Mikrokern für zulassungspflichtige Hochsicherheitssysteme

Thomas Günther[1] · Michael Hohmuth[2]
Adam Lackorzynski[2] · Matthias Lange[2]

[1]INFODAS GmbH
vertrieb@infodas.de

[2]Kernkonzept GmbH
info@kernkonzept.com

Zusammenfassung

Sichere Netzübergänge sind Systeme, die einen Datenaustausch zwischen Netzen ermöglichen, in denen staatliche Verschlusssachen mit unterschiedlichen, maximalen Geheimhaltungsgraden verarbeitet werden. Solche Systeme werden an der sensiblen Schnittstelle zweier unterschiedlicher Sicherheitsdomänen betrieben und müssen den Informationsfluss hinsichtlich des Geheimhaltungsgrad der zu übertragenden Informationen exakt überprüfen. Sichere Netzübergänge dürfen nur eingesetzt werden, wenn sie durch das Bundesamt für Sicherheit in der Informationstechnik (BSI) zugelassen wurden. Das BSI stellt erhebliche sicherheitstechnische Anforderungen an sichere Netzübergänge. Unter anderem ist eine so genannte „sichere Ablaufplattform" einzusetzen. Als Ablaufplattform wird die Kombination aus Hardware und Betriebssystem verstanden. Wesentliche Eigenschaften eines geeigneten sicheren Betriebssystems sind Evaluierbarkeit, Isolationsfunktionen, Hardware-Zugriffsschutz, Separierung von und vertrauenswürdige Kommunikation zwischen so genannten Compartments. Der Mikrokern L4Re ist die technologische Basis für das sichere Betriebssystem SDoT MOS, auf welchem der durch das BSI bis GEHEIM allgemein zugelassene sichere Netzübergang SDoT Security Gateway 6.0 läuft.

1 Problemstellung

1.1 Sichere Netzübergänge

Sichere Netzübergänge sind Systeme, die einen bidirektionalen Datenaustausch zwischen „roten" und „schwarzen" Sicherheitsdomänen ermöglichen. „Rote" Sicherheitsdomänen verarbeiten dabei regelmäßig staatliche Verschlusssachen, z.B. bis zu einem maximalen Geheimhaltungsgrad von GEHEIM. „Schwarze" Sicherheitsdomänen wiederum enthalten Daten, die z.B. NUR FÜR DEN DIENSTGEBRAUCH (VS-NfD) eingestuft sind.

Ein technisches System zum effektiven und ggfs. auch automatisierten Datenaustausch zwischen Domänen mit unterschiedlichen maximal verarbeiteten Geheimhaltungsgraden ist zulassungspflichtig. Eine Zulassung kann nur das BSI erteilen [Bund10]. Ein solches System darf die Übertragung von „rot" nach „schwarz" nur erlauben, wenn der Geheimhaltungsgrad der zu übertragenden Daten zweifelsfrei als niedrig genug für den „schwarzen" Bereich identifiziert werden konnte. Andernfalls muss eine Übertragung unbedingt vermieden werden.

P. Schartner · A. Baumann (Hrsg.) · D•A•CH Security 2017 · syssec (2017) 316-326.

Zur Entscheidung, ob eine Information von „rot“ nach „schwarz“ übertragen werden darf, kommen zwei grundsätzliche Verfahren infrage: Prüfung des Dokumenteninhaltes oder Prüfung eines so genannten Sicherheitslabels.

Beim ersten Verfahren (Prüfung des Dokumenteninhaltes) muss der sichere Netzübergang in die Lage versetzt werden, den Inhalt der Daten zu „verstehen“ und auf diesem Verständnis basierend einen Geheimhaltungsgrad festlegen. Anhand dieses Geheimhaltungsgrades kann dann eine Entscheidung getroffen werden, ob die Information übertragen werden darf.

Beim zweiten Verfahren (Prüfung eines Sicherheitslabels) ist kein inhaltliches Verständnis der zu übertragenden Daten durch den sicheren Netzübergang notwendig. Stattdessen wird eine zusätzliche Information den Daten mitgegeben. Diese zusätzliche Information besteht aus Metadaten und heißt Sicherheitslabel. Das Sicherheitslabel enthält unmittelbar die Angabe des Geheimhaltungsgrades der betreffenden Information und kann verwendet werden, um eine Entscheidung über die Übertragung zu treffen.

Beide Verfahren haben Vor- und Nachteile, die im Folgenden detaillierter dargestellt werden:

