Playing in the Vault

Dr. Marcus Holthaus, OpenCloud Day 2018

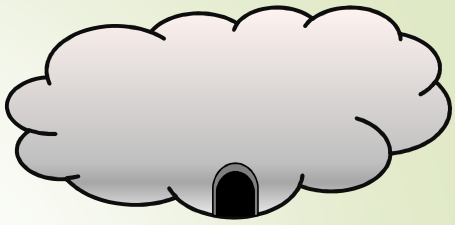
Playing in the Vault Overview

- Motivation & Classic Model
- Setup
- Requirements
- Approach & Indirect Model
- Restrictions
- Clouds
- Conclusion

Teaser

- Keeping secrets to yourself is inherently difficult when putting them onto someone else's computer. So the cloud providers and their technology need to be instructed in detail on how to handle those secrets, and you yourself must take care in configuring these instructions. Also there must be automatic authentication of those requesting the secrets, and there needs to be some interaction with on-site and local solutions. This talk gives an overview - not too technical, but sufficiently to get you started in a secure direction.
- The referent Marcus Holthaus has been doing security improvements of technical systems for decades as a consultant and as a security architect, he talks to management and to implementers and supports them in keeping the bad guys out and the secrets in.

Motivation



- *Motto: Each cloud needs a vault*
- Original Real-World Use case:
  - A Database in the cloud – for „big“ data analysis
  - Storage in one of the standard (classic SQL-based) database services provided by the standard clouds
  - Evaluation of which cloud is best fit for the task, which DBMS etc.
  - Evaluation of which cloud is best fit to keep the secrets – the database is not public, and the connection string must be guarded
  - Several applications need access
  - Applications shall be able to scale (multiple instances)
- **Challenges for adaption of security**
  - Patterns unknown or unfamiliar
  - Services unknown, or unfamiliar, or uncommon, or assumed quality problems
  - No „toolset“
- **Someone has to show it does work.**
  - Security Toolchain

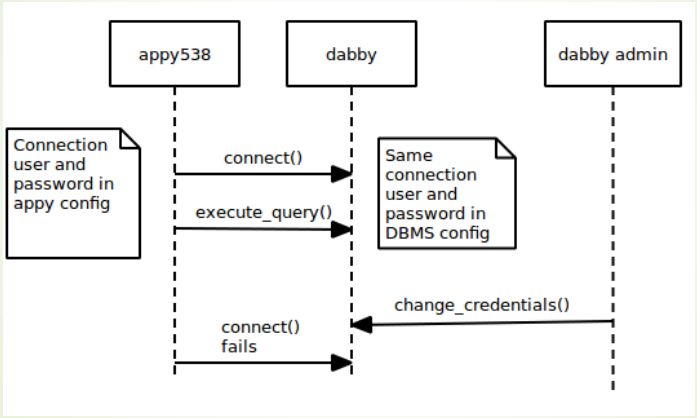
Requirements

- Data is confidential
- No secrets in source code
- Some secrets in app configuration files are OK, but need to be "well-protected"
- DB access secrets must be sharable among several apps
- All secrets need to be exchangeable / updateable with little impact and independently of each other
- Users (data providers) need to be able to dynamically add data with new secrets, i.e new tables with new encryptions

