
EIN WEB-MAGAZIN VON INDUSTR.COM

Zu viel Aufwand für den Hacker

# SOFTWARESCHUTZ IN 7 SCHRITTEN

**TEXT: RÜDIGER KÜGLER, WIBU-SYSTEMS**

**11.10.2017 | Mit der Blurry-Box-Technologie lässt sich Software gegen Reverse Engineering und das Raubkopieren schützen. Das funktioniert so gut, dass sich bei einem Hackers Contest selbst Meister ihres Fachs die Zähne daran ausgebissen haben.**

TAGS | SCHÄDLICHE SOFTWARE HACK HACKING HACKERANGRIFF HACKER RAUBKOPIEN DATENVERSCHLÜSSELUNG REVERSE ENGINEERING WIBU-SYSTEMS AG

Dem Kerckhoffs'schen Prinzip zufolge sollte die Sicherheit eines kryptographischen Verfahrens auf der Geheimhaltung des Schlüssels und nicht auf der Geheimhaltung des Algorithmus basieren. Diese Maxime aus dem Jahre 1883 ist heute aktueller denn je. Öffentlich bekannte Verschlüsselungsverfahren sind in der Regel besser untersucht als geheime Algorithmen. Haben sie einer Überprüfung standgehalten, sind sie deshalb vertrauenswürdiger.

## Schlüssel können leichter geheim gehalten werden als Algorithmen nicht

Aber auch andere Gründe sprechen für das Prinzip. Es ist zum Beispiel schwieriger, einen Algorithmus geheim zu halten als einen Schlüssel, da sich Algorithmen durch Reverse Engineering der Soft- und Hardware analysieren lassen. Algorithmen können außerdem kompromittiert werden, wenn Insider, meist unzufriedene Mitarbeiter, Firmengeheimnisse verkaufen. Zudem ist es in der Praxis deutlich aufwendiger, den Algorithmus auszutauschen als lediglich einen Schlüssel. Wird ein Algorithmus offengelegt, wird er auch stärker nach

Hintertüren, Implementierungsfehler und Sicherheitslücken überprüft. Diese können von den Programmierern dann beseitigt werden.

In der Vergangenheit basierte Softwareschutz meist auf Geheimniskrämerei. Jeder Anbieter hatte den vermeintlich sichersten und einfachsten Schutz implementiert. Marketing-Prospekte überschlugen sich mit Superlativen. Ein Vergleich verschiedener Anbieter war auf Grund der Geheimhaltung jedoch nicht einmal ansatzweise möglich, geschweige denn eine neutrale Prüfung.

## Hacker kann Software ohne Lizenz laufen lassen

Ein beliebtes Verfahren ist zum Beispiel die Verwendung von Kopierschutzsteckern, auch Dongles genannt. Dabei wird ein Algorithmus im Dongle implementiert und aus der Software heraus aufgerufen. In der Software sind Tabellen mit Anfragen und den dazu passenden Antworten abgelegt. Wenn die vom Dongle gelieferte Antwort mit der erwarteten Antwort übereinstimmt, wird die Software korrekt ausgeführt, andernfalls wird eine Fehlermeldung angezeigt und die Software beendet. Im Wesentlichen reduziert sich der Schutz also auf eine simple Ja/Nein-Entscheidung. Auf Assembler-Ebene ist dies ein JNZ oder ein JZ (Jump Not Zero oder Jump Zero) mit den entsprechenden binären Codes 74 beziehungsweise 75. Ein Hacker muss also nur an der richtigen Stelle eine 74 durch eine 75 ersetzen oder umgekehrt, und schon läuft die Software ohne Lizenz. In den meisten Fällen wurden Dongles deshalb bereits nach kurzer Zeit geknackt.

Heute setzen alle großen Anbieter auf Standard-Algorithmen im Dongle. In der Regel wird AES verwendet mit einer Schlüssellänge von 128 oder 256 Bit. Für die Sicherheit ist die Länge des Schlüssels aber nicht das entscheidende Kriterium. Wichtiger ist, mit welcher Technik der Algorithmus in der Software abfragt wird. Die meisten Anbieter geben zwar an, dass die Software verschlüsselt ist und nur mit dem passenden Dongle entschlüsselt werden kann. Damit hört das offengelegte Wissen aber in den meisten Fällen auf.

An dieser Stelle beginnt die Erfolgsgeschichte des Softwareschutzes Blurry Box. Entworfen wurde er von Forschern des Karlsruher Institut für Technologie (KIT) und Entwicklern des Securityunternehmens Wibu-Systems. Ihr Ziel war es, einen Softwareschutz zu entwickeln, bei dem auch die Integration in die Software offengelegt werden kann. Das Ergebnis dieser Kooperation erwies sich als so überzeugend, dass Blurry Box 2014 auf Anhieb den 5. Deutschen IT

Sicherheitspreis gewann.