1.1.1 Regelwerksbasierte Prüfung

Um dem sicheren Netzübergang ein rudimentäres „Verständnis“ der Daten mitzugeben, kommen Regelwerke zum Einsatz. Bei der regelwerksbasierten Prüfung von Informationen vergleicht der sichere Netzübergang die zu übertragenden Daten mit vorgegebenen Schemata. Die Nutzung solcher Schemata kann verständlicherweise nur erfolgen, wenn Struktur und zu erwartender Inhalt der Daten im Vorfeld bekannt sind. Die regelwerksbasierte Prüfung kann daher auch nur für stark strukturierte Daten genutzt werden, wobei hier nicht die technische Struktur eines Datenformates gemeint ist, sondern, dass die Daten auch logisch einer gewissen Struktur folgen. Word- oder JPG-Dokumente beispielsweise besitzen natürlich jeweils eine technische Struktur, der wesentliche Teil der enthaltenen Informationen ist aus dieser Perspektive jedoch nicht strukturiert, sondern „Freitext“ oder beliebige Bildpunkte. Für solche Daten lässt sich kein Regelwerk erstellen, dass den Ansprüchen eines sicheren Netzübergangs zum Verständnis der Daten genügt. Es wäre zwar ein leichtes zu prüfen, ob eine von „rot“ nach „schwarz“ zu übertragende Information ein korrekt formatiertes Word- oder JPG-Dokument ist, Aussagen über den Inhalt sind jedoch ungleich schwieriger.

Existierende Systeme, die auch in Word- und JPG-Dokumenten Gemeinsamkeiten feststellen können und eine Analyse der Inhalte vornehmen können, sind für die Ansprüche des Geheimschutzes (noch) nicht geeignet. Diese Systeme arbeiten mit heuristischen Verfahren und haben üblicherweise anhand von Beispieldaten „gelernt“, wie bestimmte Dokumente aussehen und mit welchen Geheimhaltungsgraden sie assoziiert werden können. Eine statistische Aussage zum Geheimhaltungsgrad einer Information ist jedoch nicht ausreichend, um eine Information aus einem geheimen Netz in ein niedriger eingestuftes Netz zu übertragen. Dies ergibt sich aus den Regelungen im Umgang mit staatlichen Verschlusssachen. [Bund10]

Bei stark strukturierten Datentypen ist eine inhaltliche Untersuchung jedoch nicht auf stochastische Verfahren angewiesen. Hier kann neben der Struktur (also der Angabe darüber, welche Information an welcher Stelle in den Daten zu finden ist) auch der variable Anteil genau geprüft werden (nämlich, ob die Information an sich in einem erlaubten Wertebereich liegt). Für Texte lassen sich die Werte mit den Angaben auf einer Whitelist vergleichen. Für komplexe Datentypen sind reguläre Ausdrücke anwendbar.

Entscheidend für den sicheren Netzübergang und den Anforderungen des Geheimschutzes ist, dass die Prüfung quasi für jedes einzelne Byte einer zu übertragenden Information erfolgt, um auszuschließen, dass potenziell eingestufte Daten „zwischen“ den erlaubten Informationen versteckt sind. Sollte auch nur ein einziges Byte nicht den Vorgaben des Regelwerks entsprechen, so ist die gesamte Information abzuweisen.

Zusätzlich sind Sanitarisierungsmaßnahmen, also das Ersetzen von bestimmten Stellen innerhalb der Daten durch vorgegebene Werte, möglich und oft auch notwendig. Durch Sanitarisierung können Freitextanteile in einem vorgegebenen Datenformat mit vorgegebenen Werten überschrieben werden, sofern die erlaubten Wertebereiche im Vorfeld nicht festgelegt werden können. Somit können diese Datenformate trotzdem übertragen werden, ohne dass eingestufte Informationen abfließen können. Das funktioniert natürlich nur, wenn die sanitarisierten Stellen keine für die weitere Verarbeitung in der Zieldomäne relevanten Daten enthalten.

Im Gegensatz zu einer üblichen Firewall findet also nicht nur eine Prüfung dahingehend statt, ob ein bestimmtes Protokoll und ein bestimmtes Datenformat gesendet wird, sondern vor allem, ob die Daten inhaltlich in Ordnung sind.

Wenn die regelwerksbasierte Prüfung erfolgreich ist, wird der Datei der Geheimhaltungsgrad zugeordnet, der mit dem entsprechenden Regelwerk verbunden ist. Es ist möglich, dass ein sicherer Netzübergang mit verschiedenen Regelwerken konfiguriert wird und die Information erst final abgewiesen wird, wenn keines der Regelwerke erfolgreich war.

Der Vorteil einer regelwerksbasierten Prüfung ist, dass eine vollautomatische, schnelle Entscheidung durch den sicheren Netzübergang getroffen werden kann. Nachteilig wirkt sich aus, dass sie nur für stark strukturierte Daten anwendbar ist und die Struktur sowie die Wertebereiche der variablen Anteile im Vorfeld bekannt sein müssen, damit entsprechende Regelwerke erstellt werden können.

1.1.2 Label-basierte Prüfung

Die Nachteile der regelwerksbasierten Prüfung (Einschränkung der Datentypen, Kenntnis der variablen Anteile) gibt es bei der Label-basierten Prüfung nicht. Durch die in einer externen Datenstruktur (dem Sicherheitslabel) mitgelieferten Metadaten lässt sich der Geheimhaltungsgrad festlegen bzw. ablesen. Eine inhaltliche Prüfung muss daher nicht mehr erfolgen. Eine Einschränkung der Datenformate ist nicht notwendig.

