

SINGLE-SIGN-ON CON G SUITE SULLA CONSOLE DI AMAZON WEB SERVICES.

DevOps

GSuite

Single-Sign-On (SSO)



Simone Merlini | 7 Aprile 2017

Chi, tra gli utilizzatori di della console di AWS, non si è mai scontrato con l'annoso problema di **gestire molteplici utenti su altrettanti account**, dovendo creare diversi IAM user e, per ognuno di essi, password complesse, oltre alla quanto mai fondamentale (ma decisamente scomoda, diciamoci la verità) **two-factor-authentication**?

Proprio riguardo a quest'ultima, dando per scontato che non si voglia utilizzare un token hardware dedicato per ogni singolo IAM user, la scelta ricade quasi obbligatoriamente sul **Google Authenticator**, con codici e QR code che proliferano come funghi e che diventa complicato salvaguardare da eventi nefasti legati allo smartphone (furti, smarrimenti, rotture, backup, cambio device...)

AWS in realtà offrirebbe **un servizio di cross-account access** per la sua management console, che ha però diversi limiti, tra i quali:

- ~~il limite massimo di 5 account AWS gestiti;~~ **[UPDATE: il limite è stato rimosso :)]**
- l'essere basato sui cookie del browser da cui si accede (se cambiamo browser o cancelliamo la cache si azzerà tutto);
- il richiedere almeno uno IAM user "master" da cui partire, che necessita di credenziali di login e TFA dedicate.

La risposta più consona alla necessità di gestire centralmente utenti e accessi, per AWS così come per la stragrande maggioranza delle applicazioni che necessitano autenticazione multi-utente, si chiama **Single-sign-on** (SSO).

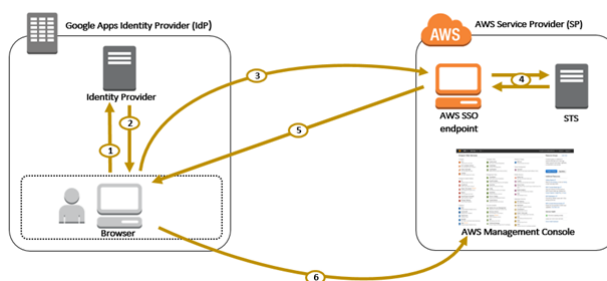
Tipicamente, il meccanismo del SSO si basa su un **Identity Provider** (un repository centralizzato di tutte le identità aziendali con relativi attributi - username, password, gruppi, ruoli ecc...) e una serie di **Service Provider** (le applicazioni dove gli utenti si possono loggare con le loro identità aziendali) che vengono federati all'Identity Provider con **relazioni di trust forte**, basate tipicamente sulla

condivisione di chiavi, certificati o token. In questo modo gli utenti possono utilizzare un'unica utenza (e quindi una sola password e una singola TFA), gestita in maniera centralizzata, per loggarsi in tutte le applicazioni per le quali sono stati abilitati.

Se i Service Provider possono essere le applicazioni più disparate (Web, desktop, mobile, accesso remoto, CLI, API ecc....), gli Identity Provider sono quasi sempre dei **server LDAP** o **Microsoft Active Directory**. In particolare MS AD è lo standard de facto nella maggior parte delle aziende più strutturate per la gestione delle identità aziendali, ed è infatti supportato di default da tutte le applicazioni che prevedano la possibilità di fare SSO.

Tuttavia **non è così comune trovare implementata un'infrastruttura MS AD** (ma il discorso vale in parte anche per LDAP), in particolar modo nelle realtà più piccole o giovani e agili, per ragioni che spaziano **dai costi alla complessità di gestione** (specie se si necessita del servizio di AD erogato in alta affidabilità), senza trascurare il fatto che MS AD è tipico di realtà Microsoft-centriche (quasi tutte le grandi aziende legacy) ed è quindi meno diffuso dove il parco client è più eterogeneo (Windows+Mac+Linux...).

Una tendenza molto diffusa in realtà è quella di utilizzare come Identity Provider l'account Google Apps (da poco rinominato in **G Suite**) aziendale, servizio ampiamente diffuso e largamente utilizzato per le funzionalità di posta elettronica e collaboration. In questo modo si può fare SSO sulle numerosissime **applicazioni che già nativamente supportano la funzione "login with Google"**, ma anche su quelle (come è il caso della console di AWS) che supportano lo **standard SAML**, standard per il quale da circa un anno G Suite eroga il servizio di Identity Provider.