## Den Hacker mit zu viel Aufwand abschrecken

Blurry Box funktioniert folgendermaßen: Hacker, die einen Softwareschutz entfernen wollen, gehen entweder statisch oder dynamisch vor. Bei der statischen Analyse wird der Code nur angesehen, verschlüsselte Teile werden gefunden, manuell entschlüsselt und wieder zusammengesetzt. Bei der dynamischen Analyse hingegen wird die ablaufende Software beobachtet und Teile aus dem Speicher, vor allem bereits entschlüsselter Code, werden entnommen. Die Blurry-Box-Technologie soll beide Angriffsformen sowie beliebige Kombinationen daraus verhindern. Ein hundertprozentiger Schutz ist unmöglich, da jede Software irgendwann einmal irgendwo ausgeführt wird. Sinnvoll ist es deshalb den Aufwand für das Hacken der Software soweit zu vergrößern, das er gleich oder höher ist als der Aufwand für die Neuerstellung einer vergleichbaren Software.

Folgende Grundannahmen wurden für Blurry Box außerdem getroffen:

- Die Software ist hinreichend komplex.
- Der Hacker versteht sein Handwerk.
- Der Hacker kann die Funktionsweise der Software nicht im Detail beurteilen.

Vor allem die erste Annahme ist wichtig, um das Schutzziel zu erreichen. Eine Anwendung, die lediglich eine Ausgabe in der Kommandozeile ausspuckt, ist nicht komplex genug, um mit Blurry Box geschützt zu werden. Schließlich braucht es nur wenige Minuten, um die Anwendung neu zu schreiben. Bei einer realen Anwendung verwendet ein normaler Anwender circa 10 Prozent der verfügbaren Funktionen. Testprogramme schaffen ungefähr eine Codeabdeckung von 80 Prozent. Anhand dieser Zahlen lassen sich die Herausforderungen bei einer dynamischen Analyse erahnen. Der Angreifer müsste alle Funktionen einmal aufrufen, um diese beobachten zu können. Das ist eine Aufgabe, die selbst die interne Qualitätssicherung des Softwareherstellers nicht zu 100 Prozent erfüllen kann! Wenn die Anwendung hinreichend komplex ist, wird ein rein dynamischer Hack deshalb immer unvollständig sein.

## In 7 Schritten zum Softwareschutz

Blurry Box besteht aus sieben Einzelschritten, die teilweise

bereits im Feld erprobt, im Rahmen von Blurry Box verbessert oder neu erfunden wurden.

**Schritt 1 - Bilden von Varianten:** Jede Software besteht aus einzelnen Methoden, die zur Laufzeit ausgeführt werden. Blurry Box erschafft Varianten dieser Methoden. Beispielsweise werden aus einer Methode sechs neue Methoden erzeugt. Zusätzlich wird eine Auswahlmethode (Wrapper) eingefügt. Der Wrapper wird anstelle der originalen Methode aufgerufen und wählt abhängig von den Eingabeparametern eine der neuen Methoden aus, die dann aufgerufen wird. Die Varianten erhöhen die Komplexität einer dynamische Analyse. Um nämlich alle Varianten beobachten zu können, müsste der Angreifer die Wertebereiche aller Parameter kennen und die Programmdurchläufe so modifizieren, dass sie alle Varianten einmal aufrufen.

**Schritt 2 - Modifikation der Varianten:** Gemäß dem Kerckhoffs'schen Prinzip kennt der Angreifer Blurry Box, das heißt er weiß, dass eine Auswahlmethode eingefügt wurde. Daher könnte er den Wrapper finden und so einstellen, dass immer die Variante 1 verwendet wird. Das wird verhindert, indem die Varianten so modifiziert werden, dass sie nur im entsprechenden Wertebereich der Eingabeparameter korrekt funktionieren. Dadurch liefert ein potentieller Variante-1-Hack fehlerhafte Ergebnisse. Die Modifikationen reichen von einfachen Wertebereichsprüfungen über Hinzufügen von „* 1“ und „+ 0“, wenn die Parameter genau diese Werte annehmen, bis zur Optimierung der Methoden. Es werden etwa Operationen weggelassen oder Variablenlängen verkürzt. Eine Rücknahme der Modifikation erfordert eine detaillierte Analyse jeder Funktion sowie ein fundiertes Wissen über die Funktionsweise der Software. Das erschwert die Arbeit des Hackers deutlich.

**Schritt 3 - Verschlüsselung der Varianten:** Jede einzelne Variante wird mit AES verschlüsselt. Eine Analyse des Codes in der Variante ist damit ohne Schlüssel überhaupt nicht und mit Schlüssel erst nach Entschlüsselung möglich. Diese Technologie kommt im Rahmen der Codemeter Protection Suite von Wibu bereits seit vielen Jahren zum Einsatz. Neu an Blurry Box ist, dass sie auf die einzelnen Varianten angewendet wird, während im klassischen Fall einfach die originale Methode verschlüsselt wurde.