Allerdings müssen die Angaben im Sicherheitslabel und die Daten, auf die es sich bezieht, integritätsgeschützt werden, um Manipulationen zu verhindern. Dieser Manipulationsschutz greift auf etablierte und geeignete kryptographische Verfahren zurück, auf die an dieser Stelle nicht weiter eingegangen werden soll. Entscheidend ist, dass die Implementierung im sicheren Netzübergang die Prüfung nachweislich korrekt durchführt. Außerdem muss das Sicherheitslabel erzeugt werden, idealerweise durch einen Dienst, der in der „roten“ Domäne berechtigten Nutzern zur Verfügung steht.

Die NATO Communications and Information Agency (NCI Agency) hat dazu in zwei Technical Notes ([OuLR12] und [OuLR13]) die Syntax und den Integritätsschutz der Sicherheitslabel beschrieben, wobei der Integritätsschutz hier „Metadata Binding“ genannt wird, um zum Ausdruck zu bringen, wie Informationsobjekt und Sicherheitslabel aneinander „gebunden“ werden.

Vorteil der Label-basierten Prüfung ist, dass alle denkbaren Daten- und Dokumententypen abgedeckt und mittels eines sicheren Netzübergangs geprüft und übertragen werden können. Der

Nachteil an der Label-basierten Prüfung ist, dass eine externe Instanz die Sicherheitslabel zunächst erstellen muss und dass es erhebliche Anforderungen an den Integritätsschutz (siehe oben, Stichwort „Metadata Binding“) und die damit verbundene Kryptographie gibt.

1.2 Zulassung

Systeme zur Trennung von Netzen mit unterschiedlichen maximalen Einstufungen sind durch das BSI zuzulassen [Bund10]. Für die Zulassung eines sicheren Netzübergangs durch das BSI sind eine Reihe von unterschiedlichen und ganz erheblichen sicherheitstechnischen Anforderungen zu erfüllen. Unter allen Umständen muss sichergestellt werden, dass keine eingestuften Daten aus dem „roten“ Bereich abfließen können.

Neben Anforderungen an die Hardware, die Filterkomponenten, die Kryptographie und die Robustheit gegen physische Angriffe und Manipulationen ist eine zentrale Forderung eine so genannte „sichere Ablaufplattform“.

2 Sichere Ablaufplattformen

Unter einer sicheren Ablaufplattform wird üblicherweise die Kombination aus Hardware und Betriebssystem verstanden, deren sicherheitstechnische Funktionsweise nachgewiesen wurde. Das nationale Schutzprofil (NPP, National Protection Profile) „Sichere Netzübergänge“ verlangt mit Blick auf die „Secure Software Platform“, dass die zu Anwendung kommende Plattform über starke Separierungs- und Isolierungsmechanismen sowie die Fähigkeit verfügen soll, die Kommunikation zwischen Prozessen zu überwachen und einzuschränken. (Das angesprochene Schutzprofil ist in VS-NUR FÜR DEN DIENSTGEBRAUCH eingestuft und kann daher nicht referenziert werden.)

Für eine aus Sicht des BSI sichere und vertrauenswürdige Hardware wären umfangreiche Herstellernachweise notwendig. Dem BSI müssten Entwicklungs- und Produktionsprozess, sowie sämtliche Firmware zur Evaluierung offengelegt werden. Das kann bei in Fernost produzierter Hardware nicht erwartet werden. Vielmehr müssen wird davon ausgehen, dass die Hardwarebasis grundsätzlich nicht vertrauenswürdig ist. Die Grundproblematik einer sicheren und vertrauenswürdigen Hardware soll daher an dieser Stelle nicht weiter vertieft werden.

An dieser Stelle soll vielmehr der Fokus auf das Betriebssystem gelegt werden. Der übliche Weg zum Nachweis der Sicherheit einer Software – und ein Betriebssystem ist letztlich nichts Anderes als Software – ist zunächst eine Evaluierung entlang von gewählten Sicherheitsstandards. Naheliegend sind dabei natürlich die Common Criteria, wobei das Niveau der Evaluierung dabei mindestens EAL 4+ sein muss. Hinsichtlich des Schutzniveaus ist darüber hinaus vom höchsten postulierbaren Angriffspotenzial auszugehen. Daher muss das Gesamtsystem inkl. sicherer Ablaufplattform gegen AVA.VAN.5 geprüft werden.

Leider gab es bis ins Jahr 2013, als wir mit unserem sicheren Netzübergang SDoT (Secure Domain Transition) Security Gateway die Anforderung der „sicheren Ablaufplattform“ erfüllen wollten, kein entsprechendes System zu kaufen. Es gab kein marktverfügbares Betriebssystem, das sämtliche Anforderungen des BSI erfüllte, denn eine Evaluierung von monolithischen Betriebssystemen wie Linux oder Windows mit ihren vielen Millionen Codezeilen, ist aus wirtschaftlichen Gründen nicht praktikabel. Es bestand also der dringende Bedarf, ein Betriebssystem zu finden, das einerseits evaluierbar ist, andererseits zusätzlich eine Reihe von Sicherheitsmerkmalen mitbringt, die für eine Zulassung im Kontext GEHEIM (und insbesondere für einen

sicheren Netzübergang) unerlässlich sind. Infodas und Kernkonzept haben sich daher 2013 dazu entschlossen, ein solches System gemeinsam selbst zu entwickeln: das SDoT Microkernel Operating System (SDoT MOS), das auf Kernkonzepts L4Re-Mikrokern basiert, sollte die Basis für den sicheren Netzübergang SDoT Security Gateway 6.0 werden.

