# Policy-Framework – Eine Methode zur Umsetzung von Sicherheits-Policies im Cloud-Computing

Alexander Blehm, Volha Kalach, Alexander Kicherer, Gustav Murawski,
Tim Waizenegger, Matthias Wieland

Institut für Architektur von Anwendungssystemen (IAAS),
Institut für Parallele und Verteilte Systeme (IPVS)
Universitätsstraße 38
70569 Stuttgart
{blehm.ar, olga.kalach}@gmail.com
{Matthias.Wieland, Tim.Waizenegger}@ipvs.uni-stuttgart.de

**Abstract:** Cloud-Computing gewinnt immer mehr an Bedeutung bei der kosteneffizienten und skalierbaren Bereitstellung von IT-Diensten. Damit sich Cloud-Computing jedoch durchsetzen kann, muss die Sicherheit und Compliance der Dienste garantiert werden, d. h. die Einhaltung von Gesetzen, Richtlinien und Datenschutzvorgaben. Um diese Ziele zu erreichen, wird in diesem Beitrag ein Policy-Framework vorgestellt, welches die Umsetzung von Sicherheits-Policies im Cloud-Computing ermöglicht. Eine Policy beschreibt dabei nicht-funktionale Anforderungen an Cloud-Dienste. Des weiteren werden verschiedene prototypisch umgesetzte Policies beschieben und es wird ein Moodle-System als Anwendungsbeispiel für einen Cloud-Dienst mit den vorhandenen Policies annotiert. Dadurch erfolgt eine Evaluation des Policy-Frameworks.

## 1 Einleitung

Unter Cloud-Computing versteht man das Bereitstellen von IT-Infrastrukturen über ein Netzwerk, wie beispielsweise das Internet. In diesem Beitrag werden nur anwendungsspezifische Infrastrukturen betrachtet, die Cloud-Dienste genannt werden. Bekannte Beispiele sind Google Drive oder OwnCloud. Die Architektur eines Cloud-Dienstes kann durch eine Topologie modelliert werden. Darunter verstehen wir in diesem Beitrag einen hierarchischen Bauplan für die Anwendung und alle anderen Komponenten des Cloud-Dienstes. Um diesen Bauplan zu beschreiben, gibt es die standardisierte Sprache Topology and Orchestration Specification for Cloud Applications (TOSCA) [Kni13]. Cloud-Dienste, die in TOSCA beschrieben sind, haben den entscheidenden Vorteil, dass sie automatisiert bereitgestellt werden können. Cloud-Computing trifft aber häufig auf Kritik bezüglich der Sicherheit. Sämtliche Daten liegen nicht mehr im Einflussbereich des Endnutzers, sondern im Bereich des Betreibers der Cloud-Infrastruktur. Die Umsetzung von effektiven Sicherheitsstandards bringt hohe Kosten mit sich. Aufgrund von mangelnden und unausgereiften Alternativen wird auf die bestehenden Sicherheitslösungen der Technologie vertraut, ohne ihre Funktionalitäten und Limitationen ausreichend zu kennen [MSM08].

OpenTOSCA ist eine Open-Source-Implementierung einer TOSCA-Laufzeitumgebung [BBH+13]. Sie wird verwendet, um Cloud-Dienste automatisiert bereitzustellen. Dies wird auch „Deployment“ genannt. Um während des Deployments Sicherheitsanforderungen umzusetzen, wurde das Policy-Framework entwickelt. Dieses erlaubt es, Policies umzusetzen und beim Design des Cloud-Diensts zu nutzen. Policies definieren Richtlinien, die nicht-funktionale Anforderungen, die an Cloud-Dienste stellen. In diesem Beitrag werden vor allem Sicherheits-Policies betrachtet. Eine Policy kann an einen Cloud-Dienst annotiert werden, was ausdrückt, dass die darin definierten Anforderungen für diesen Dienst gelten sollen. Die Syntax und die Struktur einer Policy werden durch den TOSCA-Standard definiert [Kni13]. Dieser definiert dazu sogenannte Policy-Types und Policy-Templates. Die so definierten Policies können in der OpenTOSCA Umgebung bereitgestellt, und von allen Cloud-Diensten derselben Umgebung genutzt werden. Ein Vorreiter der hier beschriebenen Policies wurde schon vom Bundesamt für Informationstechnik unter dem Namen „IT-Sicherheitsleitlinie“ definiert [fSidI]. Mit den Instituten IAAS und IPVS der Universität Stuttgart wurden in einem Studienprojekt mehrere beispielhafte Policies für OpenTOSCA implementiert. Diese Policies, deren Funktionsweise sowie eine generische Anleitung zur Erstellung neuer Policies werden in diesem Beitrag beschrieben.

