

CloudTrust

Digitalization, Security and “CASB”

Sébastien Pasche
Lead Architect Defense & CyberSecurity



Who am I

I am a computer security enthusiast

Security contests, Malware, Web architecture

Open source contributor since 2001

Volunteer in multiple organizations

Addict to absurd & abstract arts

Sebastien Pasche

Twitter @braoru

@ELCA since 01.2017

Architect @leshop 2012-2017

Freelance (mainly biomed) Architect 2006-2017



“The corollary of constant change is ignorance. This is not often talked about: we computer experts barely know what we’re doing. We’re good at fussing and figuring out. We function well in a sea of unknowns. Our experience has only prepared us to deal with confusion. A programmer who denies this is probably lying, or else is densely unaware of himself.”

Ellen Ullman, Close to the Machine: Technophilia and Its Discontents

Agenda

- A Few words on ELCA, Security and Digitalization
- Digitalization and what problems we are facing with our customers
- The CloudTrust project and it's current status with some fresh crunchy tools



We make it work.

Founded in 1968

Over 800
employees

Turnover of CHF
118.8 millions in
2016 (growth 11%)

1'000 customer
projects in ten
years

ELCA conducted many digital migrations related project

More than 100 (IDaaS, IAM , IGA, User Management Process) projects

Infrastructure migration

Developing private clouds for ELCA and customers

Partner of center for digital trust @ EPFL

Nowady, we have to open to everybody services we formely only opened internaly

We have to bridge legacy, mobile and so on ... with newly hosted services and make them run somewhere

Digitalization

Lets Focus on Data & Runtime environment

Digitalization & Dreams

Externalize Runtime and/or Operating platform

Local PAAS, Remote PAAS, Hybrid PAAS

Reduce your operational cost on OPEX and CAPEX

Pure SAAS Software, Remotely managed PAAS

Ability to change from one providers to another

No Vendor lock-in, Keep your own key approach

Don't care about availability problematics

Let providers admin deal with my almost running containers

Simplify our architecture

Try to move from an assembly of product to an unified approach

Our customer current dreams about Digitalization

No Vendor lock-in

Keep you own keys approach (KYOK)

Flexibility between on-premise and SAAS

Replace assembly of multiple **complex** Product by an unified approach

Expertise on integration and help with process and strategy design

Transparency on how it's working and what we are doing with data (can be audited anytime)

Support on **open source** software base

High **availability** on a large multi-cluster scale

THAT'S NOT HOW THIS WORKS



**THAT'S NOT HOW ANY OF
THIS WORKS**

What could go wrong



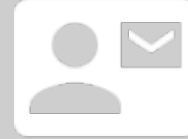
Data privacy

Ensure that sensible information is not disclosed



Access & location

Keep control over data access and its location



B2B collaboration

Allow secure collaboration with B2B while keeping lower costs



Compliance

Ensure compliance with legal regulations

What could go wrong



When you protect content with Azure RMS, Azure RMS uses a 2048-bit RSA asymmetric key with SHA-256 hash algorithm for integrity to encrypt the content. The symmetric key for Office documents and email is AES 128-bit (CBC mode with PKCS#7 padding).

In a default Azure RMS implementation, Microsoft generates and manages the root key that is unique for each tenant. Customers can manage the lifecycle of their root key in Azure RMS with SharePoint Online by using a method called [Bring your Own Key \(BYOK\)](#) that allows you to generate your key in on-premises HSMs, and stay in control of this key after transfer to Microsoft's FIPS 140-2 Level 2-validated HSMs. Access to the root key is always limited to Office 365 applications (such as Exchange Online and SharePoint Online) and is not given to any personnel. In addition, customers can access a near real-time log showing all access to the root key at any time. For more information, see [Logging and Analyzing Azure Rights Management Usage](#). Source : Data Encryption Technologies in Office 365

Shield Platform Encryption allows Salesforce administrators to manage the lifecycles of their data encryption keys while protecting the keys from unauthorized access. To ensure this level of protection, data encryption keys are never persisted on disk. Instead, they're derived on demand from the master and tenant secrets.

