

Komponentenidentifikation: Voraussetzung für IT-Sicherheit im Automobil

André Weimerskirch ¹, Christof Paar ^{1,2} und Marko Wolf ²

¹ escrypt GmbH, Bochum, Germany
email: {aweimerskirch, cpaar}@escrypt.com
<http://www.escrypt.com>

² Lehrstuhl Kommunikationssicherheit, Ruhr-Universität Bochum, Germany
email: {cpaar, mwolf}@crypto.rub.de
<http://www.crypto.rub.de>

Zusammenfassung

Gefälschte Produkte, Produkte minderer Qualität und gestohlene Komponenten sind ein ernstzunehmendes Problem der Automobilindustrie. Der weltweite ökonomische Schaden aller Branchen wird auf 250 Milliarden USD geschätzt. Es ist möglich, kryptographische Methoden zu nutzen, um resultierende Gefahren deutlich zu verringern und den wirtschaftlichen Schaden zu minimieren. In dieser Arbeit wird ein automatischer Mechanismus zur Vermeidung von Diebstahl, zur Vermeidung von gefälschten Produkten und Produkten minderer Qualität sowie dem Schutz vor geklonten Teilen vorgestellt. Mit Hilfe eines in alle Komponenten integrierten Mikrochips wird beim Ein- und Ausbau einer Komponente als auch während des Betriebs des Autos eine kryptographische Prüfung vorgenommen. Nur wenn sich alle Komponenten korrekt identifizieren, ist ein ordnungsgemäßer Betrieb des Autos gewährleistet.

1 Einführung

Der Einsatz gefälschter Ersatzteile in Automobilen ist ein ernstzunehmendes Problem für die Hersteller und Zulieferer. Ersatzteile minderer Qualität werden als Originalteile angeboten, so dass ein enormer wirtschaftlicher Schaden entsteht. Eine Gefahr entsteht, wenn Komponenten, die für ein sicheres und zuverlässiges Auto notwendig sind, gefälscht wurden oder im Auto überhaupt nicht vorhanden sind. Zum Beispiel entsteht eine echte Gefahr, wenn der Austauschairbag nicht korrekt funktioniert, oder überhaupt nicht in das Auto eingebaut wurde. Ein weiteres Problem entsteht durch die Manipulation von Komponenten. Als gängiges Beispiel geben wir die Manipulation des Kilometerstands an. Für einige 100 Euro bieten diverse Anbieter im Internet eine Manipulation an. Ein anderes Beispiel ist die Manipulation der Motorelektronik, um die Leistung des Motors zu steigern.

In dieser Arbeit zeigen wir Möglichkeiten zur Identifikation von Komponenten auf. Diese Identifikation ermöglicht den Schutz gegen Diebstahl und gefälschte Komponenten. Unsere Lösung ermöglicht die Entdeckung manipulierter Komponenten, so dass das Auto eine Alarmmeldung ausgibt oder nicht startet. Als Beispiel geben wir ein elektronisches Kennzeichen an. Dieses könnte in Zukunft das herkömmliche Kennzeichen ablösen. Das elektronische Kennzeichen sendet eine eindeutige Kennung des Autos aus, die kryptographisch abgesichert ist. Es ist jedoch klar, dass die Komponente, die das elektronische Kennzeichen beinhaltet, gegen Manipulation geschützt sein muss. Der Ausbau dieser Komponente oder das Stehlen aus einem anderen Auto darf nicht möglich sein bzw. muss dadurch unterbunden werden, dass das Auto daraufhin nicht mehr startet. Unsere Lösungen sind in der Lage, genau dies zu erreichen. Sie sind einfach in ihrer Anwendung und laufen automatisch ab.

Nach unserem besten Wissen gibt es bisher kein System, das Komponentenidentifikation mit Hilfe kryptographischer Methoden umsetzt. Traditionelle Methoden benutzen holographische Aufkleber, die als fälschungssicher gelten. Jedoch sind immer wieder Beispiele in der Flugzeugindustrie bekannt geworden, bei denen diese Aufkleber auf dem Schwarzmarkt gekauft werden konnten [Bus96]. Weitere Verfahren, die mit eindeutigen Siegeln und Stickern arbeiten, können in [Ver04] gefunden werden. Eine einfache Methode, Diebstahlschutz zu gewährleisten, ist das Verbauen von Komponenten mit besonderen mechanischen Einbausperren. Zum Beispiel werden Autoradios verbaut, die nur in spezielle Einbauschächte passen. Hier entstehen jedoch Probleme beim Verkauf des Radios. Komponentenidentifikation wurde bereits von einigen Automobilherstellern betrachtet [Dai01, Vol01]. Hier wird jedoch keine Kryptographie erwähnt, und es wird davon ausgegangen, dass es ein vertrauensvolles Werkstattnetz gibt, über das Komponenten ausgetauscht werden können. Es ist jedoch anzunehmen, dass es immer einige Mechaniker gibt, die nicht vertrauenswürdig sind und das Sicherheitssystem somit umgehen.