Dieser Beitrag ist wie folgt aufgebaut: In Kapitel 2 wird ein Beispielszenario für einen Cloud-Dienst mit Sicherheitsanforderungen vorgestellt. Danach werden in Kapitel 3 spezifische Policies und deren Implementierungen im Policy-Framework gezeigt. Daraufhin beschreibt Kapitel 4 eine Methode, welche die Integration dieser Policies in Cloud-Dienste ermöglicht. Kapitel 5 evaluiert das Policy-Framework anhand von Laufzeitmessungen. Abschließend fasst Kapitel 6 unsere Erkenntnisse zusammen und gibt einen Ausblick auf zukünftige Arbeiten.

## 2 Beispielszenario für einen abgesicherten Cloud-Dienst

Im Folgenden wird ein Beispielszenario für einen Cloud-Dienst mit annotierter Policy vorgestellt. Abbildung 1 stellt die Topologie dieser Anwendung in der Vino4Tosca Notation [BBK+12] dar. Ein Dienst kann manuell oder mit Werkzeugen wie Winery [KBBL13] modelliert und anschließend in einem standardisierten Format gepackt werden. Ein Standard-Format für OpenTOSCA ist beispielsweise das „Cloud Service ARchive“ Format, kurz „CSAR“[BBH+13]. Dieses CSAR-Archiv wird anschließend in OpenTOSCA geladen und ist dann zum Deployment bereit. Die in Abbildung 1 gezeigte Moodle-Topologie besteht aus einem Web-Server-Zweig für die PHP-Anwendung und aus einem Datenbank-Zweig, der auf einer separaten virtuellen Maschine läuft. Für diesen Cloud-Dienst schreibt eine Passwort-Policy Richtlinien für sichere Passwörter vor. Man sagt auch: Es ist eine Passwort-Policy am Moodle-Cloud-Dienst annotiert. Wichtige Passwörter sind in diesem Beispiel: ein Passwort für den Admin-Account der Anwendung, ein Passwort für den Root-Account der MySQL Datenbank und ein Passwort für den darin erstellten Datenbankbenutzer der Moodle-Anwendung. In der OpenTOSCA Laufzeitumgebung ist ein Business Process Execution Language Plan, kurz BPEL-Plan, dafür verantwortlich, dass das Deploy-

ment der Moodle-Topologie in der richtigen Reihenfolge durchgeführt wird. Beim Ausführen des Plans greift die Passwort-Policy und prüft, ob die Passwörter sicher genug sind, oder ob sie durch sichere, neu generierte Passwörter ersetzt werden müssen.

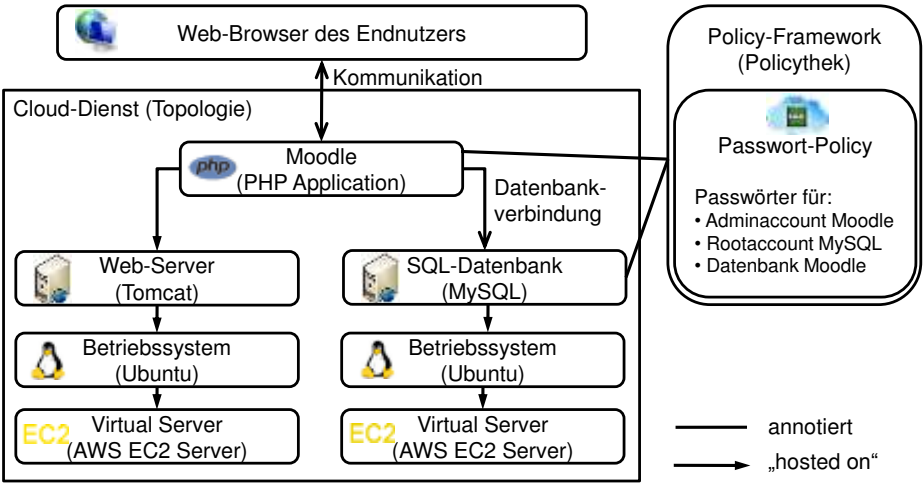


**Abb. 1.** Die Topologie einer Moodle-Anwendung mit Passwort-Policy

## 3 Prototypisch umgesetzte Policies im Policy-Framework

In diesem Kapitel werden die im Rahmen des Studienprojekts Policy-Framework prototypisch umgesetzten Policies vorgestellt. Auf Basis dieser ersten Policies wurde eine allgemeine Methode zur Umsetzung und Bereitstellung von weiteren Policies für die Realisierung anderer nicht-funktionaler Anforderungen erstellt. Diese wird in Kapitel 4 beschrieben. Die Definition der Syntax und die Struktur der Policies werden durch den TOSCA-Standard definiert [Kni13], in den folgenden Abschnitten geht es um die Implementierung der Policies, damit diese bei dem automatischen Bereitstellen des Cloud-Dienstes eingehalten und umgesetzt werden.