Early Conclusions

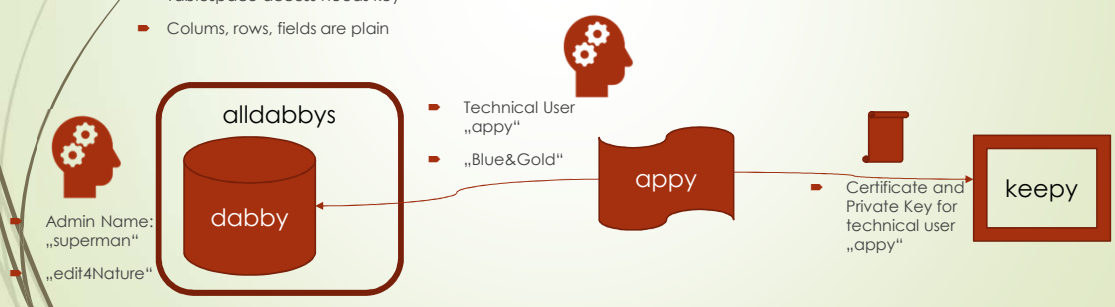
- **The big clouds are stable. They work reliably.**
- All cloud providers offer sufficient database performance and features – it's a price tag question related to usage scenarios and previous investment
- All cloud providers offer some sort of secret guarding service("vaults")
- All offer data encryption
  - On storage level (transparent)
  - On other levels, mostly leveraging the respecitve DBMS's features
    - Database, table space (tables), columns, records (rows), fields
- No default security pattern for credential protection
- **Configuration is a challenge!**
- **Automation is a challenge!**
- ***Applied* security is a challenge!**

Classic Authentication Model



Evaluation Model Setup

- A database inside a standard DBMS inside a storage allocation inside a cloud subscription
- Storage encryption (transparent, out of scope)
- Table space encryption, i.e
  - Db access is plain
  - Tablespace access needs key
  - Colums, rows, fields are plain
- An application program needing to access the database.
- Connection using classic "connection string" (i.e. technical user and passphrase)
- Table space access requiring key materials
- A storage environment which can handle secrets
- "Decent" authentication

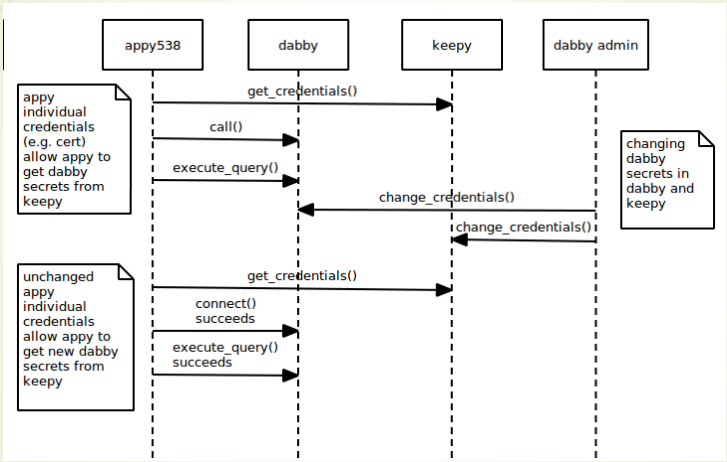


- Names are inspired by „Die Tyrannei des Schmetterlings“ by Frank Schätzing, 2018, and by „Superman“

Approach

- **Need to be explicit in secret keeping**. Why?
  - Possibly wide access to standard storage services as in typical use by applications
  - Unclear, or optional, or intransparently implemented encryption of standard storages; inappropriate for secrets
  - Separation of secrets from standard data; they are two different kinds of data; e.g. backup, duplication
- **Implement „Credential Indirection" pattern**
  - Decoupling authentication credentials from individual applications' and resource's lifecycles
- Sequence:
  - Select potential cloud providers
  - Setup databases inside each cloud provider's environment using as much standardized functionality as necessary with required features
    - one database, with connection string
    - Two tables, each encrypted with different keys
    - minimal test data
  - Write a minimally viable product to access this
  - **Change the access password and both keys**
  - Try again

Sequence Model with Indirection



This is a Standard Challenge
From «Secret-Verwaltung mit Cloud KMS»

https://cloud.google.com/kms/docs/secret-management

- «**Warnung**: Eine andere gebräuchliche Option besteht darin, Ihre Secrets direkt im Code zu speichern. Diese Option ist nicht ratsam. Obwohl dies die einfachste zu implementierende Lösung ist, ermöglicht sie jedem, der Zugriff auf Ihren Code hat, auch Zugriff auf Ihre Secrets. Dadurch sind Sie anfällig für Angriffe von innerhalb und außerhalb Ihrer Organisation. Diese Sicherheitslücke kann bestenfalls dazu führen, dass Ihre Daten und Konten an anderer Stelle missbraucht werden. Im schlimmsten Fall könnte der Angreifer sogar Zugriff auf noch mehr Daten erhalten.»