Im Folgenden präsentieren wir eine Lösung zur Komponentenidentifikation. Unsere Lösung ermöglicht Diebstahlschutz, Schutz vor gefälschten Komponenten sowie Schutz vor Manipulation. Falls eine Onlineverbindung vorhanden ist, können wir sogar den Schutz vor geklonten Komponenten sicherstellen. Alle Methoden können automatisch ausgeführt werden, und unser System funktioniert auch in einer nicht-vertrauensvollen Umgebung. Wir beginnen mit einer Lösung, die ein zentrales Hochsicherheitsmodul (HSM) in jedem Auto erfordert. Ein HSM ist ein spezielles Sicherheitsmodul, das kryptographische Operationen durchführt und geheimes Schlüsselmaterial speichert, und gegen Manipulation geschützt ist. Beim Versuch, geheime Daten aus dem HSM auszulesen, wird dieses mit den Daten zerstört. In einem weiteren Protokoll verteilen wir die Rolle des HSM auf alle Komponenten, so dass es keinen zentralen Angriffs- und Fehlerpunkt mehr gibt. Weitere Details findet der interessierte Leser in [Hoe02, WHPW05].

2 Komponentenidentifikation

Wir beginnen mit einer Lösung, die ein zentrales HSM in jedem Auto vorsieht. Der Schutz vor gefälschten Teilen ermöglicht das Erkennen minderwertiger Teile, die nicht vom Originalhersteller produziert wurden. Diese Kategorie beinhaltet auch geklonte Komponenten. Wir definieren

dabei *Klonen* als das genaue Kopieren einer Komponente bis hin zu eventuellen kryptographischen Schlüsseln. Einen Schutz gegen Klonen können wir nur dann zur Verfügung stellen, falls jedes Auto in regelmäßigen Abständen eine Onlineverbindung zu einer zentralen Datenbank herstellt. Der Systemschutz sichert die Systemintegrität des Autos, indem alle Komponenten regelmäßig überprüft werden. Wir nehmen an, dass jede Komponente mit einem Mikrochip, im nachfolgenden *Komponentenchips* genannt, ausgestattet wird. Alle Komponentenchips werden mit einem kryptographischen Geheimnis ausgestattet, und können über den Systembus kommunizieren. Falls einzelne Komponenten nicht an den vorhandenen Systembus angeschlossen sind, ist auch eine Kommunikation über einen drahtlosen Kanal denkbar, z.B. mit Hilfe von Bluetooth. Im Nachfolgenden gehen wir davon aus, dass die Komponentenchips alle Aufgaben bezüglich Komponentenidentifikation für die Komponente übernehmen. Im einfachsten Fall gibt es einen zentralen Prüfer, in unserem Fall das HSM, das alle Komponenten nacheinander überprüft. Dabei sind drei Phasen zu betrachten: (1) Die Installation einer neuen Komponente, (2) das Auto im Betriebszustand, und (3) der Ausbau einer Komponente. Wir beschreiben unser System nun an einem Beispiel. Nur Originalteile können in ein Auto eingebaut werden. Jedes Mal, wenn das Auto gestartet wird, werden alle im Auto verbauten Komponenten auf Manipulation hin überprüft, und nur im Falle eines fehlerfreien Ergebnisses startet der Motor. Ein Display im Auto zeigt den Status aller vorhandenen Komponenten an. Dies ist z.B. sinnvoll, um zu überprüfen, ob eine Werkstatt tatsächlich eine alte Komponente gegen eine neuwertige ausgetauscht hat. Weiterhin kann die Systemprüfung manuell gestartet werden, z.B. um einem Polizisten zu beweisen, dass tatsächlich ein gültiges elektronisches Kennzeichen eingebaut ist. Wir machen folgende Annahmen:

1. In jedem Auto befindet sich ein HSM, das gegen Manipulation geschützt ist.
2. Ein einfacher Mikrochip (Komponentenchip) wird so in jede Komponente eingelassen, dass das Entfernen des Chips die Komponente und den Chip zerstört.
3. Das HSM sowie der Komponentenchip sind in der Lage, kryptographische Operationen wie digitale Signaturen auszuführen.
4. Es gibt eine temporäre Verbindung zwischen dem Auto und einem zentralen Server.

Später werden wir unsere Lösung so erweitern, dass Annahme (1) nicht mehr notwendig ist. Annahme (4) kann weggelassen werden, wobei dann jedoch kein Schutz gegen geklonte Komponenten mehr möglich ist. Im Folgenden benutzen wir digitale Signaturen und Zertifikate. Dabei hat jede Komponente einen öffentlichen Schlüssel PK als Teil des Zertifikats sowie den zugehörigen geheimen Schlüssel SK . Die digitale Signatur einer Nachricht m kann nur mit Kenntnis von SK erzeugt werden, und von jeder anderen Komponente mit Hilfe von PK überprüft werden. Die Erzeugung einer digitalen Signatur von m schreiben wir kurz als $S := \text{Sig}(m, SK)$, und die Verifikation als $\text{Ver}(S, PK)$. Eine Verifikation liefert entweder den Wert *gültig* oder *ungültig*. Wir erweitern unsere Annahmen nun wie folgt:

- Es gibt eine vertrauensvolle Autorität, die Zertifikate ausstellt. Diese Institution kann z.B. staatlich sein, oder ein Zusammenschluss aller Automobilhersteller.
- Das HSM führt kryptographische Operationen aus und speichert geheime Schlüssel. Weiterhin ist das HSM in der Lage, bei einem Sicherheitsverstoß selbständig einen Alarm auszulösen und das Auto stillzulegen. Zum Beispiel ist es vorstellbar, dass das HSM in den Bordcomputer integriert ist und einen speziellen kryptographischen Schlüssel mit dem Motor teilt. Dann ist das HSM in der Lage, dem Motor eine kryptographisch gesicherte Nachricht zu senden, so dass dieser nicht mehr anspringt.
- Jede Komponente ist mit einem Komponentenchip ausgestattet. Dieser Chip kann z.B. ein Smart-Karten Chip sein, der in der Lage ist, kryptographische Operationen auszuführen. Es ist vorteilhaft, den Chip so an die Komponente zu binden, dass das Herauslösen oder die Manipulation am Chip bzw. der Komponente den Chip sowie die Komponente zerstört. Der Komponentenchip könnte z.B. nach dem Gießen des Motors in diesen eingebettet werden.
- Jede Komponente erhält ein Zertifikat $\langle PK, ID \rangle$ sowie den zugehörigen geheimen Schlüssel SK . Der Wert ID beinhaltet dabei einen eindeutigen Namen sowie weitere Informationen wie z.B. das Herstellungsdatum.
- Das HSM eines Autos speichert eine Liste \mathcal{UL} aller Komponenten, die im Auto eingebaut sind. Diese Liste wird regelmäßig mit einer globalen Liste \mathcal{GL} aller Komponenten synchronisiert. Die Synchronisierung ist dabei kryptographisch abgesichert, und kann nicht durch eine dritte Partei manipuliert werden. Jede Komponente kann in der globalen Liste eindeutig anhand von ID identifiziert werden.
- Es ist möglich, eine globale Liste gesperrter Zertifikate (\mathcal{CRL}) zu benutzen. Ein Zertifikat wird zurückgezogen, wenn die zugehörige Komponente gestohlen oder geklont wurde.
- Wenn eine Überprüfung innerhalb des Protokolls fehlschlägt, wird ein Alarm ausgelöst.

Es ist bekannt, dass Geräte wie ein HSM, die gegen Manipulation geschützt sind, sehr kostspielig sind. Wir glauben jedoch, dass aufgrund des hohen Preises eines Autos ein HSM möglich ist, zumindest in der Luxusklasse. Somit ist der Aufwand, der betrieben werden muss um solch ein Gerät erfolgreich zu brechen, extrem hoch, so dass sich ein solcher Angriff nicht lohnt. Wir stellen später noch eine Lösung vor, die ohne HSM auskommt. Für die Komponentenchips erwarten wir, dass sie mit Hilfe von Techniken des Maschinenbaus in solch einer Art in die Komponenten eingebettet werden können, dass der Versuch, sie zu entfernen in der Zerstörung der Komponente und des Chips endet.

Alle Nachweise über die Kenntnis geheimer Schlüssel werden durch *Challenge-Response* Protokolle mit Hilfe digitaler Signaturen ausgeführt wie in Algorithmus 1 gezeigt. Dies kann in ähnlicher Weise mit Hilfe eines message authentication code (MAC) geschehen. Ein MAC

funktioniert dabei ähnlich wie eine digitale Signatur, wobei jedoch A und B denselben geheimen Schlüssel nutzen, um Nachrichten zu authentifizieren. Alle Nachrichten, die zwischen Komponenten ausgetauscht werden, sind verschlüsselt und gegen Manipulation geschützt.

Algorithmus 1 Challenge-Response Beweis

- 1: $B \rightarrow A : r_B$
 - 2: $A \rightarrow B : r_A, S := \text{SIG}(r_A || r_B || B, SK)$
 - 3: B prüft ob $\text{VER}(S, PK) \stackrel{?}{=} \text{'gültig'}$
-

Wie schon erwähnt, umfasst der Lebenslauf einer Komponente drei Phasen, die wir im Detail beschreiben. Zu Beginn gibt es eine Komponente, z.B. das HSM, das zufällig einen geheimen Schlüssel K wählt. Jede Komponente ist mit einem Zertifikat ausgestattet. Wenn eine Komponente zu einem Auto hinzugefügt wird, dann wird das Zertifikat dieser Komponente geprüft. Nach einem erfolgreichen Prüfvorgang erhält die neue Komponente den geheimen Schlüssel K , der später im laufenden Betrieb des Autos geprüft wird, um Manipulationen nach dem Einbau zu verhindern. Letztlich erlaubt unser Protokoll den kontrollierten Ausbau einer Komponente, um diese von einer gestohlenen unterscheiden zu können. Die drei Phasen, die wir im Nachfolgenden besprechen, sind folgende:

1. *Einbau einer Komponente*
2. *Systemüberprüfung*
3. *Ausbau einer Komponente*

2.1 Einbau einer Komponente

Eine neue Komponente wird zu einer Menge schon vorhandener Komponenten hinzugefügt. Sobald eine neue Komponente hinzugefügt wird, wird die Einbauphase durchlaufen, die aus den nachfolgenden Schritten besteht:

- *Schlüsselüberprüfung*: Zuerst wird überprüft, ob eine Komponente schon Teil dieses Autos war, z.B. weil sie für eine Reparatur ausgebaut wurde.
- *Echtheitsnachweis*: Falls die Komponente noch nicht Teil des Autos ist, wird ihr Zertifikat überprüft.
- *Schlüsselinitialisierung*: Nach erfolgreichem Echtheitsnachweis wird die neue Komponente mit dem Systemschlüssel K ausgestattet.