The master secret is generated by a master HSM at the start of each release. The master HSM is "air-gapped" from Salesforce's production network and stored securely in a bank safety deposit box. Only designated Salesforce security officers can access the safety deposit box and the master HSM stored within.

Business <-> Technical words translators

Access

SSO, ABAC, Access policy, MFA

Visibility, Compliance

Manage

B2E, B2B, B2C, Users provisioning,
Workflow, Branding

Visibility, Compliance, Identity management

Control

Shadow IT, License metering, behavior
analytics, reporting

Visibility, Compliance, Threats protection,
Cost optimization

Protect

Field Protection, Searchable
encryption, Proxy/Reverse

Threats protection, Data Loss Prevention,
Privacy

Business context

What is a CASB ?

Cloud Access Security Broker (CASB) is a visibility & control point between users of your organisation and the cloud



Visibility

who is using which app and which data is stored where



Threat protection

detects malware stored in the cloud and suspect behaviours



Data Loss Prevention

handle information according to its specificities (ciphering, tokenization)



Compliance

ensure compliance with specific industry regulations



CloudTrust

CloudTrust vision

OpenSource and Transparent

Support multiple deployment target

SAAS, Appliances, Customer hosted PAAS, Hybrid

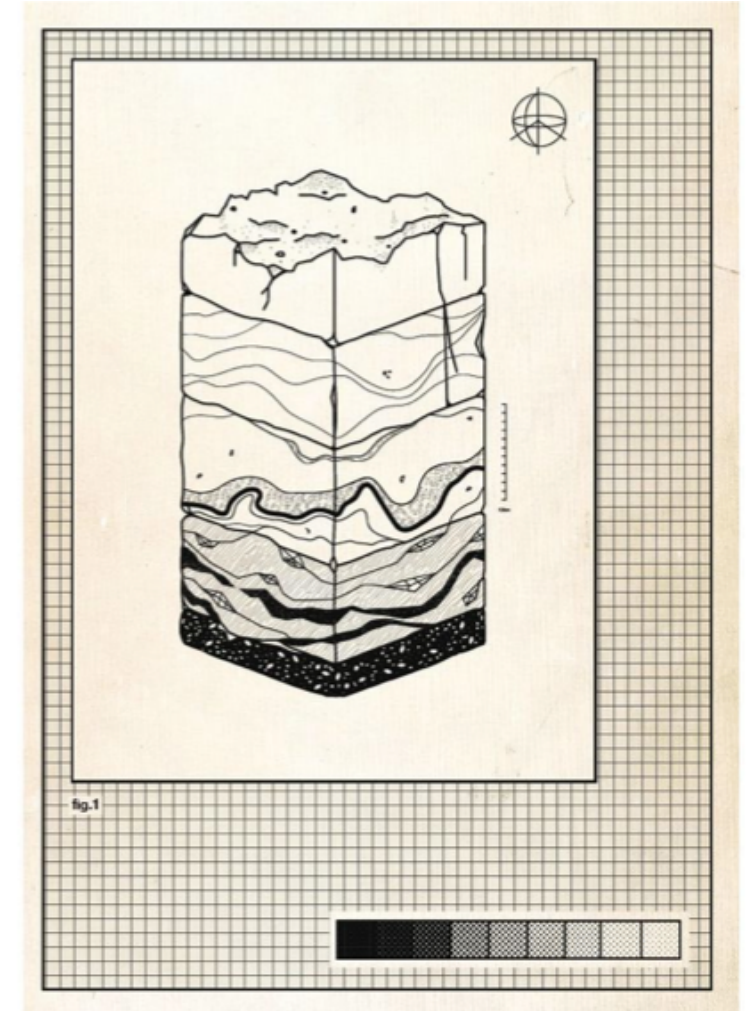
Organized as a toolkit with dedicated distribution

Fix gap between OpenSource product and our fields experiences

Keep OPEX stable

Multi tenancy, Use vanilla technologies,
Use/Improve open source, clustering.