Schema di funzionamento dell'autenticazione tramite SAML tra G Suite e la console di AWS

Vediamo quindi come **configurare i nostri account AWS e G Suite per far funzionare il Single-Sign-On con SAML**.

Innanzitutto, nella pagina di amministrazione di G Suite, dobbiamo aggiungere degli attributi custom ai nostri utenti, tramite i quali il nostro Identity Provider (Google) comunicherà al Service Provider (AWS) oltre all'identità dell'utente loggato, informazioni aggiuntive che spiegheremo in seguito.

Manage user attributes		
Last signed in	Email usage	
Feb 16	0 GB	
Feb 16	0 GB	
5:37 PM PDT	0 GB	

Creiamo una categoria di attributi custom e chiamiamola “AWS SAML” e creiamo gli attributi “IAM Role” e “SessionDuration”. E’ importante che entrambi gli attributi siano privati (ovvero non visualizzabili da tutti gli utenti della vostra organizzazione) e che l’attributo “IAM Role” supporti valori multipli.

User attributes

View and manage user attributes. Find out [how to create attributes](#).

Default categories

- Basic information 3 Attributes
- Contact information 3 Attributes
- Employee details 6 Attributes

Custom categories

- AWS SAML 2 Attributes

ADD CUSTOM CATEGORY DONE

AWS SAML

Attribute name	Type	Multiple values	Private
IAM_role	Text	Yes	Yes
SessionDuration	Integer	No	Yes
Enter an attribute name	Text	No	Yes

DELETE CATEGORY BACK SAVE

Fatto questo andiamo nella sezione “Apps” e aggiungiamo una nuova applicazione SAML, partendo dal template pre-configurato per AWS.

Google Admin Search for users, groups, and settings (e.g. migrate email)

Apps

APPS SETTINGS Marketplace settings

9

G Suite

Gmail, Calendar, Drive & more

These services are governed by your G Suite agreement.

51

Additional Google services

Blogging, photos, video, social tools and more

These services are not governed by your G Suite agreement and other third-party terms. [Learn more](#)

