PS Kryptographie: Programmierprojekt

Dominik Baumgartner, Samy Dafir, Vivien Wallner

1 Projektbeschreibung

Ziel dieses Projekts ist es, ein Programm zu entwickeln, welches vortäuscht eine sinnvolle Funktion zu erfüllen, in unserem Fall die, einen Ordner zu säubern oder zu optimieren. Dieses Programm führt jedoch statt der Optimierung eine Verschlüsselung aller in diesem Ordner enthaltenen Dateien durch. Anschließend erhält das Opfer der Schadsoftware eine Aufforderung, eine bestimmte Geldmenge zu überweisen, um den Entschlüsselungs-Key zu erhalten. Nach Erhalt des Schlüssels kann die Entschlüsselung mit dem gleichen Tool vorgenommen werden.

2 Ablauf

Das Opfer öffnet die Anwendung und erhält eine Meldung, dass der jeweilige PC nicht optimiert ist. Daraufhin wählt es einen Ordner aus, der öptimiert", also gesäubert werden soll. Mit etwas Glück für den Malware-Ersteller, enthält der gewählte Ordner wichtige Dokumente. Diese Dokumente werden dann verschlüsselt, und dem Benutzer eine Meldung präsentiert, wonach er eine bestimmte Geldsumme an einen gegebenen PayPal-Account überweisen soll. Ist dies geschehen, erhält das Opfer (eventuell) den Schlüssel, um die Daten zu entschlüsseln. Die Entschlüsselung muss mit dem gleichen Malware Tool erfolgen wie die Verschlüsselung. Das Tool entschlüsselt die Dateien anschließend mit dem angegebenen Schlüssel. Als zusätzliche Unannehmlichkeit für den Benutzer wurde das Programm folgendermaßen ausgelegt:

- Der Entschlüsselungsdialog ist nur über den Verschlüsselungsdialog erreichbar. Das heißt: Sollte der Benutzer das Programm schließen, muss er es erneut starten, um den Dialog zu erreichen. Dadurch wird der Ordner erneut verschlüsselt. Da die Entschlüsselungs-Funktion nur einen Entschlüsselungsdurchgang durchführt, ist es für den Normal-Benutzer nicht mehr möglich, die Daten wiederherzustellen.

- Sollte der Benutzer versuchen, die Daten unter Angabe eines falschen Schlüssels zu entschlüsseln, wird der Algorithmus mit dem angegebenen Key ausgeführt. Sollte das Opfer sich dann doch entscheiden, den Key zu kaufen und die Daten zu entschlüsseln, sit dies nicht mehr möglich, da die Dateien ja bereits durch Verwendung des falschen Keys verändert wurden. In diesem Fall ist es dem Benutzer also ebenfalls unmöglich, die ursprünglichen Dateien wiederherzustellen.

3 Implementierung

Das Programm wurde in Java 8 implementiert. Für die Benutzer-Oberfläche wurde JavaFx verwendet.
Es unterstützt die Verschlüsselung und Entschlüsselung reiner ASCII-Dateien. Andere gängige Dateiformate können zwar ver- jedoch nicht mehr erfolgreich entschlüsselt werden. Das Programm arbeitet wie folgt:

1. Eine neue Datei mit Namen "temp.txt"wird erstellt
2. Ein Zeichen der zu verschlüsselnden Datei wird gelesen (mittels BufferedReader)
3. Das Zeichen wird verschlüsselt (genaueres folgt unten)
4. Das verschlüsselte Zeichen wird in die neue Datei geschrieben (mittels BufferedWriter)
5. Der Ablauf wird ab Schritt 2 für jedes Zeichen in der Datei wiederholt
6. Sind alle Zeichen verschlüsselt, wird die Originaldatei gelöscht und die verschlüsselte auf deren Namen umbenannt
7. Ablauf ab Schritt 1 für jede Datei im Ordner wiederholen
8. Sind alle Datein durchlaufen, ist der Vorgang beendet