Embark ELCA security knowledge and project experiences



Find the gaps then, fill the holes

Make Keycloak work with cockroach-db

<https://github.com/cloudtrust/keycloak-cockroach>

Create a module to send everything happened within keycloak

<https://github.com/cloudtrust/event-emitter>

Implement WSFED protocol within keycloak

<https://github.com/cloudtrust/keycloak-wsfed>

Keycloak Reactive programming interface & others missing features within a pet services

<https://github.com/cloudtrust/keycloak-bridge>

Format preserving encryption for go

<https://github.com/cloudtrust/fpe>

Unique and non colliding id generator : Flaki Das kleine Generator

<https://github.com/cloudtrust/fpe>

More project available

SRP go implementation

GRPC Load balancing

Hashicorp Vault pet services

...

More project to come soon

Keycloak OICD access control for SAML and WSFED (06.2018)

Sample java implementation of protect

...

Conclusion

Federating Access and identity management is the first step to secure your new generation of services

By design it's a very sensible and critical part

This talk focused only on 1 axes

We are doing a lot of work on the protect axis

Digitalization implies lot of security tradeoff

Don't hesitate to explore our GIT repositories

We plan to release a lot of other current works

We would like to create a communities around security and digitalization

Questions ?

This slide deck is available online
<https://github.com/cloudtrust/talks>

Access

