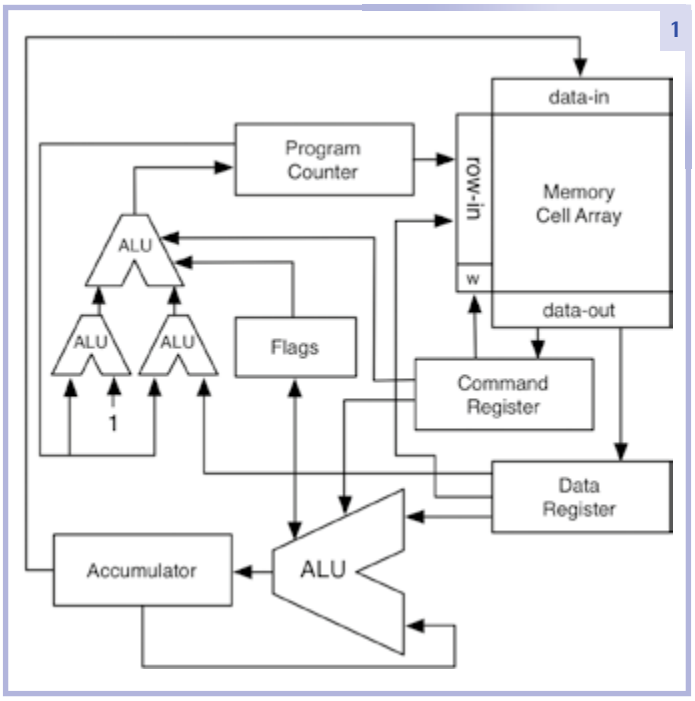
Vertrauliche Datenverarbeitung in der Cloud

WIE SCHÜTZT MAN PROGRAMME UND DATEN IN VERTEILTEN SYSTEMEN?

Die Cloud ist wohl das meistzitierte Stichwort in der IKT-Branche der jüngeren Vergangenheit. Sie ist maßgeblich dafür verantwortlich, dass verteilte IT-Systeme, zu denen auch das Grid oder Web-Applikationen zählen, in das öffentliche Bewusstsein gerückt sind. Neben der externen Datenspeicherung umfassen verteilte Systeme auch zunehmend Services, die extern gespeicherte Daten aktiv verarbeiten. Dadurch ergeben sich neue Herausforderungen zur Wahrung der Vertraulichkeit sowie der Privatsphäre des Anwenders.



1

Die Welt wird mobiler. Das zeigt sich für den Benutzer von elektronischen Geräten vor allem dadurch, dass er plötzlich in der Lage ist, auf seine Fotos, Videoclips, Apps oder sonstige Daten von überall aus zugreifen zu können. Vorausgesetzt, er hat sie in der Cloud abgelegt, um sie dann beliebig abrufen zu können; bevorzugt über Breitband-Funknetze, über die beispielsweise Smartphones mit dem Internet verbunden sind. Neben der Mobilität hat die Cloud für den Anwender den Vorteil, dass die Daten leicht von allen seinen internetfähigen Geräten erreichbar sind, also beispielsweise auch vom Desktop-Computer. Außerdem werden so genannte Online-Desktops und Online-Offices angeboten, bei denen der Anwender seine Daten direkt über den Browser auf einem virtuellen Computer *im Internet* bearbeiten kann.

Ein verbreitetes Cloud-Szenarium ist die Anmietung externer Rechenleistung und Infrastrukturkomponenten durch professionelle IT-Anwender und -Unternehmen, die diese Rechen-, Netzwerk- und Speicherkapazitäten nutzen, um eigene Ressourcen auszulagern (Stichwort Outsourcing) oder Lastspitzen in der eigenen Verarbeitung durch bedarfsgerechten Leistungszukauf abzufedern. Eine hohe *Elastizität* kennzeichnet dieses Vorgehen, weil Abruf und Rückgabe von Mietressourcen auch in kurzen Intervallen erfolgen können. In diesem Zusammenhang werden vorwiegend virtuelle Maschinen angefragt, die über preisbestimmende Parameter für CPU-Leistung und Hauptspeichergröße konfiguriert werden. Ergänzend werden Datenspeicher angeboten, die transparent integriert als virtuelle Festplatten fungieren. Es werden hier also nicht nur Daten gespeichert, sondern auch durch Programme auf diesen Fremdressourcen verarbeitet. Die aktive Programmausführung bedeutet derzeit jedoch eine unverschlüsselte Ablage der Programme und der zu verarbeitenden Daten auf der externen Ressource, da die Prozessoren der gemieteten Maschinen den Programmcode nur im Klartext verarbeiten können.