**Schritt 4 - Einfügen von Fallen:** Der Codemeter-Dongle (CmDongle) besitzt die Eigenschaft, dass eine entsprechende Sperrsequenz die AES-Schlüssel im Dongle ungültig machen kann. Bei Blurry Box werden weitere Methoden in die Anwendung eingefügt, die als Fallen dienen. In der obigen

Abbildung sind zum Beispiel vier Fallen im Code platziert. Bevor die eingefügten Methoden verschlüsselt werden, wird an deren Anfang eine Sperrsequenz geschrieben. Die Auswahlfunktion wird so modifiziert, dass sie für ungültige Eingabeparameter diese Fallen-Methoden aufruft. Ohne ein tieferes Verständnis der Programmlogik kann der Hacker nicht entscheiden, ob es sich um eine Falle oder eine benötigte Variante handelt. Entschlüsselt der Hacker eine dieser Fallen, so wird der CmDongle gesperrt und eine Entschlüsselung der nachfolgenden Methoden ist unmöglich. Jetzt müsste er sich einen neuen CmDongle mit der passenden Lizenz besorgen, was bei einer hinreichend großen Anzahl von Fallen nicht im Verborgenen erfolgen kann.

**Schritt 5 - Variantenauswahl im Dongle:** Durch eine Codeanalyse der Auswahlfunktion wäre es jedoch noch möglich, Wertebereiche und entsprechende Fallen zu erkennen. Um das komplett zu verhindern, wird die Variantenauswahl in den CmDongle verlagert. Die Auswahlfunktion schickt eine ID und einen Parameter an den CmDongle. Dieser entscheidet dann, welche der Varianten ausgewählt werden soll. In der Grafik auf der nebenstehenden Seite trifft er beispielsweise die Wahl zwischen 10 Varianten.

**Schritt 6 - Dongle als State-Engine:** Die ersten fünf Schritte von Blurry Box erschweren die Analyse der Software. Dennoch wäre es denkbar, dass ein Hacker durch systematisches Abfragen des CmDongles die Zuordnung der einzelnen Varianten zu den Wertebereichen der Parameter herausfindet. Um das zu verhindern, dient der CmDongle als State-Engine. Indem Ketten von erlaubten Zuständen gebildet werden, weiß der Softwareentwickler, welche Methoden vor beziehungsweise nach einer bestimmten Methode aufrufbar sind. Diese Zustände werden im CmDongle gespeichert. Dieser antwortet nur noch auf die Anfrage der Variantenauswahl, wenn vorher die richtige Vorgängerfunktion aufgerufen wurde. Für den Hacker ist es damit nicht mehr möglich, jede Auswahlfunktion einzeln zu analysieren.

**Schritt 7 - Geschwindigkeitsbeschränkung:** Wie auch im Straßenverkehr ist beim Entschlüsseln die maximale Geschwindigkeit manchmal eher gefährlich als nützlich. Der Entwickler weiß genau, wie seine Software funktioniert und in welchen zeitlichen Abständen geschützte Methoden entschlüsselt werden. Daraus ergibt sich eine sinnvolle Geschwindigkeit für die Entschlüsselung im CmDongle. Indem der Entwickler bei Blurry Box eine durchschnittliche Pausenzeit zwischen zwei Entschlüsselungsoperationen

festlegt, dauert es für einen Hacker nochmals deutlich länger, alle Variantenkombinationen durchzuprobieren.

Blurry Box sorgt somit dafür, dass die Analyse für den Hacker so verkompliziert und die Zeit für die Erstellung eines Hacks so erhöht wird, dass es wirtschaftlicher ist, die Software komplett neu zu schreiben. Dabei kann der Hacker alle von Blurry Box verwendeten Strategien kennen, ohne dass ihm daraus ein Vorteil erwächst. Damit wird Blurry Box nicht nur dem Kerckhoffs'schen Prinzip gerecht, sondern macht es endlich auch möglich, Softwareschutz zu messen und zu vergleichen.

Nach dem Gewinn des 5. Deutschen IT Sicherheitspreises wurde die Blurry-Box-Technologie in die Codemeter Protection Suite von Wibu-Systems integriert. Außerdem hatte das Unternehmen die Idee, Blurry Box im Rahmen des Hackers Contest 2017 von echten Profis auf Herz und Nieren testen zu lassen. Als Beispielanwendung für den Contest wurde ein Reisespiel mit Fragen zu deutschen Städten ausgewählt. Den 315 Teilnehmern wurde kostenfrei das CmDongle mit allen für die vollständige Ausführung des Spiels notwendigen Schlüsseln zur Verfügung gestellt. Am Ende des Contests wurden zwei Einsendungen registriert. Beide waren Replay-Attacken auf den CmDongle, die aber nicht zu einem ohne CmDongle korrekt lauffähigen Spiel führten.