Because without authentication there is nothing

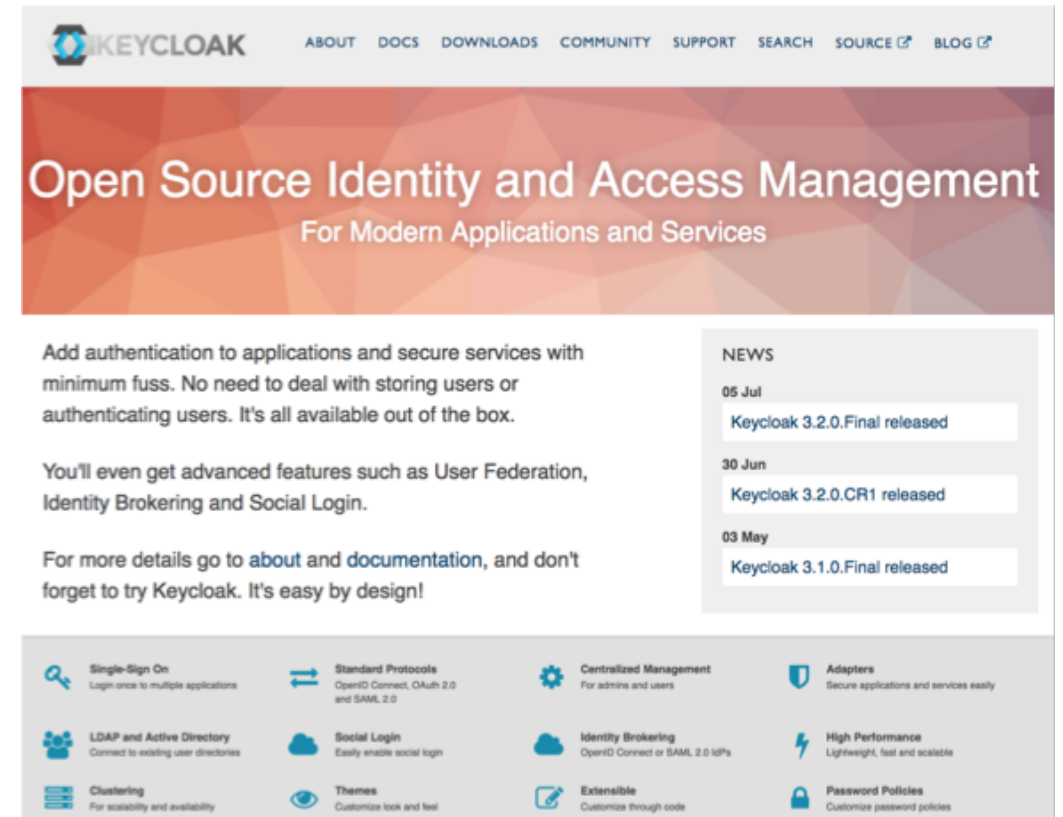
Choose a good elementary block

Offers most of required features

Clean codebase and architecture

Highly extensible and customizable
Modules, extensions, themes, ...

Documented and complete REST API



The screenshot shows the Keycloak website. At the top is a navigation bar with the Keycloak logo and links for ABOUT, DOCS, DOWNLOADS, COMMUNITY, SUPPORT, SEARCH, SOURCE, and BLOG. Below this is a large orange banner with the text "Open Source Identity and Access Management" and "For Modern Applications and Services". The main content area has three paragraphs: "Add authentication to applications and secure services with minimum fuss. No need to deal with storing users or authenticating users. It's all available out of the box.", "You'll even get advanced features such as User Federation, Identity Brokering and Social Login.", and "For more details go to [about](#) and [documentation](#), and don't forget to try Keycloak. It's easy by design!". To the right of the main text is a "NEWS" section with three entries: "05 Jul Keycloak 3.2.0.Final released", "30 Jun Keycloak 3.2.0.CR1 released", and "03 May Keycloak 3.1.0.Final released". At the bottom is a grid of 12 feature icons and descriptions: Single-Sign On, Standard Protocols, Centralized Management, Adapters, LDAP and Active Directory, Social Login, Identity Brokering, High Performance, Clustering, Themes, Extensible, and Password Policies.



