– Praktikumsaufgabe 1 –

Thema: *Passphrasen, Codeanalyse, ROT-13*

Zielstellung: Einführung der Thematik anhand typischer Problemstellungen; Detektierung schwacher Passphrasen, Analyse unbekannten Codes, Trivialverschlüsselung ROT-13; (Wieder-)Erlernen der C-Programmierung mittels gcc unter UNIX

1. Klicken Sie sich kurz durch `http://minimizedistraction.com/` und versuchen Sie, die Grundidee zu beherzigen.

   Stellen Sie sich vor, Sie seien ein Angreifer und haben durch einen Exploit sich einen lokalen Zugang zu seinem Rechensystem verschafft und waren durch eine (grobe) Unachtsamkeit des Administrators zusätzlich in der Lage, sich eine Kopie der Passwortdatei anzulegen. Versuchen Sie, so viele Passworte wie möglich aus der bereitgestellten Datei `http://www.informatik.htw-dresden.de/~robge/is/auth/hash.zip` zu extrahieren.

   Was können Sie über die geknackten Passphrasen aussagen?

   Hinweise:

   - Prüfen Sie zu Beginn, ob Sie genügend Speicherplatz in Ihrem Home-Verzeichnis haben und räumen Sie dieses ggf. auf. Sie benötigen etwa 330 MiB (Die Hashdatei umfasst komprimiert 17 MiB, unkomprimiert etwa 50 MiB, dazu für Quellen und Binaries von `john` ca 260 MiB).

   - Die aktuelle so genannte „community-enhanced“-Version müssen Sie aus einem git-Repository clonen:

     ```
     git clone git://github.com/magnumripper/JohnTheRipper -b bleeding-
         jumbo john
     cd john/src
     ./configure
     make -s clean && make -sj4
     ```

   - Das Archiv enthält so genannte ungesalzene („rohe“) MD5-Hashes; dies müssen Sie John beim Aufruf mitteilen (Option `--format=raw-md5`).

2. Stellen Sie sich vor, Sie finden auf Ihrem Rechner die Datei `system.c` Analysieren Sie den Code! Worum handelt es sich? Wie arbeitet das Programm? Ist es korrekt, oder weist es offenkundige Fehler auf?

   Achtung! Führen Sie das kompilierte Programm **keinesfalls** aus! (Man sollte generell nur Code ausführen, den man entweder vollständig versteht oder dem man vertrauen kann.)

3. Schreiben Sie ein Programm, das zeilenweise von `stdin` liest, jedes eingelesene Zeichen ROT-13-kodiert und die kodierte Zeile nach `stdout` schreibt. Verschlüsseln Sie

dann einen beliebigen Text mit Ihrem Programm (Nutzen Sie die Kommandozeile!)! Was passiert bei zweimaliger Verschlüsselung? Überlegen Sie sich einen passenden Mechanismus zur Beendigung des Programms.

ROT-13 bedeutet, dass jeder Buchstabe um 13 Positionen im Alphabet nach vorn rückt mit Wrap-Around ('a' → 'n', 'b' → 'o', . . . , 'm' → 'z', 'n' → 'a' usw.). Vergleiche auch *http://de.wikipedia.org/wiki/ROT-13*, aber schreiben Sie den Code nicht ab!

Bewerten Sie die Sicherheit dieses „Verschlüsselungsverfahrens".