0

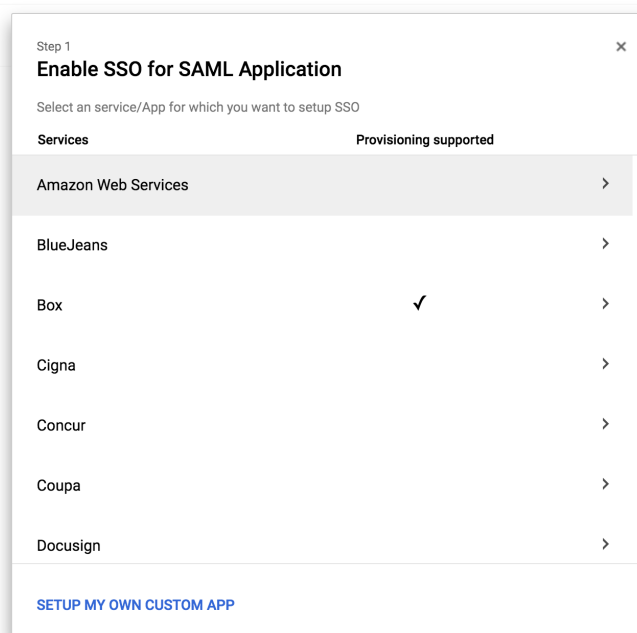
Marketplace apps

Microsoft, Salesforce, SAP and more

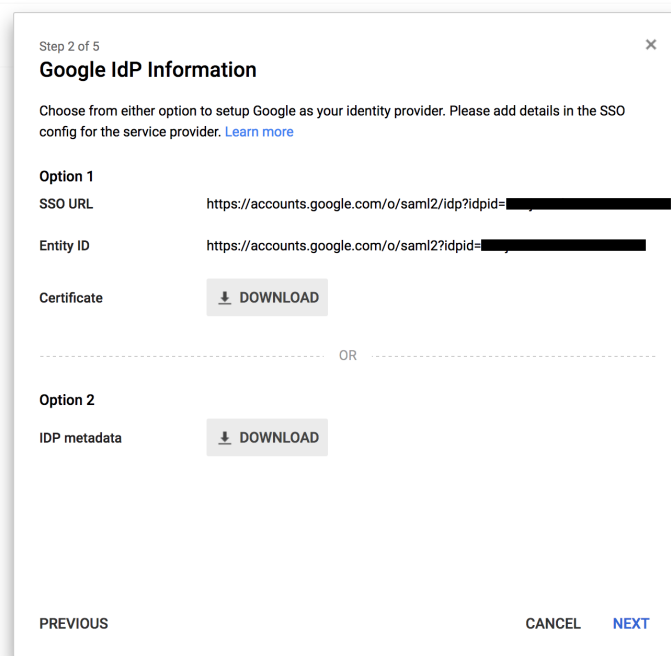
0

SAML apps

Manage SSO and user provisioning



Scarichiamo (opzione 2) gli IDP Metadata (di fatto un file .xml che contiene alcuni parametri di configurazione e il certificato X509 su cui si basa la relazione di *trust* tra IdP e SP) e teniamoli da parte per un passaggio successivo. **ATTENZIONE!!! Il contenuto di questo file non deve essere diffuso per nessun motivo; sulla sua segretezza si basa la sicurezza di tutta la soluzione!**



Proseguiamo nella configurazione mappando l'entità SAML "Name ID" su "Primary Email" (ovvero l'utente verrà presentato alla console AWS con l'indirizzo mail come identificativo univoco).

Step 4 of 5

Service Provider Details

Please provide service provider details to configure SSO for Amazon Web Services. The ACS url and Entity ID are mandatory. [Learn more](#)

ACS URL *	https://signin.aws.amazon.com/saml	
Entity ID *	https://signin.aws.amazon.com/saml	
Start URL		
Signed Response	☐	
Name ID	Basic Information	Primary Email
Name ID Format	EMAIL	

PREVIOUS CANCEL NEXT

Nel passaggio successivo dobbiamo configurare tre mapping aggiuntivi (attenzione che qui la UI di G Suite non è molto chiara, in quanto non si leggono bene gli URL nella colonna di sinistra):

- L'attributo <https://aws.amazon.com/SAML/Attributes/RoleSessionName> va mappato nuovamente su "Primary Email"
- L'attributo <https://aws.amazon.com/SAML/Attributes/Role> va mappato sull'attributo custom "IAM role" che abbiamo creato prima. In questo modo stiamo dicendo ad AWS quali ruoli l'utente è autorizzato ad assumere e su quali account.
- Va aggiunto l'attributo <https://aws.amazon.com/SAML/Attributes/SessionDuration> che va mappato sull'attributo custom "SessionDuration" che abbiamo creato prima. Qui stiamo dicendo ad AWS quanto deve durare la sessione di un determinato utente prima di essere sloggato in automatico.

Step 5 of 5

Attribute Mapping

Provide mappings between service provider attributes to available user profile fields.

https://aws.amazon.com/SAML/Attributes/RoleSessionName	Basic Information	Primary Email
https://aws.amazon.com/SAML/Attributes/Role	AWS SAML	IAM_role
https://aws.amazon.com/SAML/Attributes/SessionDuration	AWS SAML	SessionDuration

ADD NEW MAPPING

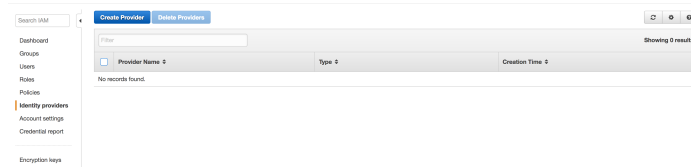
PREVIOUS CANCEL FINISH

È molto utile poter personalizzare questo parametro perché di default la sessione dura 1 ora, e, per esperienza di chi lavora parecchio sulla console AWS, è un valore molto basso, che comporta

numerosi e scomodi logout forzati durante l'operatività quotidiana. **ATTENZIONE!!! Questo “trick” è un'esclusiva di beSharp; non è documentato sulle guide ufficiali ne' di Google ne' di AWS! (nda).**

A questo punto la configurazione di G Suite è terminata e passiamo alla configurazione di AWS.