Find the gaps

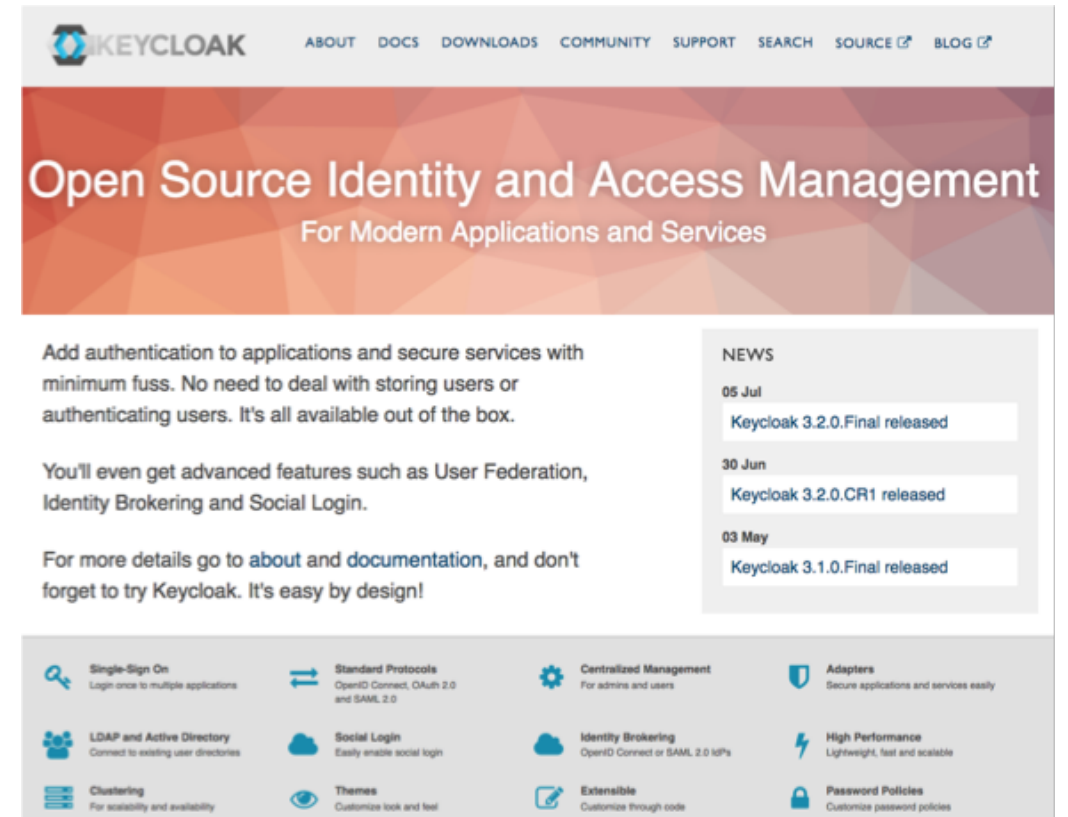
No easy hotplug clustering

No support of Microsoft WSFED

No support for Access control on SAML

No easy way to get Business & Technical audit logs & metrics

API not really adapted to Reactive programming patterns



The screenshot shows the Keycloak website. The header includes the Keycloak logo and navigation links: ABOUT, DOCS, DOWNLOADS, COMMUNITY, SUPPORT, SEARCH, SOURCE, and BLOG. The main banner features the text "Open Source Identity and Access Management" and "For Modern Applications and Services". Below the banner, there is a paragraph about adding authentication with minimum fuss, followed by a paragraph about advanced features like User Federation, Identity Brokering, and Social Login. A "NEWS" section on the right lists three releases: Keycloak 3.2.0.Final released (05 Jul), Keycloak 3.2.0.CR1 released (30 Jun), and Keycloak 3.1.0.Final released (03 May). At the bottom, there is a grid of features including Single-Sign On, Standard Protocols, Centralized Management, Adapters, LDAP and Active Directory, Social Login, Identity Brokering, High Performance, Clustering, Themes, Extensible, and Password Policies.



Fill the holes

High availability, node hotplug and multi DC support

Make Keycloak work with cockroach-db

<https://github.com/cloudtrust/keycloak-cockroach>

Still work in progress (Should be released on June 2018)

Standalone mode with a shared but distributed DB

Require modification of SQL queries and a KeyCloak Data-Store module

Fill the holes

Audits logs and performances monitoring

Create a module to send everything happened within keycloak
<https://github.com/cloudtrust/event-emitter>