3 Sicherheitstechnische Anforderungen

Aus der eingangs dargestellten Voraussetzung, dass das Betriebssystem evaluierbar sein muss, ergeben sich zwei Konsequenzen: erstens müssen die kritischen Komponenten des Betriebssystems ausreichend klein sein und zweitens muss der Quellcode zumindest den durch das BSI akkreditierten Prüfstellen und dem BSI offengelegt werden können. Damit fallen die meisten kommerziellen Closed-Source-Betriebssysteme von vornherein aus, aber auch die üblichen Varianten der verbreiteten Open-Source-Betriebssysteme sind zu umfangreich für eine Evaluierung.

Weitere Anforderungen sind:

- Nachweisbare Separierung von Prozessen

  Einzelne Systemkomponenten müssen umfassend voneinander isoliert werden können, um eine gewollte oder ungewollte Datenübertragung zwischen den Komponenten zu verhindern.

  Die Isolationseigenschaften eines herkömmlichen Betriebssystems wie Windows oder Linux sind dafür aus mehreren Gründen nicht ausreichend. Selbst wenn ein solches Betriebssystem mit Mechanismen für Mandatory Access Control erweitert und gehärtet wurde, können unerwünschte Übertragungskanäle nicht ausgeschlossen werden, unter anderem aus folgenden Gründen:

  Erstens sind die Programmierschnittstelle (API) und das Objektmodell herkömmlicher Betriebssysteme so komplex, dass der Nachweis der Abwesenheit von Kommunikationsmöglichkeiten über gemeinsam verwendete Betriebsmittel ein sehr schwieriges oder sogar unlösbares Problem darstellt.

  Zweitens erschwert die Größe herkömmlicher Betriebssysteme den Nachweis, dass die API (einschließlich ihrer Sicherheitsfunktionen) auch korrekt implementiert ist und erfolgreiche Angriffe darauf hinreichend unwahrscheinlich sind. Jede Fehlfunktion im privilegierten Kern dieser großen, monolithischen Betriebssysteme ist ein potentielles Sicherheitsloch, das die Separierungseigenschaften schwächt.

- Nachweisbare Separierung von Hardware-Komponenten und Peripherie

  Die meisten Peripheriegeräte, etwa Netzwerkgeräte und Massenspeichercontoller, nutzen direkte Speicherzugriffe (direct memory access – DMA), um Anwendungsdaten effizient zu verarbeiten, zu speichern oder zu übertragen. Traditionell ist DMA für den gesamten physischen Speicher eines Computers möglich. Besonders für Geräte und Treiber, die eine potentielle Angriffsfläche von außen haben, z.B. Netzwerkgeräte, -treiber und -protokolle, ist dies unerwünscht. Daher muss das Betriebssystem moderne Architekturmechanismen unterstützen, die eine Einschränkung der DMA-Bereiche auf einzelne Systemkomponenten erlauben, etwa IOMMUs (näheres dazu im nächsten Kapitel).

- Sichere Unterstützung von Standard-PC-Hardware

  Für die meisten Anwendungen ist die Verwendung von besonders sicherer, speziell entwickelter Hardware aus Kostengründen nicht möglich. Das Betriebssystem muss daher

Standard-PC-Hardware auf eine Weise unterstützen, die das Restrisiko einer Verwendung minimiert. Solche Restrisiken resultieren beispielsweise aus unbekannter Firmware, Managementfunktionen, unsicheren Boot-Mechanismen oder nicht vollständig vertrauenswürdigen Hardwarekomponenten, und die im letzten Absatz genannte IOMMU-Unterstützung ist ein Beispiel für eine Risikominimierung.

- Unterstützung eines „sicheren Bootens“

  Der Bootvorgang ist der Mechanismus, der von einem initialen Systemzustand (z.B. nach Power-On) zum Betrieb der Anwendung führt. Um von einem als vertrauenswürdig angenommenen initialen Zustand in einen sicheren Betriebszustand der Anwendung zu gelangen, muss auch der Bootvorgang sicher sein. Das Betriebssystem muss daher Mechanismen unterstützen (bzw. nicht behindern), die eine Unterwanderung des Systems beim Bootvorgang ausschließen. (In diesem Papier gehen wir auf diese Mechanismen nicht weiter ein.)

- In separierten Compartments müssen mindestens Linux, evtl. auch Windows-Betriebssysteme laufen.

  Diese Anforderung steht in einem Konflikt zu den bisher genannten Sicherheitsanforderungen, ist jedoch aus praktischen und ökonomischen Gründen nötig: Sowohl zur Unterstützung von Standard-PC-Peripheriekomponenten und zur Nutzung von Standard-Softwarefunktionen (z.B. Netzwerkprotokolle) ist die Nachnutzung vorhandener Treiber und Systemsoftware wünschenswert. Eine Herausforderung stellt jedoch die Isolation von Anwendungskomponenten dar, die diese Funktionen nutzen müssen. Die Lösung bietet ein Systemkonzept, in dem mehrere Linux- oder auch Windows-Gastsysteme in jeweils eigenen, separierten Compartments laufen und höchstens über wohldefinierte Schnittstellen miteinander kommunizieren können.