### 3.1 Einschränkung der Datenschutzregion durch die Region-Policy

Das Benutzen von virtuellen Maschinen (kurz VMs) zum Betrieb von Cloud-Diensten kann beim Mieten von VMs im Ausland zu Verstößen gegen das Datenschutzgesetz führen, was die Datenhaltung möglicherweise illegal macht. Ebenso möchte man unter Umständen für seine Cloud-Dienste nur bestimmte Rechenzentren zulassen, deren Betreibern man vertraut. Die Region-Policy ist eine Policy, welche mit einem Webservice und einer dazugehörigen Datenbank Regionen verwaltet, wie in Abbildung 2 dargestellt. Bei Cloud-Diensten, an denen eine Region-Policy annotiert ist und bestimmte zugelassene Regionen definiert sind, wird von dem Webservice vor dem Erstellen einer virtuellen Maschine (auch Instanziieren genannt) überprüft, ob diese in

einer gültigen Region instanziiert wird. Ein Deployment des Systems in einer nicht zugelassenen Region wird durch einen Abbruch verhindert.

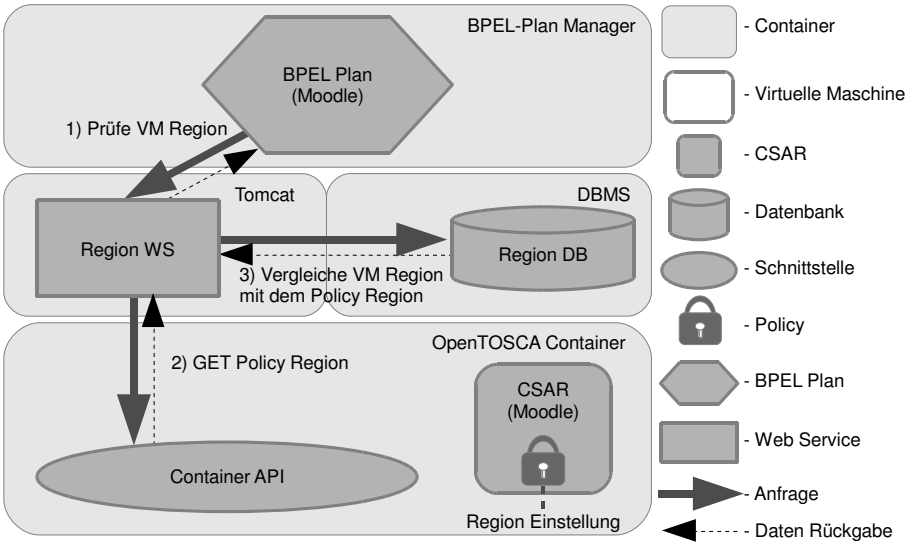


**Abb. 2.** Der Region-Policy Webservice prüft die Zulässigkeit einer Region für das Deployment der Anwendung.

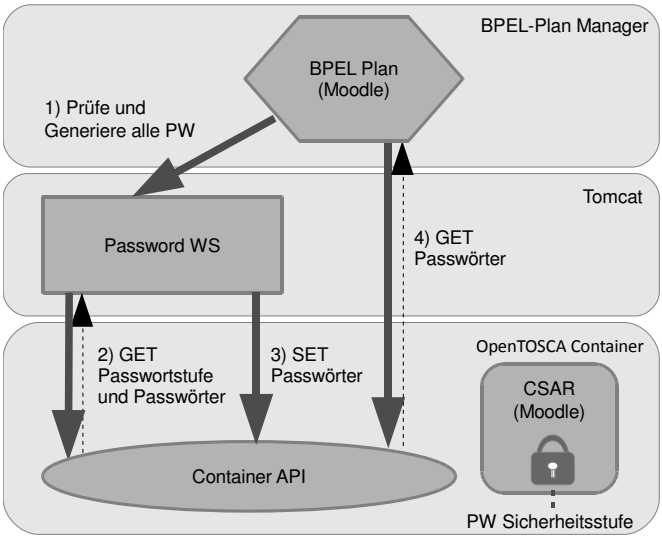


**Abb. 3.** Der Passwort-Policy Webservice erzeugt sichere Passwörter für die Anwendung.

### 3.2 Absicherung der Passwortstärke durch die Passwort-Policy

Passwörter sind im Bereich von Cloud-Diensten wichtig, weil initiale Passwörter oft trivial, oder Standardpasswörter wie „passwort“ sind. Solche Passwörter können heutzutage schnell und automatisiert erraten werden, was ein klares Sicherheitsrisiko darstellt [Eal03]. Die Passwort-Policy kann für bestimmte Teile eines Cloud-Dienstes Passwörter generieren, falls sie an dem Cloud-Dienst annotiert ist. Die Passwörter werden gemäß vordefinierter Sicherheitsstufen und Parameter erstellt. Auch hierfür wurde ein Webservice entwickelt, welcher diverse Methoden zur Generierung von Passwörtern anbietet. Wie dieser analog mit einer Anwendung funktioniert, sieht man in folgender Abbildung 3.