Handle retry and buffering messages

Can send event with in JSON or flat buffer

Embed a local snowflake to generate unique and non-colliding event id

Fill the holes

Microsoft WSFED support

Implement WSFED protocol within keycloak

<https://github.com/cloudtrust/keycloak-wsfed>

Working and release planned on April 2018 within a productive environnement

Kerberos SSO with Microsoft Sharepoint

Thousands of users

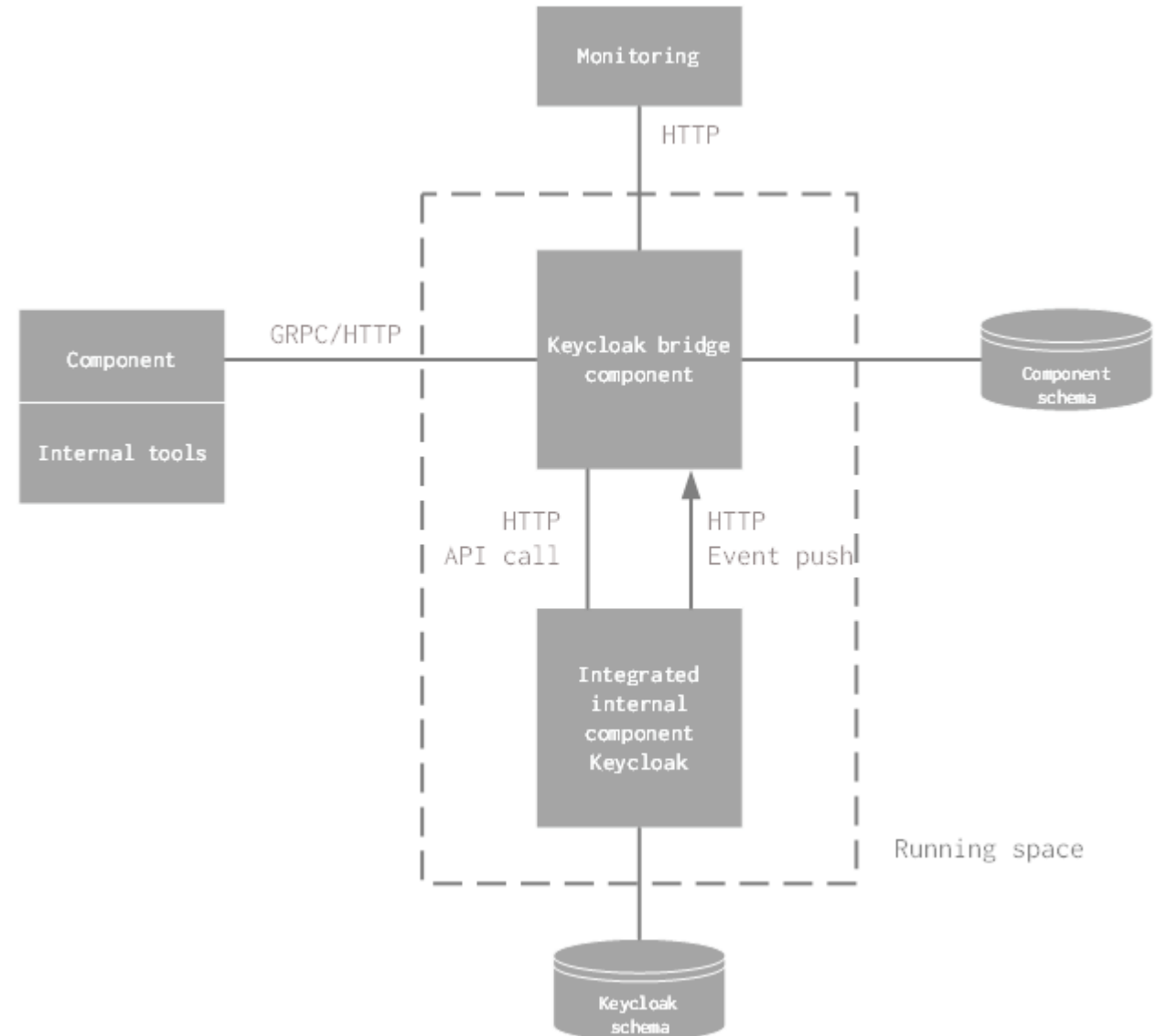
Able to replace Microsoft ADFS

Support Token claim mapping

Still not supporting sso Brokering from WSFED (in the roadmap)

Fill the holes

Reactive programming interface
& others missing features



Fill the holes

Reactive programming interface & others missing features

Create pet services

<https://github.com/cloudtrust/keycloak-bridge>

Work in progress and elementary release planned for April 2018

Normalize and store technical and business event to an Elasticsearch cluster

Normalize and store technical and business metric to an (Influx/DalmatinerDB) Cluster

Fill the holes

Reactive programming interface & others missing features

Implement interfaces for Metrics, Audit tracks & Keycloak API

Implement keycloak's event receiver endpoints

Expose a reactive GRPC interface

Expose a reactive HTTP interface

Expose a business monitoring interface (Create-user, Fake Login, ..)

Expose a technical monitoring interface (DB, Keycloak status, ...)

Fill the holes

Access control on SAML & WSFED

Extend access control available to OICD to SAML and WSFED

Planned to be released on April 2018

Our current runtime setup

First an appliance then, build your own saas

Appliance design