4 Zulassungsfähigkeit eines Betriebssystems

4.1 Evaluierbarkeit

Um die grundsätzliche Evaluierbarkeit zu realisieren, wird für zulassungspflichtige Systeme, die eine sichere Ablaufplattform benötigen, ein Mikrokern eingesetzt. Ein Mikrokern zeichnet sich im Gegensatz zu einem so genannten monolithischen Betriebssystem dadurch aus, dass nur ein sehr kleiner Anteil des Betriebssystems im „privilegierten“ Modus des Prozessors läuft. Nur in diesem Modus hat Software Vollzugriff auf die Hardware.

Bei monolithischen Betriebssystemen verfügt das gesamte Betriebssystem über diesen Vollzugriff. Erst die Anwendungsebene läuft im Nutzermodus, welcher nur eingeschränkte Zugriffsrechte auf die Hardware hat. Anders ausgedrückt: Software, die im privilegierten Modus läuft, muss „niemanden fragen“, ob die Festplatte oder der Arbeitsspeicher angesprochen werden darf. Software, die im Nutzermodus operiert, muss immer erst das Betriebssystem „um Erlaubnis bitten“, bevor ein Zugriff auf Ressourcen möglich ist.

Bei Mikrokernen ist der Teil des Betriebssystems, der im privilegierten Modus läuft, auf das absolut Notwendige reduziert. Dieser Teil muss daher auch regelmäßig evaluiert werden, ist aber auf Grund seiner vergleichsweise geringen Größe auch evaluierbar. Andere Betriebssystemkomponenten (einschließlich Gerätetreibern) laufen hingegen im Nutzermodus. Sie werden von vornherein durch den Mikrokern eingeschränkt und besitzen nur jene Zugriffsrechte auf Systemressourcen, die diese Komponenten zur Erfüllung ihrer Aufgabe benötigen.

Das mikrokernbasierte L4Re-System besteht aus mehreren Komponenten, die verschiedene Betriebssystemfunktionen implementieren. Jede Komponente ist dabei für einen Teilaspekt zuständig. Die Aufteilung in mehrere Komponenten ergibt ein modulares System, wobei die Komponenten so kombiniert werden können, dass jeweils nur diese Komponenten benutzt werden, die für die Funktionalität benötigt werden. Dieser Grundsatz gilt nicht nur systemweit, sondern explizit auch für Applikationen, d.h. eine Applikation muss nur den Komponenten vertrauen, von denen sie abhängt. Durch die vom Mikrokern durchgesetzte Isolation muss die Applikation anderen Komponenten im System nicht vertrauen. Demzufolge erlaubt ein komponentenbasiertes Mikrokernsystem es, die „Trusted Computing Base“ (TCB) einer Anwendung zu minimieren. Auf Grund dieser Eigenschaften kann das System Software mit verschiedenen Vertrauenseigenschaften auf einer Plattform ausführen.

4.2 Isolation und Zugriffsschutz

Die Implementierung der Isolation im System basiert auf mehreren Bausteinen. Der Speicherschutz wird mit Hilfe von Hardwaremechanismen umgesetzt. Die „Memory Management Unit“ (MMU) des Prozessors stellt hierbei virtuellen Speicher zur Verfügung, der es erlaubt, Applikationen ausschließlich selektiven Zugriff auf Speicher zu gewähren. Das gleiche Prinzip wird von so genannten IOMMUs (Input/Output MMU) angewendet, um Hardwaregeräten eingeschränkten Zugriff auf den Speicher des Systems zu gewähren. Mit Hilfe von IOMMUs werden DMA-basierte Angriffe und Fehler effektiv unterbunden.

Der zweite Baustein ist das Schutzkonzept für Kommunikation innerhalb des mikrokern-basierten L4Re-Systems, welches auf sog. Capabilities basiert. Capabilities sind Zugriffsrechte auf Objekte im System. Der Besitz einer Capability erlaubt somit den Zugriff auf ein Objekt in einer anderen Komponente. Capability-basierte Zugriffsmechanismen in Betriebssystemen werden allgemein als beste Variante der Umsetzung angesehen und stellen damit den aktuellen Stand der Technik dar. Mit Hilfe von Capabilities kann man feingranulare Rechte vergeben, so dass Komponenten nur die Rechte auf andere Komponenten haben, die sie brauchen. Dieses Konzept setzt das „Principle of Least Privilege“ um und sorgt dafür, dass die Ausbreitung eines Fehlers im System minimiert wird.

4.3 Separierung von Compartments

„Compartments“ sind Anwendungskomponenten bzw. virtualisierte Gast-Betriebssysteme (wie z.B. Linux oder Windows), die eine bestimmte Anwendungsfunktionalität erbringen, aber dennoch vollständig voneinander isoliert sein sollen. Eine Kommunikation zwischen Compartments soll nur auf wohldefinierte Weise möglich sein, nicht jedoch über unerwünschte Kanäle.