Die Schlüsselüberprüfung läuft wie in Abbildung 1 gezeigt ab. Zuerst prüft das HSM für eine neue Komponente C mit der Kennung ID , ob $ID \in \mathcal{UL}$. Ist dies der Fall, überprüft das HSM mit Hilfe von Algorithmus 1, ob C den geheimen Schlüssel K kennt. Falls C neu ist, läuft der Echtheitsnachweis ab.

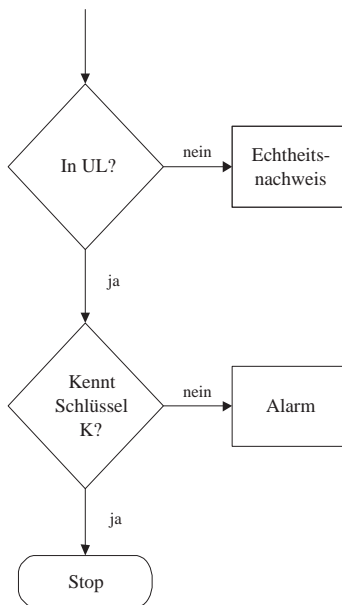


Abbildung 1: Schlüsselüberprüfung einer neuen Komponente

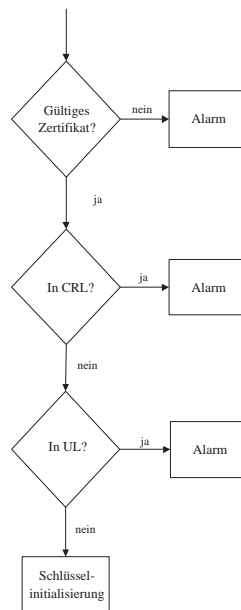


Abbildung 2: Echtheitsnachweis

Während des Echtheitsnachweises prüft das HSM, ob C ein gültiges Zertifikat vorweist. Dabei geht das HSM wie in Algorithmus 2 und Abbildung 2 gezeigt vor. Im letzten Schritt wird C vorläufig auf die Liste \mathcal{UL} gesetzt. Wenn C schon auf der globalen Liste \mathcal{GL} steht, so bedeutet dies, dass C zur Zeit in einem anderen Auto eingebaut ist und somit entweder gestohlen oder geklont wurde.

Algorithmus 2 Echtheitsnachweis

- 1: Das HSM verifiziert das Zertifikat $\langle PK, ID \rangle$ und prüft, ob C den geheimen Schlüssel SK kennt mit Hilfe von Algorithmus 1.
 - 2: Dann prüft das HSM, ob $\langle PK, ID \rangle$ auf der Liste ungültiger Zertifikate \mathcal{CRL} steht.
 - 3: Das HSM stellt C auf die Liste aller eingebauten Komponenten \mathcal{UL} , wobei die neue Komponente vorläufig markiert wird.
 - 4: Nachdem \mathcal{UL} mit der globalen Liste aller verbauten Komponenten \mathcal{GL} synchronisiert wurde, wird die vorläufige Markierung von C gelöscht. Wenn C schon auf \mathcal{GL} stand, wird ein Alarm ausgelöst.
-

Nachdem alle Tests erfolgreich abgelaufen sind, wird C mit dem Schlüssel K initialisiert. Dazu sendet das HSM den Wert $E(K, PK)$ an C , der die Verschlüsselung des Systemschlüssels K mit dem öffentlichen Schlüssel PK von C ist. C kann nun mit Hilfe des geheimen Schlüssels SK , den nur C kennt, K entschlüsseln.

2.2 Systemüberprüfung

Im vorigen Abschnitt haben wir sichergestellt, dass alle Komponenten, die in ein Auto eingebaut werden, geprüft wurden. Nach dem Einbau könnten diese Komponenten jedoch manipuliert, ausgetauscht oder entfernt werden. Daher führen wir zur Laufzeit des Autos die