Anbieter von Internetdiensten verwenden oft selbst Cloud-Ressourcen, um darauf aufsetzend Mehrwertdienste für den Endanwender zu erstellen. Dadurch wird dieser indirekt auch von der Vertrauenswürdigkeit des Cloud-Providers abhängig. Der Endanwender kann in aller Regel aber nicht technisch kontrollieren, wo genau seine Daten verarbeitet

werden. So könnte der tatsächliche Standort Frankfurt, Berkeley County oder Hongkong sein, samt der dort gültigen Rechtsprechung. Dabei haben beipielsweise US-amerikanische Internet-Unternehmen aufgrund des *USA PATRIOT Act* von 2001 alle gespeicherten Daten für die US-Behörden offenzulegen. In vielen Fällen ist es nicht erwünscht oder aus vertragsrechtlichen Gründen nicht möglich, die *unverschlüsselte Delegation* einer Rechenleistung unter diesen Umständen durchzuführen.

Homomorphismus

Homomorphe Kryptografie basiert auf einer strukturerhaltenden Abbildung zwischen den zwei Mengen *Klartext* und *Chiffretext*. Konkret bedeutet dies, dass beispielsweise die Addition zweier Klartextoperanden das gleiche Ergebnis liefert, wie das entschlüsselte Additionsergebnis derselben verschlüsselten Operanden. Mit den Funktionen V und E für die Ver- und Entschlüsselung, zwei Operanden a und b, sowie dem Operator + gilt E(V(a)+V(b)) = a+b. Gilt gleichzeitig für einen Operator *, dass E(V(a)*V(b)) = a*b, handelt es sich um einen algebraischen Homomorphismus.

Der kryptografische Schutz eines aktiven Programms ist technisch allerdings ungleich schwieriger als die verschlüsselte passive Datenablage. Die entfernte Prozessorressource muss unter Wahrung der Vertraulichkeit in die Lage versetzt werden, ein verschlüsseltes Programm auf verschlüsselten Daten auszuführen, ohne beides auch nur partiell zu entschlüsseln. Dies schien lange Zeit gar nicht möglich und ist bis heute nicht effizient gelöst. Allerdings ist die Forschung gerade in den letzten beiden Jahren einen großen Schritt vorangekommen. Während *Yao's Garbled Circuits* [3] bis heute das effizienteste Modell zur verschlüsselten Ausführung boolescher Schaltkreise und damit auch von Algorithmen darstellen (allerdings mit sehr begrenzter Funktionalität), werden seit einer Grundlagenarbeit aus dem Jahre 2009 [1] vor allem *mathematische Homomorphismen* (siehe Infobox) zur Verschlüsselung von booleschen Schaltkreisen untersucht. Der konzeptionelle Vorteil liegt vor allem darin, dass Algorithmen von unbeschränkter Funktionalität abgebildet werden können.

Ein verschlüsselter Mikroprozessor

Auch die verschlüsselte Programmausführung mittels homomorpher Kryptografie basiert auf der Überführung eines Programms in eine Repräsentation aus booleschen Schaltkreisen. Dies verringert die erforderliche strukturelle Komplexität des Kryptosystems, weil lediglich recht einfache Basisoperationen, wie AND und XOR oder NAND dargestellt werden müssen. Die Schaltkreisabbildungen lassen sich nach dem Schritt der *Arithmetisierung* (siehe Infobox) mit den entsprechenden Mitteln des Kryptosystems verschlüsseln und werden anschließend dem Cloud-Anbieter als Datenpaket übergeben. Dieser kann mit der unverschlüsselten Verarbeitungsvorschrift – dem booleschen Schaltnetz des Anwendungsprogramms – die verschlüsselten Operanden in der richtigen Reihenfolge homomorph addieren bzw. multiplizieren. Das so berechnete, verschlüsselte Ergebnis (siehe Infobox *Beispiel*) kann nur der Auftraggeber mit seinem geheimen Schlüssel entschlüsseln. Der Programmausführer kann hier zwar den verwendeten Algorithmus nachvollziehen, da er ihm lesbar vorliegt. Allerdings kann er weder Eingabe- noch Ausgabedaten lesen.

Abbildung 1
Der Datenpfad der verschlüsselten CPU zeigt die elementaren Bestandteile des Systems.

2

X	0 gerade	1 ungerade
0 gerade	gerade	gerade
1 ungerade	gerade	ungerade

+	0 gerade	1 ungerade
0 gerade	gerade	ungerade
1 ungerade	ungerade	gerade

Abbildung 2
Die Paritäten der ganzzahligen Addition und Multiplikation entsprechen den Booleschen Operationen XOR und AND.

