative Tests: Testen das Verhalten bei unerwarteten und ungültigen Eingaben.
  - Sicherstellen von Robustheit.

Institut für Kryptographie und Sicherheit
Europäisches Institut für Systemsicherheit

Modultest (Unit Tests)

- Kryptographie
  - Implementierung basiert auf Naor-Pinkas Protokoll.
  - Enthält Logik für die Generierung von verschlüsselten Streams.
    - Polynomial und Lagrange Evaluation.
    - Berechnung von Modulo und elliptischen Kurven.
  - Entschlüsselung von Naor-Pinkas-verschlüsselten Streams.
  - Sowie Logik für die Schlüssel-Generierung.

Modultest (Unit Tests)

- Kommunikation
  - Kommunikation zwischen Client und Sever.
  - Verschiedene Kommunikation und Stream-Konstrukte.
  - Senden von verschlüsselten Streams.
  - Empfangen von gesendeten Streams.
- Server
  - Basiert auf die bestehenden Module.
    - Kryptographie und Kommunikation.
  - Verwaltung der Benutzer und ihre Schlüssel.
  - Behandelt das Broadcasting.

Modultest (Unit Tests)

- Client
  - Basiert auf die bestehenden Module.
    - Kryptographie und Kommunikation.
  - Bringt keine neue Logik mit.
    - Benutzt Java und Android API ohne neue Konstrukte oder Logik zu definieren.
  - Relativ weniger getestet.
    - Hauptsächlich Oberflächentests.

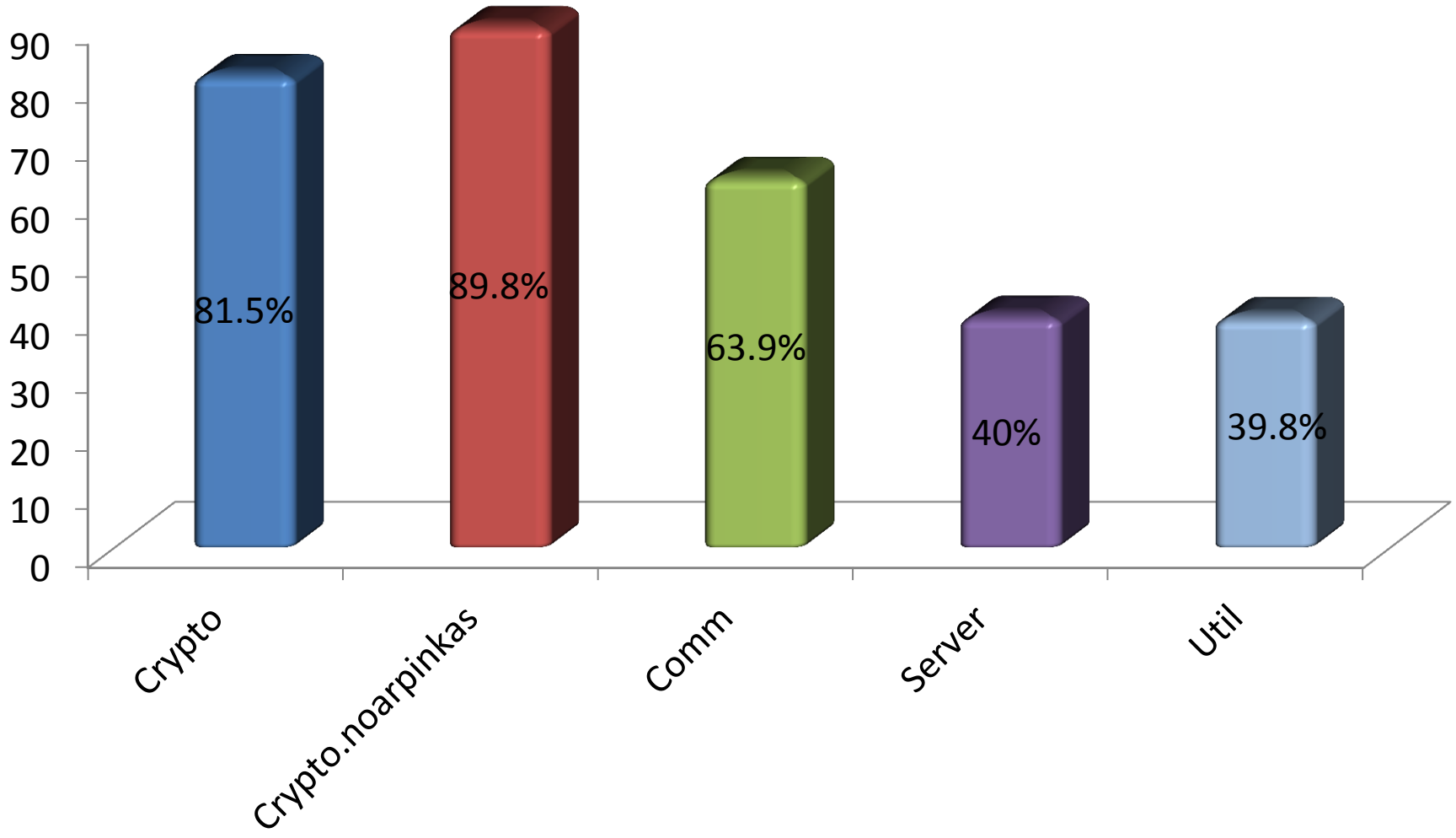
Oberflächentests (UI Tests)

- Die Oberflächentests werden von Hand durchgeführt.
- Orientieren sich an den im Pflichtenheft definierten Testszenarien.
- Client läuft auf einem Android-Gerät.
- Testen spezifischer Funktionalitäten zur Client-Server-Interaktion.

Testszenarien

- Normaler Programmablauf, indem alle Aktionen von einem autorisierten Benutzer durchgeführt werden.
- Programmablauf, indem alle Aktionen von einem ausgeschlossenen Benutzer durchgeführt werden.
- Programmablauf, indem sich der Benutzer zu einem bereits besuchten Server verbindet.
- Programmablauf, indem der Benutzer die Serverdaten oder den privaten Schlüssel falsch angegeben hat.

Testabdeckung



- ~ 66% Testabdeckung (>85% bei Kernfunktionalität)

Testabdeckung

- Die angegebenen Werte stellen nur eine grobe Richtlinie dar.
- Kryptographie Modul hat die höchste Rate an Testabdeckung , da hier die wesentliche Funktionalität und Logik steckt.
- Die Testabdeckung für Utilities ist relativ niedrig, weil die meisten enthaltenen Methoden Wrapper oder Hilfsfunktionen um Standard Java API sind.
- Der Server Projekt basiert auf die Funktionalität von den Kryptographie und Kommunikations-Modulen, ohne zusätzliche oder neue Logik zu implementieren.

Fazit

- Testen parallel zur Entwicklung verhindert viele Fehler und Bugs in den späteren Phasen eines Projekts.
- Insgesamt ist die Überdeckung durch die automatisierten Tests zufriedenstellend.
- Unit Tests zusammen mit den Oberflächentests bieten einen hinreichenden Masstab für die Robustheit und Fehlerfreiheit unseres Projekts.

Vielen Dank für eure Aufmerksamkeit

Karlsruher Institut für Technologie
Institut für Kryptographie und Sicherheit
Europäisches Institut für Systemsicherheit