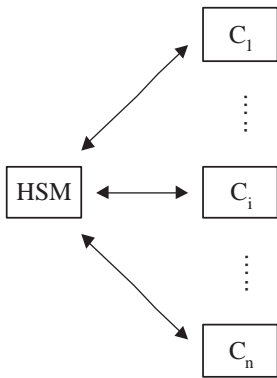


Abbildung 3: Systemüberprüfung durch das HSM

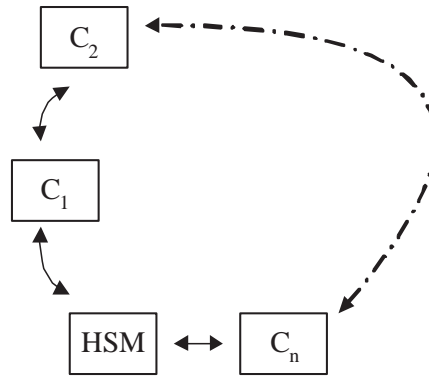


Abbildung 4: Systemüberprüfung mit zyklischer Verifikation

Systemprüfung durch, um zu überprüfen, ob alle eingebauten Komponenten noch im Besitz des Systemschlüssels K sind. Die Systemprüfung kann dabei zu verschiedenen Zeitpunkten ausgeführt werden:

- bei jedem Start des Autos
- periodisch, z.B. alle 30 Minuten
- durch den Benutzer, z.B. um die Korrektheit gegenüber einem Dritten zu beweisen

Wir stellen hier zwei Möglichkeiten vor, die Systemüberprüfung durchzuführen. Die erste Ausführung ist in Abbildung 3 dargestellt. Hier überprüft das HSM alle eingebauten Komponenten der Liste \mathcal{UL} mit Hilfe eines Challenge-Response Tests.

Die zweite Ausführung ist in Abbildung 4 dargestellt. Das HSM überprüft eine Komponente, die wiederum eine Komponente prüft, und so weiter, so dass sich ein Kreis bildet. Jede Komponente muss dazu die Liste \mathcal{UL} aller eingebauten Komponenten speichern, und alle Komponenten müssen sich auf eine Ordnung einigen, so dass jede Komponente einmal prüft und einmal überprüft wird. Wenn eine Überprüfung fehlschlägt, sendet die prüfende Komponente eine Warnmeldung an das HSM, das dann die entsprechenden Maßnahmen ergreift. Es ist leicht, diese Alarmmeldung einer Komponente an das HSM abzufangen. Daher verlangen wir, dass jede Komponente eine positive oder negative Ergebnismeldung an das HSM sendet, die mit Hilfe des Systemschlüssels K abgesichert ist. Die zyklische Systemüberprüfung kann von allen Komponenten parallel ausgeführt werden, wenn der Systembus dies unterstützt.

2.3 Ausbau einer Komponente

Die Ausbauprozedur wird aufgerufen, sobald eine Komponente aus einem Auto entfernt wird, um sie in ein anderes Auto wieder einzubauen. Die Prozedur ist notwendig, um eine korrekt ausgebaute Komponente von einer gestohlenen Komponente zu unterscheiden. Der Ausbau

einer Komponente C funktioniert dabei wie in Algorithmus 3 gezeigt. Um C auf kontrollierte Art und Weise zu entfernen, wird zuerst geprüft, ob die Komponente zum System gehört, indem der Schlüssel K geprüft wird. Dann wird C von der Liste \mathcal{UL} und \mathcal{GL} gelöscht, so dass C wie eine unverbaute Komponente erscheint. Spezielle Komponenten, die an ein gegebenes Auto gebunden sind und daher nicht ausgebaut werden dürfen, können mit einer speziellen Markierung in ihrem Zertifikat und den Listen \mathcal{UL} und \mathcal{GL} versehen werden. Das HSM wird dann den Ausbau dieser Komponenten, z.B. eines elektronischen Kennzeichens, durch eine Alarmmeldung anzeigen.

Algorithmus 3 Ausbau einer Komponente

- 1: Das HSM prüft, ob C den Systemschlüssel K kennt.
 - 2: Dann löscht das HSM C von der Liste \mathcal{UL} .
 - 3: Bei der nächsten Synchronisierung von \mathcal{UL} wird C auch auf der globalen Liste \mathcal{GL} gelöscht.
-

Falls es keine globale Liste \mathcal{GL} gibt oder \mathcal{UL} nur selten mit \mathcal{GL} synchronisiert wird, kann die Komponente selber speichern, ob sie gerade verbaut ist oder nicht. Die Komponente kann dann nur in ein neues Auto eingebaut werden, wenn sie den entsprechenden Zustand gespeichert hat, der jeweils beim Ein- und Ausbau gesetzt wird.

2.4 Dezentrale Sicherheit

Bei unserem bisherigen Ansatz sind wir davon ausgegangen, dass in jedem Auto ein HSM zur Verfügung steht. Das HSM ist jedoch ein zentraler Fehlerpunkt, und somit auch Manipulation und anderen Angriffen ausgesetzt. Im Nachfolgenden schlagen wir Mechanismen vor, um die Aufgabe des HSM an die Komponentenchips aller Komponenten aufzuteilen. Dieser Ansatz ist günstiger, da er die Kosten für das HSM spart, und sicherer, da es keinen zentralen Angriffspunkt mehr gibt. Wie zuvor wird in jede Komponente ein Komponentenchip eingebettet, der in der Lage ist, kryptographische Operationen durchzuführen. Wir beginnen mit der Definition einiger Hilfsprozeduren.