### 3.3 Energieeinsparung durch die Green-Policy

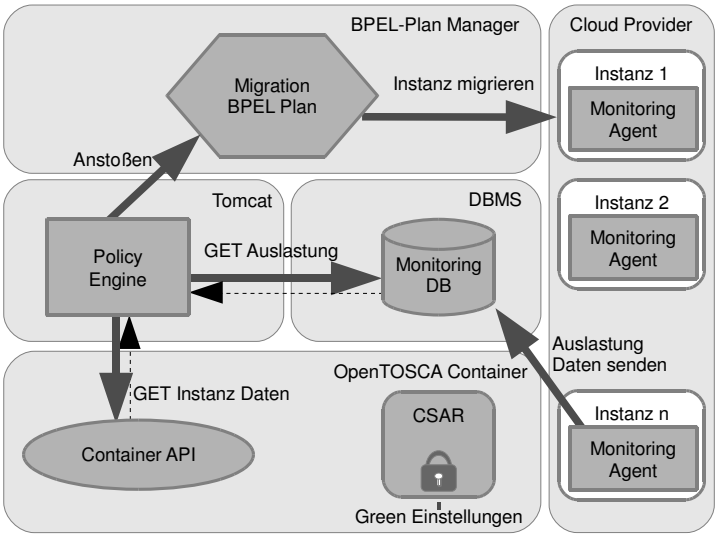


**Abb. 4.** Die Policy Engine stößt Migrationen nach Analyse gesammelter Instanz-Daten an.

Cloud-Dienste, deren Anwendungen nur sehr wenig Rechenkapazität von einer VM fordern, unterlasten die VM. Die Partei, die für die VM zahlt, muss höhere Kosten als nötig aufbringen, und die Partei, die die VM bereitstellt, hat höhere Stromkosten als nötig und ist so auch weniger umweltfreundlich. Umgekehrt kann es auch sein, dass ständig überlastete VMs der Forderung nach Rechenkapazität des Cloud-Dienstes nicht mehr gerecht werden und die Leistung des Cloud-Dienstes abnimmt. In beiden Fällen lässt sich das Problem lösen, wenn man für unterlastete VMs eine kleinere, leistungsschwächere VM wählt und für überlastete VMs eine größere und leistungsstärkere.

Die CPU-, Arbeitsspeicher- und Festplattengrößen sind vom Cloud-Provider abhängig. Bei der prototypischen Green-Policy Implementierung wurde Amazon als Cloud-Provider genutzt. Amazon gibt vordefinierte Größen für VMs an, die in allen der vorher genannten Werte mit jeder rechenstärkeren Größe auch entsprechend steigen. Wir

bezeichnen diese Kombination im folgenden als VM-Größe. Mit der Green-Policy hat der Cloud-Anbieter durch Angabe einer Liste von VM-Typen (sortiert nach ihrer Rechenleistung) die Möglichkeit automatisch auf nächstkleinere VMs bei Unterlastung und nächstgrößere VMs bei Überlastung zu wechseln. Dieser Vorgang wird hier auch Skalieren genannt, also das Anpassen der VM-Größe an die Anforderung des Cloud-Dienstes. Der Prozess des Wechselns und Kopierens aller Daten auf eine andere VM wird Migration genannt und ist in Abbildung 4 im Zusammenspiel der Green-Policy-Komponenten dargestellt. Die Green-Policy besteht aus einem Web-Service, der den Namen Policy Engine trägt. Er speichert Lastdaten seiner Instanzen in einer Datenbank und stößt mit Durchschnittswerten zu bestimmten Zeitperioden Migrationen an, wenn diese nötig sind. Folglich hat jede Instanz eines Cloud-Dienstes, an dem eine Green-Policy annotiert ist, ein Überwachungsprogramm installiert, das Monitoring Agent genannt wird und Lastdaten an die zentrale Datenbank der Policy Engine schickt.

### 3.4 Absicherung der gespeicherten Daten durch die Verschlüsselungs-Policy

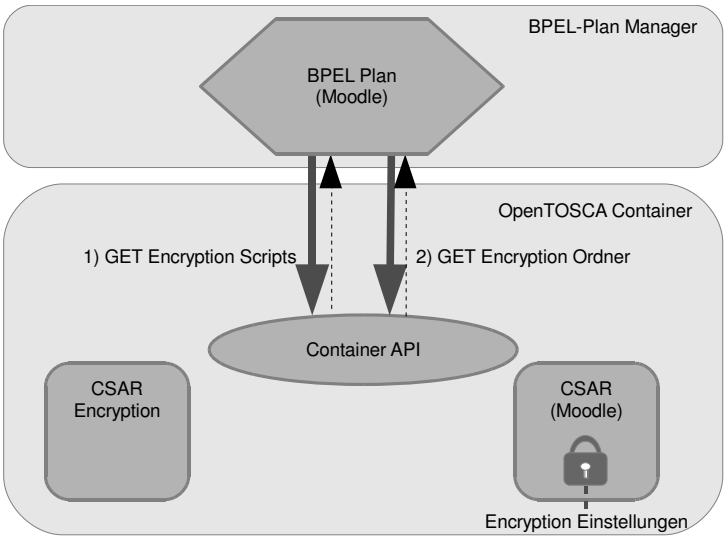


**Abb. 5.** Die Encryption CSAR beinhaltet alle Skripte, die den Encrpytion Ordner einer zu installierenden Anwendung verschlüsseln.