3.1 Ver- und Entschlüsselung

Als Verschlüsselung wird ein Vigenere-Cipher mit einem aus 31 zufällig gewählten Zahlen bestehenden Schlüssel verwendet. Diese Zahlen bilden das Array rot.
Es wird ein bei null startender Index i definiert. Je ein Zeichen aus der Originaldatei wird gelesen und um die Zahl an der Stellt $rot[i]$ zu ihrem ASCII-Wert hinzu addiert (natürlich mit wrap-around). Dies ergibt jetzt ein neues ASCII-Zeichen. Danach wird i um 1 erhöht. Damit werden jeweils 31

aufeinander folgende Zeichen verschieden rotiert. Ist das letzte Element in *rot* erreicht, beginnt i wieder bei 0. Die Entschlüsselung funktioniert exakt gleich, nur wird hier in die andere Richtung rotiert, sprich $rot[i]$ subtrahiert.

Im Weiteren folgt noch eine Erläuterung zu einer Ciphertext-only-Attacke, die eine Methode darstellt, um Vigenere-Cipher zu knacken. Durch deren Anwendung können auch die von unserem Programm verschlüsselten Dateien wieder entschlüsselt werden und somit eine Bezahlung von "Lösegeld"vermieden werden.

4 Ciphertext-only-Attacke

Der Kasiski-Test ist ein Hilfsmittel zur Entzifferung von Ciphertexten, die mit dem Vigenère-Chiffre erzeugt wurden. Zuerst durchsucht man den Geheimtext nach Buchstabenfolgen der Länge 3 oder länger, die mehrmals vorkommen. Anschließend bestimmt man den Abstand zwischen je 2 gleichen Folgen, das heißt, man zählt die Buchstaben vom ersten Buchstaben der ersten Folge bis zum ersten Buchstaben der zweiten Folge. So verfährt man mit allen gefundenen Folgen und schreibt die Abstände auf. Man erhält eine Liste von natürlichen Zahlen. Diese Zahlen werden nun in Primfaktoren zerlegt, wodurch sich gleiche Teiler leichter finden lassen. Diese ermittelte Zahl liefert einen Aufschluss über die Schlüssellänge. Allerdings wird die genaue Schlüssellänge nicht bekannt, denn der Kasiski-Test liefert nur Vielfache der Schlüssellänge. Zur genauen Betrachtung kann aber der Friedman-Test herangezogen werden, der zusätzlich einen Hinweis darauf gibt, ob es sich um eine mono- oder polyalphabetische Verschlüsselung handelt.

Tritt im Ciphertext eine Buchstabenfolge zweimal auf und wurde mit ihr dasselbe Wort verschlüsselt, so ist der Abstand zwischen den beiden Folgen ein Vielfaches der Schlüsselwortlänge. Beim Kasiski-Test wird nach gleichen Buchstabenfolgen im Ciphertext gesucht. Man setzt nun voraus, dass sie dasselbe Wort verschlüsseln. Stimmt das, so ist der Abstand ein Vielfaches der Schlüsselwortlänge. Wurde aber nicht dasselbe Wort verschlüsselt, ist der Abstand kein Vielfaches der Schlüsselwortlänge, und die beiden Stellen im Geheimtext sind nur zufällig gleich. Natürlich erkennt man nicht sofort, ob „zufällig“ dieselbe Zeichenfolge entstanden ist, oder ob wirklich dasselbe Wort verschlüsselt wurde. Deshalb werden am Ende auch gemeinsame Faktoren gesucht, um die „unpassenden“ Abstände zu finden. Selbstverständlich passiert es vor allem bei kurzen Folgen, dass sie zweimal vorkommen, obwohl nicht dasselbe Wort verschlüsselt wurde. Das ist auch der Grund, warum

man in der Regel nicht nach gleichen Folgen der Länge 2 sucht. Die Wahrscheinlichkeit, dass die Buchstabenfolgen im Klartext nicht übereinstimmen, ist einfach zu groß.

Mit der herausgefundenen Schlüssellänge lässt sich nun eine Häufigkeitsanalyse im Ciphertext durchführen. Die ermittelten Häufigkeiten können dann mit den Buchstabenhäufigkeiten der jeweiligen Sprache abgeglichen werden, um herauszufinden, welcher Buchstabe jeweils verschlüsselt wurde.
Quelle: `https://de.wikipedia.org/wiki/Kasiski-Test`