Verteilung von \mathcal{UL} : Bisher hat das HSM die Liste \mathcal{UL} aller im Auto verbauten Komponenten gespeichert und gepflegt. Diese Liste wird nun an alle eingebauten Komponenten verteilt und von diesen gepflegt. Ändert eine Komponente die Liste, so wird Algorithmus 4 zur Verteilung der Liste an alle Komponenten ausgeführt. Hierbei beschreibt $MAC(m, k)$ den MAC einer Nachricht m über den Schlüssel k . Der Zähler i verhindert dabei das wiederholte Senden alter Listen (replay-attack).

Algorithmus 4 Verteilung der Liste

- 1: Eine Komponente C erhöht den Zähler i und berechnet
 $U := [E := E(\mathcal{UL}||i, K), M := MAC(E, K)]$.
 C sendet U an alle anderen Komponentenchips.
 Ein Empfänger R nimmt U nur an, wenn M gültig ist und der Zähler i größer als sein gespeicherter Wert i' ist.
 R speichert \mathcal{UL} und aktualisiert $i' \leftarrow i$.
-

Auswahl einer zufälligen Komponente: Wir benötigen zusätzlich die Möglichkeit, eine zufällige Komponente im Auto auszuwählen. Dazu müssen zuerst alle Komponenten zusammen eine Zufallszahl x wählen, wie in Algorithmus 5 gezeigt.

Algorithmus 5 Auswahl einer Zufallszahl

- 1: Jede Komponente C_i wählt eine Zufallszahl r_i und sendet $E := E(r_i || ID_i, K)$ sowie $M := MAC(E, K)$ an alle anderen Komponenten.
Jede Komponente prüft, ob sie eine Eingabe von allen anderen Komponenten erhalten hat und prüft die Korrektheit dieser Eingaben.
Jede Komponente berechnet $x := h(\sum_i r_i)$, wobei h eine Hashfunktion ist.
-

Dieses Schema ist anfällig für einen Angriff durch eine Komponente, die den Schlüssel K kennt, jedoch nicht gegen einen Angriff von außen. Mit Hilfe von Commitment Verfahren, auf die wir hier jedoch nicht weiter eingehen, kann die Auswahl jedoch beliebig sicher gestaltet werden. Nun können alle Komponenten zusammen eine zufällige Komponente auswählen, wie in Algorithmus 6 aufgezeigt.

Algorithmus 6 Auswahl einer zufälligen Komponente

- 1: Alle Komponenten wählen eine zufällige Zahl x aus.
 - 2: Jede Komponente C_i berechnet $v_i := h(ID_i || x)$, wobei h eine Hashfunktion ist.
 - 3: Die Komponente C_i mit dem kleinsten Wert v_i agiert als Prüfer. Da die ID's durch die Liste \mathcal{UL} bekannt sind, kann jede Komponente das Ergebnis nachvollziehen.
-

Nun wissen alle, wer als Prüfer für einen Einbau oder Ausbau einer Komponente agiert und nehmen eine Aktualisierung von \mathcal{UL} durch die ausgewählte Komponente an.

Alarmmeldung: Letztlich muss es einen Mechanismus geben, um einen Alarm auszulösen. Dazu schickt jede Komponente, die eine Prüfung durchführt, ein positives bzw. negatives Resultat an alle anderen Komponenten. Wenn das Ergebnis negativ war, kann jede Komponente Maßnahmen ergreifen. Im einfachsten Fall stellt eine Komponente ihre Arbeit ein. Um das Blockieren der Kommunikationskanäle durch einen Angreifer zu unterbinden, können z.B. Zeitlimits eingeführt werden. Ist das Zeitlimit für eine Ergebnismeldung abgelaufen, gehen die Komponenten von einer Negativmeldung aus.

Nachdem wir die erforderlichen Hilfsprotokolle vorgestellt haben, betrachten wir nun noch einmal den Lebenszyklus einer Komponente und beschreiben die Vorgänge in dem Fall ohne HSM. Diese folgen in einfacher Art, indem die Phasen Einbau (Abschnitt 2.1), Laufzeit (Abschnitt 2.2) und Ausbau (Abschnitt 2.3) mit obigen Hilfsprotokollen verknüpft werden.

Einbau: Für den Einbau einer Komponente C muss eine Komponente S anstelle des HSM als Prüfer agieren. Diese Komponente S wird zufällig mit Hilfe von Algorithmus 6 ausgewählt. Nun führt S die Einbauroutine durch, aktualisiert \mathcal{UL} und sendet diese Liste an alle anderen Komponenten wie in Algorithmus 4 beschrieben.

Laufzeit: Für die Systemprüfung zur Laufzeit wird wieder eine zufällige Komponente ausgesucht, um die Prüfung zu starten. Wenn eine Prüfung fehlschlägt, wird ein verteilter Alarm ausgelöst.

Ausbau: Für den Ausbau einer Komponente C wird eine zufällige Komponente S als Verifizierer bestimmt. S führt die Ausbauprozedur durch, aktualisiert \mathcal{UL} und verteilt die neue Liste.

3 Erweiterungen

Im Folgenden stellen wir einige Erweiterungen vor, die direkt aus unserem System folgen.