Um in diesem Szenario auch ein verschlüsseltes Programm ausführen zu können, begeben wir uns auf die niedrigere Abstraktionsstufe der Prozessorebene. Die unverschlüsselte Verarbeitungsvorschrift implementiert in diesem Fall die Schaltkreise eines Mikroprozessormodells, dessen Datenpfad beispielhaft in Abbildung 1 dargestellt ist. Der verschlüsselte Teil enthält das eigentliche binäre Anwendungsprogramm samt Daten. Da die unverschlüsselte Prozessor- und Speicherschaltung in jedem Takt immer gleichförmig abgearbeitet werden muss, sind wegen der verschlüsselten Operanden von außen keine Folgerung auf im Inneren ablaufende Operationen, Speicherzugriffe, Programmsprünge oder -schleifen möglich. Eine Implementierung dieses Konzeptes [6] ist auf unserer Projekt-Webseite http://www.hcrypt.com als Prototyp mit einem verschlüsselnden Assembler zur Programmentwicklung verfügbar.

Der Preis für die universelle Einsetzbarkeit eines solchen Systems ist vor allem die benötigte Rechenzeit, was auf zwei Faktoren zurückzuführen ist. Zum einen erfordert das unterliegende Kryptosystem aufgrund seiner mathematischen Beschaffenheit eine recht große Zusatzrechenleistung. Diese dient der zyklischen Laufzeitkorrektur von systembedingten Eigenschaften, wie der Genauigkeit einzelner Chiffretexte. Zum anderen benötigt der Speicherzugriff aufgrund der seriellen Schaltkreissimulation verhältnismäßig viel Zeit. Tatsächlich steigt hier die Zugriffszeit auf eine einzelne verschlüsselte Speicherzelle linear mit der gesamten verschlüsselten Speichergröße an.

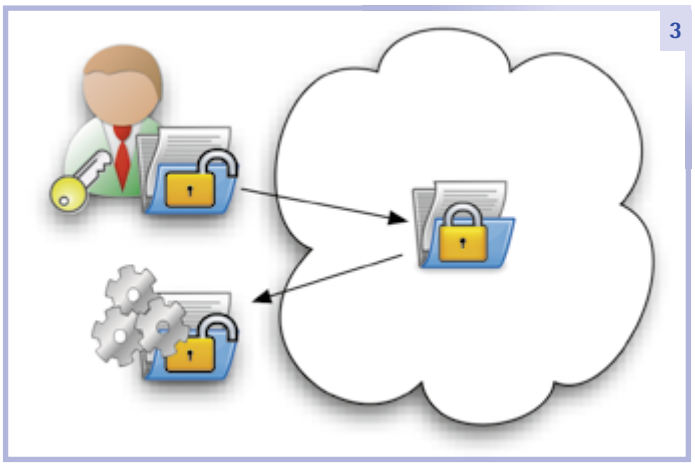


Abbildung 3
Der gewohnte Ablauf: der Anwender speichert die Daten verschlüsselt in der Cloud und bearbeitet unverschlüsselt lokal.

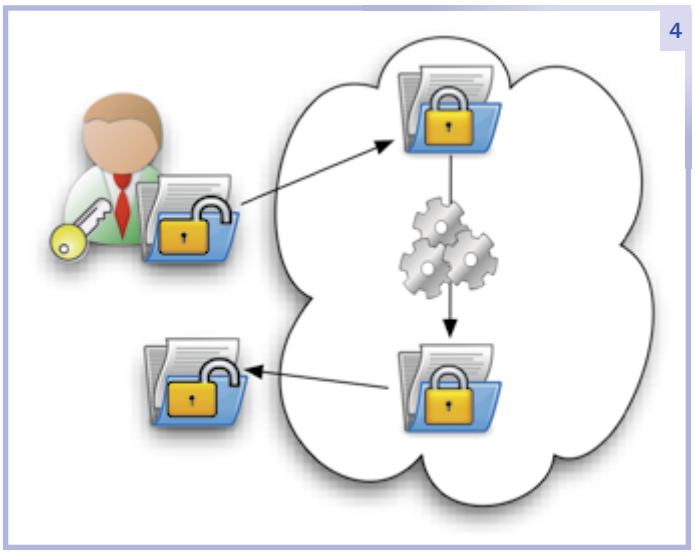


Abbildung 4
Vertrauliche Verarbeitung in der Cloud: Der Provider bearbeitet die Daten im verschlüsselten Zustand.

Beispiel für homomorphe Kryptografie

Operanden a=5, b=2; Schlüssel p=19; Zufallswerte r1=4, r2=7; Verschlüsselung: a'=a +r1*p=5+4*19=81; b'=b+r2*p=2+7*19=135; Operation +: a' + b'=216; Entschlüsselung: 216 mod 19=7; Operation *: a' * b'=10935; Entschlüsselung: 10935 mod 19=10
Zu unterscheiden sind homomorphe Schemata, die nur eine begrenzte Operationszahl unterstützen und solche, die eine *unbegrenzte Tiefe* gestatten. Das Beispiel ist korrekt, solange a*b<p. Zur tatsächlichen Verschlüsselung müssen entgegen des Beispiels eine große Primzahl für p und Zufallszahlen mit großem Primfaktor für rx gewählt werden, um die unerwünschte Entschlüsselung durch Primfaktorisierung der Chiffretexte zu verhindern.