Wenn eine Datenbank oder ein Ordner mit sensiblen Daten eines Cloud-Dienstes unverschlüsselt auf einer VM liegt, so sind diese Daten vor Datendiebstahl nicht geschützt. Ein Beispiel für so einen Diebstahl ist das Kopieren der Daten von der VM auf ein Massenspeichergerät. Da die Daten unverschlüsselt sind, lassen sie sich im Klartext auf jedem anderen Computer lesen. Das Ziel der Verschlüsselungs-Policy ist es, sensible Daten einer Anwendung zu verschlüsseln. Der entstandene Schaden bei oben genanntem Diebstahl-Szenario ist dann weniger kritisch, da sensible Daten dann nicht mehr gelesen werden können. Die Verschlüsselungs-Policy nutzt das Programm

Ecrypt-FS [ecr], um Ordner zu verschlüsseln. Dies geschieht beim Deployment automatisch, wenn am Cloud-Dienst eine Verschlüsselungs-Policy annotiert ist. Der zu verschlüsselnde Ordner wird dabei als Parameter der Policy angegeben. Abbildung 5 beschreibt den Aufbau.

## 4 Methode zur Integration von Policies in Cloud-Dienste

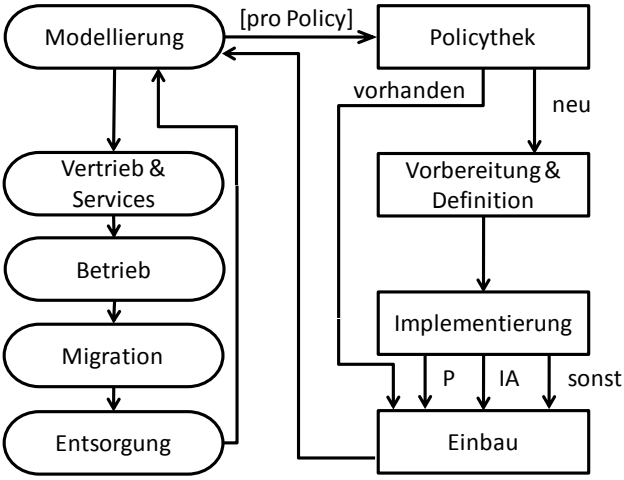


**Abb. 6.** Der Cloud-Cycle (links) und der Policy-Cycle (rechts) mit der Policythek

In diesem Kapitel wird eine allgemeine Methode zur Erstellung neuer Policies für die Realisierung nicht-funktionaler Anforderungen und zum Einsatz bestehender Policies in Cloud-Diensten beschrieben. Die Methode besteht aus vier Schritten, welche sich am Lebenszyklus einer Policy orientieren. Der Lebenszyklus von Policies (Policy-Cycle) baut wiederrum auf dem Lebenszyklus des zugehörigen Cloud-Dienstes (Cloud-Cycle) auf. Dies ist in Abbildung 6 dargestellt. Links sieht man den Cloud-Cycle, rechts den Policy-Cycle für eine annotierte Policy.

Der Policy-Cycle beginnt mit der Überprüfung, ob eine passende Policy nicht schon in der Policythek existiert (Schritt 1 siehe Kapitel 4.1), die man nutzen und sofort einbauen könnte. Wenn noch keine Policy existiert, muss eine entsprechende Policy zuerst vorbereitet und definiert werden (Schritt 2 siehe Kapitel 4.2). Anschließend wird in Kapitel 4.3 (Schritt 3) gezeigt, was mindestens nötig ist, um eine Policy zu implementieren und welche verschiedenen Möglichkeiten zur Implementierung existieren. In Kapitel 4.4 wird dann der Einbau einer Policy in einen Cloud-Dienst genauer beschrieben (Schritt 4).

### 4.1 Schritt 1: Policythek – Auswahl einer Policy

Im Rahmen des Policy-Frameworks wurde die Policythek erstellt. Sie zeigt die Verfügbarkeit der wichtigsten Bestandteile des Policy-Frameworks: die Policies. Hier kann eingesehen werden, ob Policies in einer OpenTOSCA Laufzeitumgebung installiert sind.