This is a Standard Case
From «Protecting Sensitive Data with Azure Key Vault»

https://blogs.msdn.microsoft.com/data_insights_global_practice/2015/09/24/protecting-sensitive-data-with-azure-key-vault/

- *"[…] why not just store the client's Twitter parameters directly in the Web App runtime configuration and avoid all of the indirection?* The logic here is that the Twitter authentication parameters are valuable customer resources that may be used in multiple contexts, and the Key Vault provides a secure way to manage these resources on behalf of any Azure component that is authorized to make use of them. The Key Vault credentials are different: these are application-specific resources so it makes sense to provision them in the authenticated context provided by the Azure portal for each managed application.
- One of the key security principals that is implicitly being applied here is to compartmentalize management of privileged data to security domains for which this is appropriate. An instance of Key Vault is used to manage the Twitter keys as a shared resource in the customer's environment, with access granted by whomever manages the Twitter account on an as-needed basis to specific applications and users. Applications are then responsible for managing only their application-specific Key Vault access tokens."

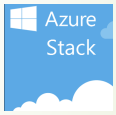
Restrictions

- **There will always be "some" secret**
  - How to protect secrets in code is an entirely different and complex topic
  - Possible case: Have someone unlock at startup and keep it plaintext in memory
- **Key Vaults**
  - May use hardware security modules (HSM) for cryptographic key generation and storage
  - May (but probably will not) use HSMs for secret storage
- **Extra effort required**
- **Additional communication links and dependencies**
- **Additional service cost**

- Also:
  - Cross-Cloud-implementations technically doable but difficult to keep stable

Potential Cloud Providers

- The standard bunch
  - Microsoft Azure
  - Amazon Web Services
  - Google Cloud Platform
- The "private" ones
  - Microsoft Azure Stack
  - OpenStack (at RackSpace)



Azure Services (Excerpt)

IDENTITY (12)

Azure Active Directory	★	Azure AD B2C	☆
Azure AD Domain Services	☆	Azure Information Protection	☆
Groups	☆	Users	☆
Azure AD Connect Health	☆	Azure AD Privileged Identity Management	☆
Azure AD Identity Protection	☆	Enterprise applications	☆
App registrations	☆	Access reviews	☆

SECURITY (6)

Security Center	★	Key vaults	☆
Application gateways	☆	Azure Information Protection	☆
Virtual network gateways	☆	Azure Active Directory	★

AWS Services (Excerpt)

Amazon Glacier
AWS Storage Gateway
AWS Snowball
AWS Snowball Edge
AWS Snowmobile

Datenbank
Amazon Aurora
Amazon RDS
Amazon DynamoDB
Amazon ElastiCache
Amazon Redshift
Amazon Neptune
AWS Database Migration Service

Migration
AWS Migration Hub
AWS Application Discovery Service
AWS Database Migration Service
AWS Server Migration Service
AWS Snowball
AWS Snowball Edge
AWS Snowmobile

Verwaltungs-Tools
Amazon CloudWatch
AWS Auto Scaling
AWS CloudFormation
AWS CloudTrail
AWS Config
AWS OpsWorks
AWS Service Catalog
AWS Systems Manager
AWS Trusted Advisor
AWS Personal Health Dashboard
AWS-Befehlszeilenschnittstelle
AWS Management Console
AWS Managed Services