Andiamo nella sezione **IAM** -> Identity Providers e creiamone uno nuovo, di tipo SAML, che chiamiamo “GoogleApps”; in questo punto dovremo caricare gli IdP metadata che abbiamo scaricato in precedenza (una volta caricato il file in questo punto, suggerisco di cancellarlo dal proprio computer)



Configure Provider

Choose a provider type.

Provider Type*

Choose a provider type

SAML

OpenID Connect

Configure Provider

Choose a provider type.

Provider Type*

SAML

Provider Name*

GoogleApps

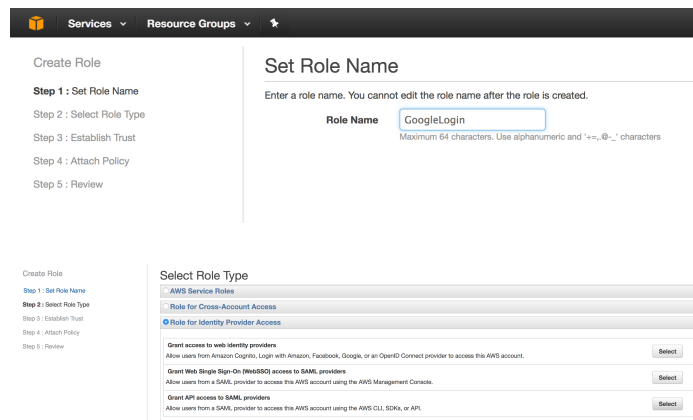
Maximum 128 characters. Use alphanumeric and '._-' characters.

Metadata Document*

C:\fakepath\GoogleIDPMetadata:

Choose File

Poi creiamo un nuovo IAM role che chiamiamo “GoogleLogin” e come tipo di ruolo selezioniamo “Identity Provider Access” -> “WebSSO” e lo associamo all'Identity Provider “GoogleApps” che abbiamo appena creato.



Create Role

Step 1 : Set Role Name

Step 2 : Select Role Type

Step 3 : Establish Trust

Step 4 : Attach Policy

Step 5 : Review

Select the SAML provider to trust. Federated users from the selected provider can access resources from your AWS account using the AWS Management Console. For WebSSO, the audience attribute (SAML:aud) is automatically set to https://signin.aws.amazon.com/saml.

SAML provider	GoogleApps
Attribute	SAML:aud
Value	https://signin.aws.amazon.com/saml
Add Conditions (optional)	

Nei passi successivi del wizard associamo una policy allo IAM role per dare i permessi (nel nostro esempio abbiamo dato i permessi di admin - **DON'T TRY THIS AT HOME!!!** 😊) e la configurazione lato AWS è terminata.

Create Role

Step 1 : Set Role Name

Step 2 : Select Role Type

Step 3 : Establish Trust

Step 4 : Attach Policy

Step 5 : Review

Verify Role Trust

You can customize the role's trust relationship by editing the following policy document. [Learn more about role trust policies.](#)