Im Policy-Framework wurden Policy-Pakete zur Installation von Policies definiert, die eine ähnliche Struktur wie CSAR-Pakete haben. Diese Pakete werden wie jedes CSAR-Paket in OpenTOSCA hochgeladen und automatisch als Policy-Paket erkannt. Danach erscheint die Policy in der Policythek mit ihrem Logo. Durch einen Klick auf das Logo öffnet sich eine Kurzbeschreibung der Policy. Der entscheidende Vorteil des Policy-Frameworks ist es, dass Policies mit ihren Webservices und anderen Daten nun an einer zentralen Stelle in der OpenTOSCA Umgebung liegen, von der aus jeder Cloud-Dienst dieser Umgebung Zugriff darauf hat. Wenn hier eine Policy existiert, die von einem Cloud-Dienst benötigt wird, so kann man diese einbauen, wie in Schritt 4 erklärt wird. Wenn keine passende Policy existiert, muss eine neue erstellt werden, wie in Schritt 2 beschrieben wird.

### 4.2 Schritt 2: Vorbereitungen und Definitionen einer neuen Policy

Will man eine Policy mithilfe des Policy-Frameworks umsetzen, so muss man zuerst folgende Überlegung anstellen: Ist die umzusetzende Policy generisch, sprich, kann die Policy ohne große Änderungen an der Policy Implementierung bei verschiedenen Cloud-Diensten annotiert und eingesetzt werden? Trifft dies nicht zu, so ist eine Implementierung über das Policy-Framework nicht anzuraten, sondern es sollte eine individuelle Implementierung für den jeweiligen Cloud-Dienst erfolgen. Bei generisch einsetzbaren Policies geht man unter Berücksichtigung folgender Kriterien vor:

- Spezifikation der Policy: Welche genaue Funktion soll die Policy umsetzen? Hierbei sollten textuelle Beschreibungen für die Entwickler und für die Endanwender, sowie eine Spezifikation erstellt werden. Zudem wird ein Logo für die Policy benötigt, damit diese in der Policythek angezeigt werden kann.

- Entwurf der Policy: Welche Funktionalitäten müssen von dem zu erstellenden Webservice und möglichen weiteren Programmen und Scripten geboten werden, um die Funktion der Policy umzusetzen?

- Architektur der Policy: Welche Dateien werden für die Umsetzung dieser Policy benötigt? Dazu gehören TOSCA Policy-Definitionen, Webservices, Scripte, Programme, Datenbanken und Ähnliches.

### 4.3 Schritt 3: Implementierung der benötigten Systemteile

Sind alle Vorüberlegungen geklärt, so können alle benötigten Dateien und Dokumente erstellt werden. Sind diese Dateien und Dokumente erstellt worden, so können diese

in ein Policy-Paket gepackt werden, welches eine spezielle Form eines CSAR-Paketes darstellt. Ein solches Policy-Paket enthält folgende Dateien, die für die Umsetzung der Policy nötig sind:

- den Ordner Definitions mit allen nötigen .xml Dateien
- den Ordner IAs mit eventuellen Webservices, wobei IA hier für Implementation Artifact steht
- den Ordner Imports. Er enthält die zu den Webservices gehörenden .wsdl-Dateien
- den Ordner TOSCA-Metadata mit der Datei TOSCA.meta
- den Ordner POLICY-Metadata, in welchem die Dateien für die Policy-spezifische Webseite der Policythek liegen. Darin ist mindestens eine Datei namens data.xml einzufügen mit einer Beschreibung der Policy und einer Referenz auf das Icon. Der Aufbau einer beispielhaften data.xml kann hier abgerufen werden: http://pastebin.com/1d8tsZvp
- den Ordner Scripts mit optionalen Scripten.

Alle diese Dateien werden in einem Ordner gespeichert und als .csar Datei gepackt. Dieses Paket kann nun als Policy-Paket in OpenTOSCA hochgeladen werden. Sobald es in der Policythek erscheint, ist es erfolgreich installiert worden und die Policy kann von nun an von allen Cloud-Diensten dieser OpenTOSCA-Laufzeitumgebung genutzt werden.

### 4.4 Schritt 4: Einbau der Policy in ein CSAR-Paket mittels Annotation

An dieser Stelle wurde ein Policy-Paket erfolgreich als Policy in einer OpenTOSCA Laufzeitumgebung installiert oder existiert dort bereits. Nun muss man Änderungen am Gegenstück vornehmen: dem Cloud-Dienst selbst, genauer gesagt am CSAR-Paket, das den Cloud-Dienst beschreibt. Hier muss die Policy am entsprechenden Service-Template oder Node-Type annotiert werden. Dies geschieht durch die Nutzung des TOSCA-Policy-Tags. Dies ist nötig, damit der Cloud-Dienst beispielsweise den Webservice ansprechen kann, den eine Policy bietet. Hier unterscheidet man je nach Implementierung des Webservices nach den zwei in [WWB+13] vorgestellten Ansätzen:

Der erste Ansatz ist der IA-Approach. Dieser ermöglicht es dem Webservice, mit der OpenTOSCA Container-API zu kommunizieren und automatisch die Struktur eines Cloud-Dienstes nach Stichwörtern zu durchsuchen. Wir stellen dies am Beispiel einer Passwort-Policy mit entsprechendem Webservice vor. Der Webservice prüft automatisch bei allen nötigen Elementen, ob ein neues Passwort zu generieren ist und generiert dieses. Dies hat den Vorteil, dass der CSAR-Ersteller, der auch Lösungspaketersteller genannt wird, nicht für jedes Element, das ein Passwort benötigt eine

Abfrage in den BPEL-Plan setzt, sondern er muss nur einen generischen Webservice Aufruf in den Workflow einfügen (Beispiel siehe http://pastebin.com/pWcAcZ5F). Allerdings muss der Lösungspakethersteller darauf achten, dass er eine Namenskonvention einhält für alle Elemente, die ein Passwort enthalten, damit der Passwort-Webservice diese auch als solche erkennt - beispielsweise durch die Konvention jedem Knoten, der ein sicheres Passwort haben soll, das Wort passwort anzuhängen.

Der zweite Ansatz ist der P-Approach, oder Plan-Approach, bei dem keine Kommunikation des Web-Services mit OpenTOSCA stattfindet. Das hat zur Folge, dass der Lösungspakethersteller genau für jedes seiner Elemente, die ein sicheres Passwort brauchen, im BPEL-Plan eine separate Anfrage an den Webservice der Policy schickt, sich eine zweite Anfrage zur Passwort-Generierung baut und ein Passwort generieren lässt. Das muss er für jedes seiner Elemente machen, das so ein sicheres Passwort braucht. Ein Beispiel für die erste Anfrage wird in http://pastebin.com/kj6U4CM3 dargestellt.

## 5 Evaluation

Zur Evaluation wurden die Laufzeiten zur Instanziierung zweier Anwendungen gemessen. Die Moodle Anwendung wurde mit der Passwort Policy, der Region Policy und einmal ohne Policy gemessen. Die SimpleBlog Anwendung wurde einmal mit der Encryption Policy und einmal ohne Policy gemessen. Alle Messungen wurden auf der gleichen OpenStack Infrastruktur (Ein Management und ein Hypervisor Server, je 12 Intel Xeon E5-2630 CPUs bei 2,3GHz, 256GB Speicher) durchgeführt und für jede Anwendung wurden die gleichen Instanzgrößen gewählt (8 vCPUs, 16GB Hauptspeicher, 20GB Festplatte). Die Verschlüsselung-Policy verändert die Instanz des Cloud-Dienstes, wohingegen die Passwort- und Region-Policy ihn lediglich umkonfigurieren. Es hat sich gezeigt, dass die Passwort- und Region-Policy die Instanziierungszeit des Cloud-Dienstes nicht verlängern (siehe Abbildung 8). Dies folgt daraus, dass die Policies bereits in der Policythek vorinstalliert sind und den Großteil der Laufzeit des Deployments die Instanzerstellung der VMs auf der verwendeten OpenStack-Plattform ausmacht. Diese Laufzeit hängt darum vor allem von der aktuellen Auslastung der OpenStack-Infrastruktur ab. Sie wird nur vernachlässigbar von den Policies beeinflusst.

Einen Unterschied stellt die Verschlüsselungs-Policy dar (siehe Abbildung 9). Diese muss bei jeder Instanziierung für jeden zu verschlüsselnden Ordner ausgeführt werden, wodurch sich die Laufzeit entsprechend verlängert. Auch hier sind die Laufzeitverlängerungen durch Annotation der Policy vernachlässigbar, da die Instanziierung des Cloud-Dienstes selbst den Großteil der Laufzeit ausmacht.

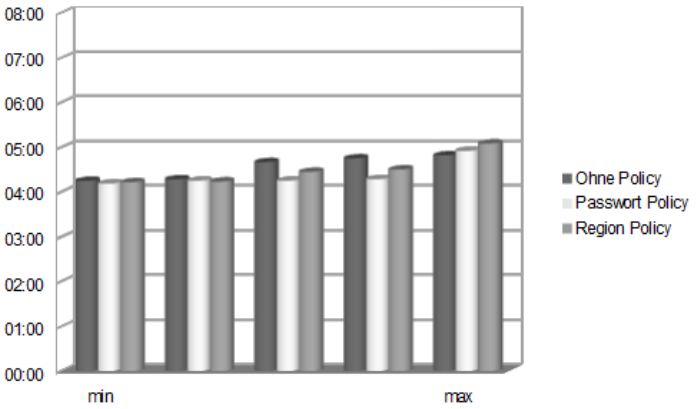


**Abb. 7.** Laufzeiten einer Instanziierung einer Moodle-Anwendung ohne Policy (dunkelgrau) mit Passwort-Policy (hellgrau) und mit Region-Policy (grau)