Medienservices
Amazon Elastic Transcoder
Amazon Kinesis Video Streams
AWS Elemental MediaConvert
AWS Elemental MediaLive
AWS Elemental MediaPackage
AWS Elemental MediaStore
AWS Elemental MediaTailor

Amazon Kinesis
Amazon Redshift
Amazon QuickSight
AWS Data Pipeline
AWS Glue

Sicherheit, Identität und Compliance
AWS Identity and Access Management (IAM)
Amazon Cloud Directory
Amazon Cognito
Amazon GuardDuty
Amazon Inspector
Amazon Macie
AWS Certificate Manager
AWS CloudHSM
AWS Directory Service
AWS Firewall Manager
AWS Key Management Service
AWS Organizations
AWS Secrets Manager
AWS Single Sign-On
AWS Shield
AWS WAF
AWS Artifact

Services für Mobilgeräte
AWS Mobile Hub
Amazon API Gateway
Amazon Pinpoint

Desktop- und App-Streaming
Amazon WorkSpaces
Amazon AppStream 2.0

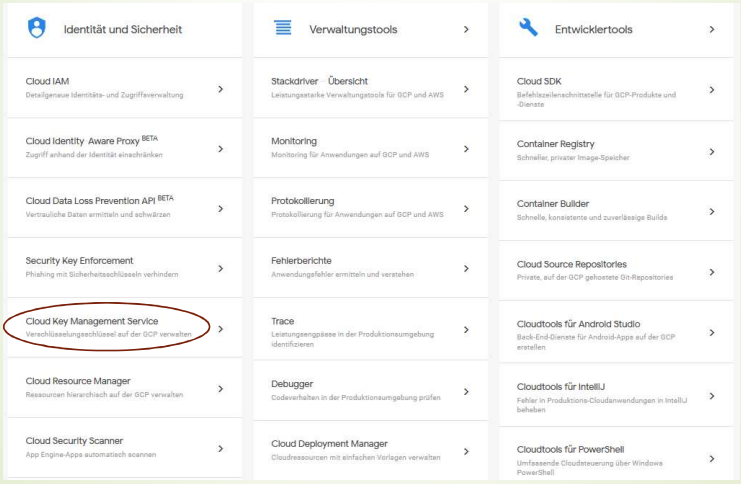
Internet of Things
AWS IoT Core
Amazon FreeRTOS
AWS Greengrass
AWS IoT 1-Click
AWS IoT Analytics
AWS IoT Button
AWS IoT Device Defender
AWS IoT Device Management

Entwicklung von Spielen
Amazon GameLift
Amazon Lumberyard

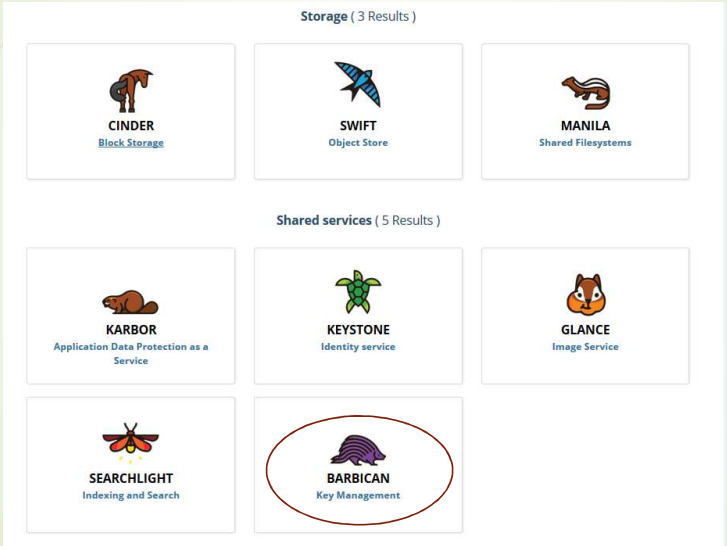
Software
AWS Marketplace

AWS-Kostenmanagement
AWS Cost Explorer
AWS-Budgets
Reserved Instance Reporting
AWS-Kosten- und Nutzungsbericht