Der Schlüssel zur Separierung von Compartments ist die Einsicht, dass Kommunikation nur über ein gemeinsames Medium möglich ist. Dies könnte beispielsweise ein gemeinsamer Speicherbereich, ein gemeinsam verwendetes Hardware-Gerät oder ein geteiltes Objekt im Betriebssystem sein. Um Kommunikation effektiv zu verhindern, ist daher eine vollständige Aufteilung aller Systemressourcen notwendig.

Da im Capability-basierten Mikrokernsystem alle Betriebssystem-Objekte durch Capabilities benannt und geschützt werden, ist eine Entwurfs- und Konfigurationsrichtlinie notwendig, die den Compartments jeweils disjunkte Capabilities für Speicher, Geräte usw. zuweist. Damit ist auf Ebene der Systemobjekte und des Nutzerspeichers eine Isolation hergestellt.

Besondere Beachtung müssen jedoch solche Ressourcen finden, die nicht auf API-Ebene mit Capabilities adressiert werden (z.B. Mikrokern-interne Daten) oder die nicht architekturell adressierbar, aber dennoch beobachtbar sind (z.B. CPU-interne Caches). Im Folgenden stellen wir beispielhaft zwei Strategien vor, um dennoch eine Kommunikation über diese Ressourcen zu verhindern.

Um Kommunikation über gemeinsam verwendete Datenstrukturen innerhalb des Betriebssystemkerns zu verhindern, verfolgen wir die Strategie, weitgehend auf solche geteilten Datenstrukturen im Kern zu verzichten. Durch die Konstruktion als Mikrokern sind die meisten Kernobjekte tatsächlich an der API sichtbar und mit Capabilities verwaltbar, sodass die oben genannte Entwurfsrichtlinie anwendbar ist. Eine Ausnahme stellt in L4Re jedoch der vom Kern für die Kernobjekte verwendete Hauptspeicher dar, da dieser nicht feingranular mit Capabilities partitioniert werden kann. Kernspeicher kann zwar von Anwendungsprogrammen nicht direkt zur Kommunikation verwendet werden; jedoch ist eine verdeckte Kommunikation durch das Erzeugen von Out-of-Memory-Situationen denkbar (verdeckter Kanal durch Möglichkeit der Beobachtung einer fehlgeschlagenen Allokation). Um dem zu entgegnen, verwendet der Mikrokern einen einfachen Quotierungsmechanismus, der tasklokale Kernspeicherlimits durchsetzt, sodass keine globale Beobachtbarkeit mehr möglich ist. Die Quotas sind Teil der Mikrokern-Konfiguration.

Um eine Datenübertragung durch Manipulation oder Beobachtung von CPU-Caches zu verhindern, bieten sich zwei Möglichkeiten an. Erstens ist die Separierung der Ausführung der Compartments auf unterschiedliche CPU-Cores möglich. Wenn diese CPU-Cores keine gemeinsamen Caches verwenden, ist das Problem bereits gelöst; ansonsten muss im Rahmen einer Restrisikobetrachtung ermittelt werden, ob über jene Caches, die zwei Cores miteinander teilen (z.B. nur Level-3-Cache), noch eine nennenswerte Bandbreite übertragen werden kann.

Ist eine Core-Separierung nicht möglich oder nicht gewünscht, kann auch ein Leeren bestimmter Caches (Cache Flush) beim Prozesswechsel zwischen Compartments sinnvoll sein. Der damit einhergehende Leistungsverlust ist bei vielen Anwendungen verschmerzbar, weil – je nach Zeitscheibenlänge und Größe der Speicher-Arbeitsmenge – zwischen zwei Zeitscheiben eines Gastbetriebssystems häufig sowieso kaum noch nützlicher Cache-Inhalt zurückbleibt.

4.4 Sichere Inter-Compartment-Kommunikation

Eine weitere, notwendige Sicherheitsanforderung ist, dass das gemeinsame Nutzen von Ressourcen minimiert wird, um Angriffsflächen weiter zu reduzieren. Ein Übersprechen ist auch dann zu minimieren, wenn Kommunikation über gemeinsame genutzte Ressourcen ausdrücklich benötigt und gewünscht wird.

Eine Möglichkeit zur Einschränkung der Angriffsfläche ist, nur spezifische Hauptspeicherbereiche zur Kommunikation zwischen Compartments zu nutzen und dem Kommunikationsmechanismus keinen generellen Zugriff auf den gesamten Speicher der Gast-Betriebssysteme in den virtuellen Maschinen (VM) / Compartments zu gestatten. Diese Lösung schließt jedoch Standardmechanismen zur Inter-VM-Kommunikation aus, denn diese nutzen zur Kommunikation virtuelle Geräte (z.B. ein virtuelles Netzwerkgerät), die wie ein echtes Gerät mit (virtuellem) DMA Zugriff auf den gesamten Gastspeicher haben.