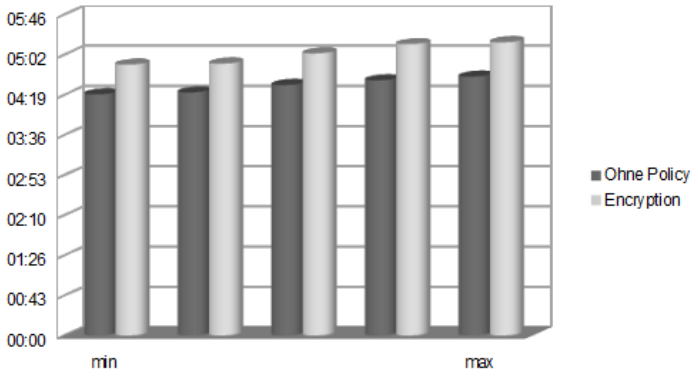


**Fig. 8.** Laufzeiten einer Instanziierung einer SimpleBlog-Anwendung. Ohne Policy (dunkelgrau) und mit Verschlüsselungs-Policy (hellgrau).

## 6 Zusammenfassung und Ausblick

Das Policy-Framework wurde entwickelt, um zu untersuchen, wie Policies generell in Cloud-Dienste integriert werden können. Es wurde erreicht die Policies zentral zu verwalten, ihren Aufbau möglichst generisch zu beschreiben und sie von der Entwicklung der Cloud-Dienste abzugrenzen. Dies wird durch die vorgestellten Prototypen aufgezeigt. Des Weiteren ist zu beachten, dass die Ausführung der Policies aufseiten des Cloud-Anbieters geschieht, sodass zwischen Kunde und Cloud-Anbieter ein Vertrauensverhältnis erforderlich ist, um sicherzustellen, dass die Policies eingehalten werden. Um dies zu unterstützen, sollte ein umfassendes Logging aufseiten des

Cloud-Anbieters durchgeführt werden, welches eine Nachvollziehbarkeit der ausgeführten Prozesse erlaubt.

In zukünftigen Arbeiten könnten, analog zur Standardisierung von TOSCA-Node-Types, standardisierte TOSCA-Policies definiert werden, sodass Entwickler neuer Node-Types und Dienste deren Unterstützung bereits vorsehen können. Dies hätte den Vorteil, dass die standardisierten Policies in jeder TOSCA-Laufzeitumgebung verfügbar wären und somit nicht installiert werden müssten und dadurch bei der vorgestellten Integrations-Methode der erste Schritt entfallen könnte.

**Danksagung:** Die Autoren danken dem Bundesministerium für Wirtschaft (BMWi) für die Förderung im Rahmen des Migrate! (01ME11055) und CloudCycle (01MD11023) Projekts. Die Autoren danken auch allen Teilnehmern, Betreuern und Kunden des Studienprojekts „Policy-Framework“ für die Zusammenarbeit.

## Literaturverzeichnis

[BBH+13] T. Binz et. al. OpenTOSCA - A Runtime for TOSCA-based Cloud Applications. In Proceedings of 11th International Conference on Service-Oriented Computing *(ICSOC'13)*, volume 8274 of LNCS, S. 692–695. Springer Berlin Heidelberg, Dezember 2013.

[BBK+12] U. Breitenbücher et.al. Vino4TOSCA: A Visual Notation for Application Topologies based on TOSCA. In Proceedings of the 20th International Conference on Cooperative Information Systems (CoopIS 2012), Lecture Notes in Computer Science. Springer-Verlag, September 2012.

[Eal03] K. Ealy. A New Evolution in Hack Attacks: A General Overview of Types, Methods, Tools, and Prevention. S. 8-9, 2003.

[ecr] Ecrypt FS Homepage . http://wiki.ubuntuusers.de/ecryptfs.

[fSidI] Bundesamt für Sicherheit in der Informationstechnik. Konzeption von Sicherheitsgateways., https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Internet sicherheit/Konz_SiGw_pdf.pdf?__blob=publicationFile. Version 1.0S. 16.

[KBBL13] O. Kopp et. al. Winery - A Modeling Tool for TOSCA-based Cloud Applications. In Proceedings of 11th International Conference on Service-Oriented Computing *(ICSOC'13)*, volume 8274 of LNCS, S. 700–704. Springer Berlin Heidelberg, Dezember 2013.

[Kni13] P. Knight. Topology and Orchestration Specification for Cloud Applications Version 1.0., November 2013.

[MSM08] M. Messerschmidt, P. Schülein, und M. Murnleitner. Der Wertbeitrag der IT zum Unternehmenserfolg. S. 57, 2008.

[WWB+13] T. Waizenegger et. al. Policy4tosca: A policy-aware cloud service provisioning approach to enable secure cloud computing. On the Move to Meaningful Internet Systems: OTM 2013 Conferences, volume 8185 of LNCS, pages 360–376. Springer Berlin Heidelberg, 2013.