Google Cloud Platform Services (Excerpt)



OpenStack Services (Excerpt)



Azure Stack Services (Excerpt)
«Azure Stack is an extension of Azure»

- Container Registry
- Container Service
- Cosmos DB
- Data Lake Analytics
- Data Lake Store
- Devtest Lab
- Dms
- Eventgrid
- Eventhubs
- Extension
- Functions
- Iot
- Key Vault
- Locks
- Maps
- Monitoring
- Mysql

- Networking
- Postgresql
- Redis Cache
- Reservations
- Resource Groups
- Resource Policies
- Resource Provider Features
- Resource Providers
- Resource Tags
- Resources
- Roles
- SQL
- Service Fabric
- Servicebus
- Storage
- Subscriptions
- VM Scale Sets

Features

Feature	Azure	aws	GCP	Azure Stack	OpenStack
Storage Encryption	Yes (transparent)	Optional with S3	Yes, with storage key	Yes (Bitlocker)	Volume encryption
Postgres SQL DBMS	Azure Postgres	Amazon Aurora PostgreSQL	Google Cloud SQL für PostgreSQL	(no Postgres) SQL Resource Provider	Trove based on MySQL; postgres custom integration
SQL DB with connection string authentication	Yes TLS	Yes (or token) TLS optional	Yes, standard PostgreSQL roles	yes	Yes
SQL DB Tablespace encryption	Standard: no In VM: yes	With VPC security group	With extension or in storage encryption	No, only by storage encryption	Yes
Firewall rules	Mandatory	With VPC and Amazon RDS	Global firewalls attached to specific networks	No, only external	FWaaS
Secret Store	Azure Vault	• AWS CloudHSM • AWS Key Management Service KMS • AWS Secrets Manager • AWS Certificate Manager	Google Cloud Key Management Service (KMS)	Key Vault in Azure Stack	castellan and Barbican, with plug-ins
Secret Store Key Storage	Yes, "Key Store"		encryption with restricted key, and encrypted secret widely available	Yes, "Key Store"	
Secret Store Connection String Storage	Yes, "Secret Store"			Yes, "Secret Store"	(depends on plugin)
Secret Store Authentication	Various, e.g. certificates	Various, e.g. certificates	Various, e.g. certificates	Various, e.g. certificates	„Pending“
Remarks	DBMS admin credentials (!)				

Conclusion

- **Yes it works**
- **Architecture applicable to several cloud environments**
- **Requirements implementable**
  - Selective proof of concept conducted with **Azure**
  - Full sample source code available in Azure Blog
- There is plenty of relevant services, documentation, know-how and security patterns / frameworks
- But
  - Some clouds react slowly (probably depending on subscription)
  - Azure Keyvault backup lirritations – or features.
  - OpenStack modularity gives great flexibility but also increases initial effort for non-mainstream solutions (such as this still is)
  - Google Secret Store concept required slightly changed architecture
  - Quite some effort to get all the calls right – the portal eases some of that and there is good documentation to allow to determine correct syntax
- **Being consequent binds you to the actual cloud provider**
  - Similar architectural concepts
  - Different APIs
  - Different „session“ principles

The Messages

- **Use the cloud security features. They work. They help.**
- **Do security by design. Start early.**
- **Create standardized security building blocks.**
- **Try. Learn. Integrate. Use.**
- **Get advice, help, support.**

Value and the Vault

(Image showing Daniel Radcliffe as Harry Potter in the Vault of Bellatrix Lestrange at Gringotts Bank)
Image removed from presentation for copyright reasons

Source: http://harrypotter.wikia.com/wiki/File:Bellatrix_Lestrange%27s_Vault_DH2.jpg used under the Fair Use doctrin as explained in https://en.wikipedia.org/wiki/Fair_use#Fair_use_under_United_States_law