Weiterführende Forschung

Zur Verbesserung der Performance untersucht die Gruppe DCSec derzeit zwei Forschungszweige. Die naheliegende *Hardware-Implementierung* eignet sich zwar nicht direkt für den Einsatz in der Cloud, jedoch können mit Chips nach diesem Design so genannte Hardware-Software-Packages abgesichert werden. Das sind zum Beispiel industrielle Steuerungsanlagen mit geheimen Steuerungsalgorithmen oder in aufwändigen

Arithmetisierung

Ausgehend von auf arithmetischen Operationen beruhenden Homomorphismen müssen boolesche Operationen zunächst in ihre arithmetische Entsprechung überführt werden. Außerdem erfordern die binären Schaltwerte ebenfalls jeweils ein verschlüsselbares, ganzzahliges Pendant. Betrachtet man die Ergebnisparitäten bezogen auf die Addition und Multiplikation ganzer Zahlen (Abb. 2), so lässt sich offenbar eine binäre 1 als ungerade, eine binäre 0 als gerade Zahl abbilden. Die boolesche XOR-Operation entspricht dann der ganzzahligen Addition und die boolesche AND-Operation der ganzzahligen Multiplikation.

Messreihen ermittelte Steuerungswerte. Darüber hinaus ist die Sicherung von Sensortechnik oder der Betriebssoftware digitaler Endgeräte denkbar.

Weiterhin werden *Hybridsysteme* untersucht, in denen ein Großteil der Verarbeitung unverschlüsselt stattfindet und nur der zur Wahrung der Vertraulichkeit erforderliche Bestandteil verschlüsselt wird. So wurde bereits eine geheime Suche in großen medizinischen Datenbeständen realisiert, bei der eine prinzipiell unverschlüsselte Vorsuche nach Kandidaten stattfindet. Durch die Verwendung von Hashes bleibt die Treffermenge jedoch verdeckt und kann in einem zweiten Schritt mit homomorph verschlüsselten Schaltkreisen effizienter als bisher konkretisiert werden.

Literatur

[1] Gentry, Craig: Fully Homomorphic Encryption Using Ideal Lattices. In: Proceedings of the 41st annual ACM symposium on Theory of computing (STOC), 2009, ISBN 978-1-60558-506-2, S. 169–178

[2] Smart, Nigel; Vercauteren, Frederic: Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. Version: 2010, In: Public Key Cryptography (PKC) Bd. 6056. Springer Berlin / Heidelberg, 2010, ISBN 978-3-642-13013-7, S. 420–443

[3] Yao, Andrew C.: Protocols for Secure Computations. In: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS), IEEE Computer Society 1982, ISSN 0272-5428

[5] Brenner, Michael; Smith, Matthew: Verfahren und Vorrichtung zur Berechnung homomorph verschlüsselter Programme., eingereichtes Patent, Deutsches Patent- und Markenamt (DPMA) amtl. Az. 10 2011 012 328.8 (2011)

[6] Brenner, Michael; Wiebelitz, Jan; von Voigt, Gabriele; Smith, Matthew: Secret Program Execution in the Cloud Applying Homomorphic Encryption. In: Proceedings of the 5th IEEE International Conference on Digital Ecosystems and Technologies (DEST), 2011, ISBN 978-1-4577-0872-5, S. 114–119

[8] Perl, Henning; Brenner, Michael; Smith, Matthew: POSTER: An Implementation of the Fully Homomorphic Smart-Vercauteren Cryptosystem. In: CD Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS), 2011, ISBN 978-1-4503-0948-6

M.Sc. Michael Brenner
Jahrgang 1974, ist seit 2008 wissenschaftlicher Mitarbeiter in der Distributed Computing & Security Group des Forschungszentrums L3S. Sein Forschungsschwerpunkt liegt im Bereich des verschlüsselten Rechnens und der Anwendungsarchitekturen für homomorphe Kryptosysteme. Kontakt: brenner@dcsec.uni-hannover.de

Prof. Dr. Matthew Smith
Jahrgang 1978, ist seit 2009 der Leiter der Distributed Computing & Security Group und Mitglied im Forschungszentrum L3S. Er forscht im Spannungsfeld von komplexen verteilten Systemen, deren Sicherheit und einfacher Benutzbarkeit. Kontakt: smith@dcsec.uni-hannover.de

Die Public-Domain Piktogramme in den Diagrammen stammen von **openclipart.org** und **iconarchive.com**.