Stattdessen haben wir für SDoT MOS einen speziellen Mechanismus entwickelt, der sich zwar aus dem Gast-System im Compartment heraus wie ein Standard-Netzwerk-Socket ansteuern lässt, der jedoch vom Mikrokernsystem bereitgestellten, separaten Speicher verwendet. Die

Kommunikationskomponente kann pro Compartment-Kommunikationsbeziehung instanziiert werden und hat nur auf diese separaten Speicherbereiche Zugriff, sodass die Angriffsfläche für diesen kritischen Kommunikationsmechanismus äußerst klein ist: Etwaige Angriffe auf diese Komponente können höchstens die (gesicherten) Kommunikationsinhalte zwischen diesen Compartments beobachten und manipulieren, aber nicht darüber hinaus die Vertraulichkeit der beteiligten Compartments verletzen.

5 Verwandte Arbeiten

Das mikrokernbasierte L4Re-System und das darauf aufbauende, GEHEIM-taugliche SDoT MOS, gehören zur Familie der L4-Mikrokerne, zu der auch andere moderne Systeme wie seL4 oder PikeOS gehören [ElHe13]. Diese Mikrokerne der zweiten Generation setzen auf ein striktes „Principle of Least Privilege“ und bieten die Möglichkeit, selbst grundlegende Betriebssystemdienste (wie zum Beispiel Adressräume und Seitenauslagerung) in nichtprivilegierten, separierten Nutzerprozessen zu implementieren. Daher sind L4-Mikrokerne besonders klein, enthalten entsprechend wenig privilegierten Code und ermöglichen so – anders als herkömmliche Betriebssysteme wie Windows oder Linux – besonders kleine und gut evaluierbare TCBs.

Während frühe L4-Mikrokerne besonders leistungsorientiert entworfen waren und anstrebten, im Vergleich zu früheren Mikrokernen wie Mach [ABBG86] – zugunsten eines geringeren Overheads – nur möglichst wenige, hardwarenahe, aber flexible Funktionen zu implementieren [Lied93, HHLS97], spezialisierten sich spätere L4-Systeme stark, sodass es heute in der Familie der L4-Mikrokerne ein großes Spektrum an Lösungen und entsprechend unterschiedlichen Eigenschaften gibt: PikeOS und OKL4 wendeten sich eingebetteten und Safety-kritischen Anwendungen zu und gaben zugunsten besserer Beherrschbarkeit, Safety-Zertifizierbarkeit und Echtzeit-Eigenschaften bestimmte dynamische Eigenschaften auf, etwa die Möglichkeit, geteilte Speicherbereiche wieder zu entziehen. L4Re, Nova und seL4 wendeten sich eher Security-kritischen Anwendungen zu und ergänzten die klassische L4-API um Capability-Systeme (inspiriert durch EROS [ShSF99]) und hardwareunterstützte Virtualisierungsfeatures. [HeLe10, LaWa09, StKa10]

Der seL4-Mikrokern wurde darüber hinaus in bestimmten Konfigurationen erfolgreich formal (mathematisch) verifiziert [KEHA09]. Um dies zu ermöglichen und die Komplexität des Mikrokerns zu senken, delegiert seL4 noch kompromissloser bestimmte komplexe Betriebssystemaufgaben an Nutzerprozesse, zum Beispiel die Verwaltung des Hauptspeichers für den Mikrokern selbst. Diese Designentscheidung verkompliziert allerdings im Gegenzug die TCB auf Nutzerprozessebene und erschwert die Implementation von flexiblen, dynamischen Systemen.

Das L4Re-System und das darauf aufbauende SDoT MOS nehmen dagegen im Mikrokern eine L4-typische, etwas höhere Komplexität in Kauf, um sowohl statische als auch dynamische Anwendungsfälle ohne signifikante Komplexität auf Nutzerprozessebene zu ermöglichen. Es kann zwar derzeit keine formale Verifikation vorweisen, aber trotzdem werden per Konstruktion bestimmte Fehlerklassen, z.B. Pufferüberläufe bei der Interpretation von Nutzerdaten im Kern, vollständig ausgeschlossen, was die Evaluierbarkeit erhöht.

SDoT MOS unterscheidet sich vom – auch als Open Source erhältlichen – L4Re-System durch einige Besonderheiten bei der Konfiguration und durch Erweiterungen, die das System zu einer GEHEIM-tauglichen sicheren Ablaufplattform aufwerten. Dazu gehört insbesondere der in Abschnitt 4.4 beschriebene Inter-VM-Kommunikationsmechanismus.

Im Anwendungsfall „Sicherer Netzübergang“, hier in Form des Produkts SDoT Security Gateway 6.0, wird SDoT MOS in einer weitgehend statischen Konfiguration betrieben. Das heißt, dass die Konfiguration der Compartments und der Verbindungen dazwischen fest ist und sich zur Laufzeit nicht ändern kann. SDoT MOS unterstützt jedoch – wie L4Re, jedoch im Unterschied zu klassischen Separation Kernels wie Integrity, LynxSecure und Muen [BuRu13] – auch dynamische Anwendungsfälle, einschließlich dynamischer Instanziierung von Compartments und Updates der Sicherheitsfunktionalität unter Beibehaltung aller Sicherheitseigenschaften.