Sicherheitsregeln: Jedes Komponentenzertifikat könnte eine Beschreibung über die Eigenschaften der Komponente enthalten. Diese Information kann z.B. eine Markierung enthalten, ob diese Komponente aus einem Auto ausgebaut werden darf oder ob sie nur ausgebaut werden darf, wenn ein Ersatz eingebaut wird. Weiterhin könnte diese Information festlegen, wie andere Komponenten auf das Fehlen bzw. fehlerhaftes Verhalten dieser Komponente reagieren. Beispielsweise könnten alle Komponenten ihren Dienst verweigern, wenn der Airbag fehlt, oder sie könnten eine Warnmeldung anzeigen, wenn das Autoradio fehlt.

Elektronische Komponenten: Wenn die Komponente von elektronischer Natur ist wie z.B. ein Mikrocontroller, dann kann die Aufgabe des Komponentenchips durch die Komponente wahrgenommen werden. Ein Beispiel ist ein Kilometerzähler. Dieser wird häufig manipuliert, um den Wiederverkaufswert eines Autos zu steigern. Diese Manipulation kann verhindert werden, indem der Zugang zum Kilometerstand auf Komponenten begrenzt wird, die den Schlüssel K kennen. Weiterhin könnte der Kilometerstand mit Hilfe von K verschlüsselt und gegen Manipulation gesichert werden. Ein Schutz gegen das Auslesen von K kann hier mit Hilfe von Trusted Computing Mechanismen ermöglicht werden.

Komponentenhistorie: Diese Erweiterung ermöglicht die Unterscheidung von neuen zu vorher verbauten Komponenten. Wenn eine globale Liste \mathcal{GL} vorliegt, ist dies einfach durch Analyse dieser Liste möglich. Andernfalls erhält jede Komponente zusätzlich zu ihrem normalen Zertifikat \mathcal{Z} ein weiteres Zertifikat $\langle PK_{new}, \mathcal{Z} \rangle$ und den zugehörigen geheimen Schlüssel SK_{new} . Wenn eine Komponente das erste Mal verbaut wird, muss sie die Kenntnis von SK_{new} nachweisen. Nach erfolgreichem Einbau löscht die Komponente SK_{new} , so dass sie ihn beim erneuten Einbau nicht mehr nachweisen kann.

Schlüsselinitialisierung: Die Schlüsselinitialisierung zur Verteilung des Systemschlüssels K kann auf zwei Arten erfolgen. Die erste Möglichkeit ist die oben beschriebene, wobei der Schlüssel durch eine einzelne Komponente ausgewählt wird, die das System begründet. Weitere Komponenten werden dann nacheinander zu dem System hinzugefügt wie oben beschrieben. Alternativ kann das Auto zuerst vollständig gefertigt werden. Danach wird eine Smart-Karte mit dem Auto verbunden, um die Initialisierung zu starten. Die Smart-Karte wählt einen zufälligen Systemschlüssel K aus und verteilt diesen Schlüssel an alle vorhandenen Komponenten. Zu jedem Auto gehört somit eine individuelle Smart-Karte. Alternativ kann das HSM oder eine designierte Komponente den Schlüssel K erzeugen und diesen an alle vorhandenen Komponenten verteilen.

Schlüsselhierarchien: Im Automobilbereich gibt es verschiedene Gruppen mit unterschiedlichen Interessen. Es gibt den Autobesitzer, die Werkstattmechaniker von unabhängigen oder

autorisierten Vertragspartnern, die Komponentenhersteller und die Automobilhersteller. Diese Gruppen mit verschiedenen Autorisationsstufen können durch die Benutzung unterschiedlicher Schlüssel implementiert werden. Zum Beispiel könnte jeder Autobesitzer mit einer Smart-Karte versorgt werden, die ihm den Aus- und Einbau nicht sicherheitsrelevanter Komponenten ermöglicht. Wenn sicherheitsrelevante Komponenten ausgetauscht oder eingebaut werden, müssen dagegen Smart-Karten vom Besitzer und einer Werkstatt vorliegen. Offensichtlich darf es dabei jedoch keine globalen Smart-Karten geben, die beliebige Manipulationen an allen Autos ermöglichen. Es ist auch klar, dass die Sicherheitsmechanismen des Autos nicht durch den Einsatz einer Smart-Karte ausgehebelt werden können, d.h. alle Sicherheitstests werden weiterhin durch das HSM bzw. die anderen Autokomponenten durchgeführt.

Untergruppen: Es ist möglich, die Komponenten in Untergruppen aufzuteilen. Diese Untergruppen können z.B. nach verschiedenen Sicherheitsleveln aufgeteilt sein. Jede Untergruppe erhält einen eigenen Schlüssel K_u . Je höher der Sicherheitsbedarf einer Untergruppe ist, desto besser könnten die Komponentenchips und die Komponenten gegen Manipulation abgesichert sein. Wenn ein Schlüssel kompromittiert wird, ist nur diese Untergruppe betroffen.