Das SDoT Security Gateway 6.0 ist das erste in Deutschland zugelassene Rot-Schwarz-Gateway, das auf einer GEHEIM-tauglichen, sicheren Ablaufplattform basiert (hier SDoT MOS) und das eine „allgemeine“ Zulassung bis GEHEIM erhalten hat. Das BSI hat die Zulassung im April 2017 erteilt [Bund17].

Frühere Versionen des SDoT Security Gateway und vergleichbare andere Systeme, etwa das Produkt „Security Exchange Gateway SEG“ in der Version 3.0 der Airbus Defense and Space, basieren lediglich auf einem gehärteten Linux-Betriebssystem [Airb16, Info16] und haben daher nur Zulassungen für geringere Geheimhaltungsstufen [Bund17] oder projektspezifische Einzelzulassungen [Info16] erhalten.

6 Fazit

Wir konnten den Mikrokern als Unterbau eines sicheren Netzübergangs erfolgreich realisieren. Das entsprechende Produkt erfüllt alle BSI-Anforderungen an eine sichere Ablaufplattform. Somit konnte erstmals die besonders herausfordernde und kritische Anforderung an die sichere Ablaufplattform erfüllt werden. Damit ist der verwendete Mikrokern geeignet, auch für zukünftige, sicherheitskritische Anwendungen die Basis eines zulassungsfähigen Gesamtsystems zu bilden.

Literatur

[ABBG86] Mike Accetta, Robert Baron, William Bolosky, David Golub, Richard Rashid, Avadis Tevanian, and Michael Young. “Mach: A New Kernel Foundation for UNIX Development,” 1986.

[Airb16] Airbus Defense and Space: “Secure Exchange Gateway,” Produktdatenblatt 2016

[Bund10] Allgemeine Verwaltungsvorschrift des Bundesministeriums des Innern zum materiellen und organisatorischen Schutz von Verschlusssachen (VS-Anweisung – VSA) vom 31. März 2006 in der Fassung vom 26. April 2010 (GMBl 2010, S. 846); hier: §37 (1) Nr.6

[Bund17] Bundesamt für Sicherheit in der Informationstechnik: „BSI-Schrift 7164: Liste der zugelassenen IT-Sicherheitsprodukte und -systeme“, online, abgerufen am 15.06.2017

[BuRu13] Reto Buerki and Adrian-Ken Rueegsegger: “Muen - an x86/64 separation kernel for high assurance.” Technical report, University of Applied Sciences Rapperswil (HSR), Switzerland, August 2013.

[ElHe13] Kevin Elphinstone and Gernot Heiser: “From L3 to seL4 what have we learnt in 20 years of L4 microkernels?” in Proceedings of the 24th ACM Symposium on

Operating Systems Principles (SOSP '13). ACM, New York, NY, USA, pp. 133–150.

[HeLe10] Gernot Heiser and Ben Leslie: "The OKL4 microvisor: Convergence point of microkernels and hypervisors," in Proceedings of the First ACM Asia-pacific Workshop on Workshop on Systems, APSys '10, pages 19–24.

[HHLS97] H. Härtig, M. Hohmuth, J. Liedtke, S. Schönberg, and J. Wolter: "The performance of µ-kernel-based systems," in Proceedings of the 16th ACM Symposium on Operating System Principles, ser. SOSP. ACM, October 1997, pp. 66–77.

[Info16] INFODAS GmbH: "SDoT Security Gateway 5.0", Produktdatenblatt 2016

[KEHA09] G. Klein, K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood: "seL4: Formal verification of an OS kernel," in Proceedings of the 22nd ACM Symposium on Operating Systems Principles, ser. SOSP. ACM, October 2009, pp. 207–220.

[LaWa09] A. Lackorzynski and A. Warg: "Taming subsystems: Capabilities as universal resource access control in L4," in Proceedings of the 2nd Workshop on Isolation and Integration in Embedded Systems, ser. IEES. ACM, March 2009, pp. 25–30.

[Lied93] Jochen Liedtke: "Improving IPC by kernel design," in Proceedings of the 14th ACM symposium on Operating systems principles (SOSP '93). ACM, New York, NY, USA, pp. 175–188

[OuLR12] S. Oudkerk, G. Lunt, A. Ross: Technical Note 1455 REV 1, NATO PROFILE FOR THE "BINDING OF METADATA TO DATA OBJECTS", VERSION 1.1, Dezember 2012, NCIA, The Hague

[OuLR13] S. Oudkerk, G. Lunt, A. Ross: Technical Note 1456 REV 1, NATO PROFILE FOR THE "XML CONFIDENTIALITY LABEL SYNTAX", VERSION 1.1, Januar 2013, NCIA, The Hague

[ShSF99] Jonathan S. Shapiro, Jonathan M. Smith, and David J. Farber: "EROS: a fast capability system," in Proceedings of the seventeenth ACM symposium on Operating systems principles (SOSP '99). ACM, New York, NY, USA, pp. 170–185.

[StKa10] Udo Steinberg and Bernhard Kauer: "NOVA: a microhypervisor-based secure virtualization architecture," in EuroSys '10: Proceedings of the 5th European conference on Computer systems, pages 209–222.