4 Sicherheit

Wir haben die Sicherheit unseres Systems auf den Manipulationsschutz des HSM sowie der Komponentenchips basiert. Wenn diese Annahme hält, ist unser System offensichtlich ausreichend vor Angriffen geschützt, da alle involvierten Parteien, die den Schlüssel K kennen, korrekt agieren. Was passiert jedoch, wenn ein Schlüssel kompromittiert wird?

Wenn ein Angreifer Mallory den Systemschlüssel K kompromittiert, könnte er gefälschte Komponenten mit K ausstatten und diese in das Auto einbauen. Diese Attacke funktioniert so jedoch nicht, da Komponenten zuerst ein gültiges Zertifikat nachweisen müssen, bevor sie vom System angenommen werden, d.h. die Attacke funktioniert nur, wenn die einzubauenden Komponenten schon auf der Liste \mathcal{UL} stehen. Somit muss Mallory alle Komponenten manipulieren, die \mathcal{UL} gespeichert haben, so dass diese Attacke aufwendig wird.

Wenn Mallory ein Zertifikat $\langle PK, ID \rangle$ sowie den zugehörigen geheimen Schlüssel SK kompromittieren kann, dann kann er gefälschte Komponenten mit diesem Zertifikat ausstatten. Er kann dann die gefälschten Komponenten erfolgreich installieren und erhält den Systemschlüssel K . Jedoch kann Mallory auf diese Art nur eine gefälschte Komponente pro Auto einbauen, da das HSM den Einbau mehrerer gleicher Komponenten in demselben Auto entdecken würde. Dieser Angriff ist dem Klonen ähnlich. Daher kann es nur effektiv verhindert werden, wenn es eine globale Liste \mathcal{GL} aller verbauten Komponenten gibt wie vorher schon beschrieben.

Mallory kann den öffentlichen Schlüssel des Zertifikatsausstellers PK_{CA} mit seinem eigenen öffentlichen Schlüssel PK_M ersetzen. Dann ist er in der Lage, Komponenten mit eigenen Zertifikaten auszustatten, die alle Tests erfolgreich bestehen. Mallory muss dazu jedoch auch den öffentlichen Schlüssel im HSM ersetzen. In einer dezentralen Lösung muss Mallory sogar den öffentlichen Schlüssel mehrerer Komponentenchips ersetzen, da Mallory nicht beeinflussen kann, welche Komponente als Verifizierer agiert.

5 Schlussfolgerung

Wir haben hier Methoden zur Umsetzung der Komponentenidentifikation im Automobil vorgestellt, die auf kryptographischen Authentifizierungsmechanismen beruhen. Unser erster Ansatz basiert auf einem zentralen Sicherheitsmodul HSM. Danach haben wir die Aufgabe des HSM dezentral auf alle Komponenten verteilt. Unser System ermöglicht die Diebstahlerkennung von Komponenten sowie die Erkennung gefälschter Komponenten und sichert das Auto vor Manipulation. Das System arbeitet automatisch im Hintergrund ohne das Zutun des Autobesitzers, und kann auch in Regionen eingesetzt werden, in denen kein vertrauensvolles Werkstattnetz vorhanden ist. Unser System ist zudem flexibel und erlaubt vielfältige Erweiterungen wie die Einführung von Schlüsselhierarchien und Sicherheitsregeln.

Unsere Ansätze basieren hauptsächlich auf dem Manipulationsschutz des HSM und der Komponentenchips. Es muss noch untersucht werden, inwieweit ein kostengünstiges HSM gebaut werden kann, dass den speziellen Anforderungen des Automobilbereichs genügt. Weiterhin ist es erforderlich zu untersuchen, wie Komponentenchips, z.B. Smart-Karten, in eine Komponente integriert werden können. Im Falle von elektronischen Komponenten wie z.B. dem Kilometerzähler und dem gesamten Armaturenbrett glauben wir, dass wir mit Hilfe des Trusted Computing die erforderliche physikalische Sicherheit herstellen können. Da immer mehr und immer wichtigere Komponenten im Automobil von elektronischer Natur sind, erwarten wir dass Trusted Computing eine kostengünstige Möglichkeit zur Umsetzung unseres Systems bereitstellt.

Literatur

- [Bus96] Business Week, W. Stern. Warning! Bogus parts have turned up commercial jets. Where's the FAA?, June 1996.
- [Dai01] Daimler Chrysler. Ausbausicherung für elektronische Komponenten bei einem Fahrzeug. Offenbarungsschrift, DE 100 21 811 A1, November 2001.
- [Hoe02] K. Hoepfer. Cryptographic protocols for component identification and applications. Diplomarbeit, September 2002. Ruhr-Universität Bochum, Communication Security Group.
- [Ver04] Verpackungsrundschau. Webpage, 2004. <http://www.verpackungsrundschau.de>.
- [Vol01] Volkswagen AG. Kraftfahrzeug mit einer Vielzahl von Bauteilen. Offenbarungsschrift, DE 100 01 986 A1, July 2001.
- [WHPW05] André Weimerskirch, Katrin Hoepfer, Christof Paar, and Marko Wolf. Component identification: Enabler for secure networks of complex systems. In *Proceedings of Applied Cryptography an Network Security 2005 (ACNS 2005)*, 2005.