Policy Document

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": "sts:AssumeRoleWithSAML",
7       "Principal": {
8         "Federated": "arn:aws:iam::5[redacted]:saml-provider/GoogleApps"
9       },
10      "Condition": {
11        "StringEquals": {
12          "SAML:aud": "https://signin.aws.amazon.com/saml"
13        }
14      }
15    }
16  ]
17 }
```

Create Role

Step 1 : Set Role Name

Step 2 : Select Role Type

Step 3 : Establish Trust

Step 4 : Attach Policy

Step 5 : Review

Attach Policy

Select one or more policies to attach. Each role can have up to 10 policies attached.

Policy Name	Attached Entities	Creation Time	Edited Time
☒ AdministratorAccess	2	2015-02-06 19:39 UTC+0200	2015-02-06 19:39 UTC+0200
☐ AmazonEC2FullAccess	1	2015-02-06 19:40 UTC+0200	2015-02-06 19:40 UTC+0200
☐ AmazonEC2InstanceMonitoringRole	1	2015-11-11 20:58 UTC+0200	2015-11-11 20:58 UTC+0200
☐ AmazonAPIGatewayAdministrator	0	2015-07-09 19:34 UTC+0200	2015-07-09 19:34 UTC+0200
☐ AmazonAPIGatewayFullAccess	0	2015-07-09 19:36 UTC+0200	2015-07-09 19:36 UTC+0200
☐ AmazonAPIGatewayPushToCloudWatch...	0	2015-11-12 00:41 UTC+0200	2015-11-12 00:41 UTC+0200
☐ AmazonKinesisFullAccess	0	2015-02-06 19:40 UTC+0200	2015-02-06 19:40 UTC+0200
☐ AmazonKinesisReadOnlyAccess	0	2015-02-06 19:40 UTC+0200	2015-12-07 22:00 UTC+0200
☐ AmazonKinesisServiceAccess	0	2015-11-19 06:17 UTC+0200	2015-11-19 06:17 UTC+0200
☐ AmazonKinesisFullAccess	0	2015-11-30 13:48 UTC+0200	2015-11-30 13:48 UTC+0200
☐ AmazonCloudDirectoryFullAccess	0	2017-03-25 01:41 UTC+0200	2017-03-25 01:41 UTC+0200
☐ AmazonCloudDirectoryReadOnlyAccess	0	2017-03-25 01:40 UTC+0200	2017-03-25 01:40 UTC+0200
☐ AmazonCognitoDeveloperAuthenticated...	0	2015-03-24 18:22 UTC+0200	2015-03-24 18:22 UTC+0200
☐ AmazonCognitoPowerUser	0	2015-03-24 18:14 UTC+0200	2016-06-02 18:57 UTC+0200
☐ AmazonCognitoReadOnly	0	2015-03-24 18:06 UTC+0200	2016-06-02 19:30 UTC+0200

[Cancel](#)
[Previous](#)
[Next Step](#)

Create Role

Step 1 : Set Role Name

Step 2 : Select Role Type

Step 3 : Establish Trust

Step 4 : Attach Policy

Step 5 : Review

Review

Review the following role information. To edit the role, click an edit link, or click **Create Role** to finish.

Role Name	GoogleLogin	Edit Role Name
Role ARN	arn:aws:iam::5[redacted]:role/GoogleLogin	
Trusted Entities	The identity provider(s) arn:aws:iam::5[redacted]:saml-provider/GoogleApps	
Policies	arn:aws:iam::aws:policy/AdministratorAccess	Change Policies

Adesso torniamo nell'amministrazione di G Suite, per assegnare ai singoli utenti i permessi relativi a quali ruoli possono assumere e su quali account AWS.

Google Admin

Search for users, groups, and settings (e.g., create group)

Users > Mario Rossi

M

Mario Rossi

m.rossi@bluewin.ch

User - Active

Last login: 5:57 PM PDT

0 GB

Mail storage used

0

Quota used

Account

Basic information

Name

Mario Rossi

Email

m.rossi@bluewin.ch

Manage user attributes

2 custom user attributes in 1 category

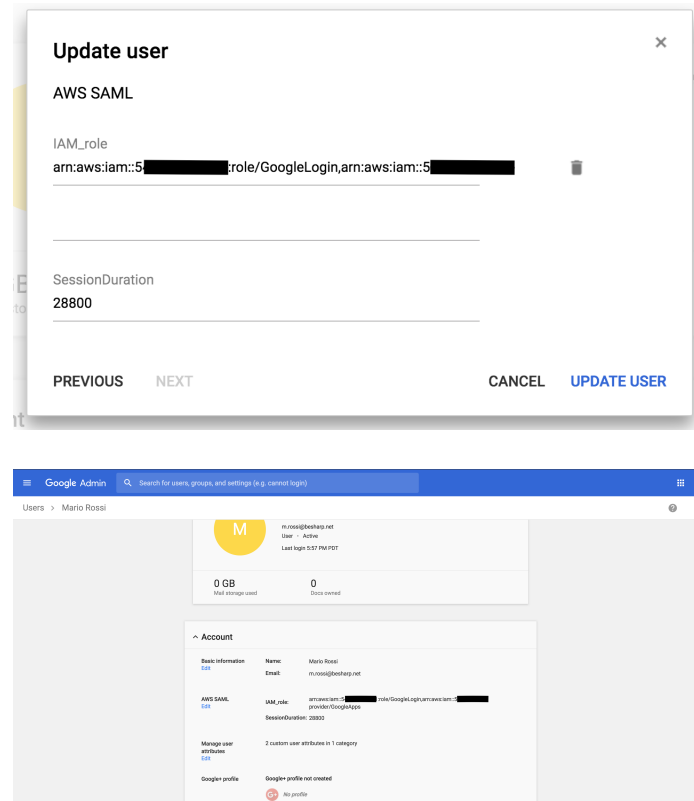
Google profile

Google profile not created

Questa è la parte meno intuitiva della configurazione: per quanto riguarda i ruoli e gli account accessibili, AWS si aspetta da Google dei valori di questo tipo:

```
arn:aws:iam::1234567891012:role/GoogleLogin,arn:aws:iam::1234567891012:saml-provider/GoogleApps
```

Come vedete si tratta di due **ARN** separati da una virgola. Il primo è l'ARN del ruolo che quell'utente può assumere, il secondo è l'ARN dell'identity provider che abbiamo creato all'interno dell'account AWS. (il numero 1234567891012 è un segnaposto che va sostituito con il vero account number del vostro account AWS). Questo valore va inserito nel campo custom "IAM role" che abbiamo creato in precedenza. In questo modo possiamo specificare per ogni utente quale ruolo può assumere e su quale account AWS.



Se vi ricordate bene abbiamo fatto in modo che l'attributo "IAM role" supportasse valori multipli; infatti è possibile, per lo stesso utente, specificare più ruoli all'interno dello stesso account, o anche su account differenti. Basterà aggiungere altre tuple del tipo:

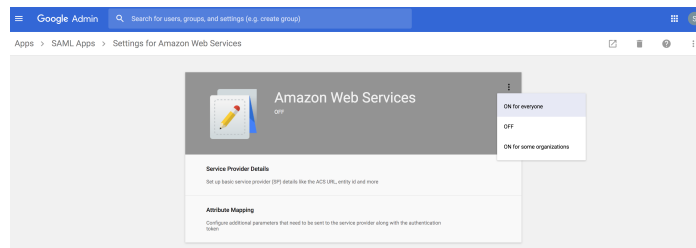
```
arn:aws:iam::112233445566:role/Ruolo1,arn:aws:iam::112233445566:saml-provider/GoogleA  
pps  
arn:aws:iam::112233445566:role/Ruolo2,arn:aws:iam::112233445566:saml-provider/GoogleA  
pps
```

e subito dopo il login ci verrà chiesto quale ruolo vogliamo assumere su quale account.

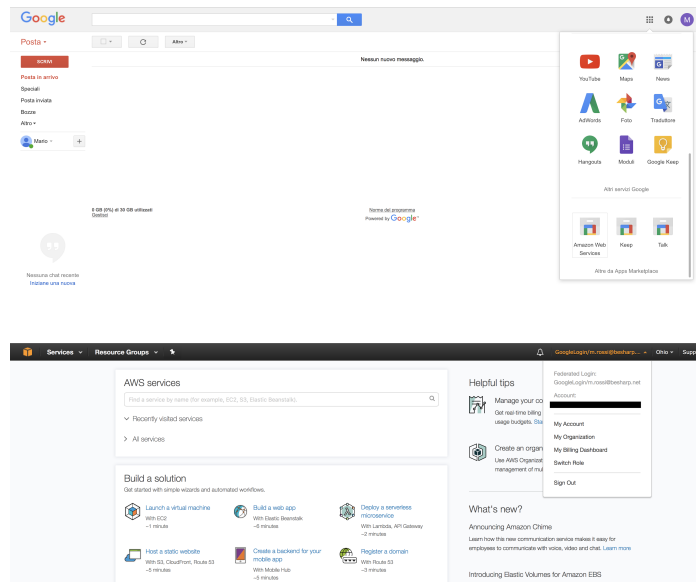
Ovviamente, perchè tutto funzioni, nel nostro esempio dovremmo ripetere la procedura di federazione SAML (con lo scambio degli IdP metadata) anche per l'account 112233445566

Nel campo custom "SessionDuration" potete specificare, per ogni utente, la durata in secondi della sessione di login. Suggerisco il valore 28800, che corrisponde a 8 ore, ovvero circa a una tipica giornata lavorativa.

A questo punto non ci resta che abilitare la SAML application che abbiamo creato, e tutti gli utenti si troveranno una nuova icona nel loro menu di accesso rapido alle Google Apps.



Facciamo login quindi con il nostro account di prova “Mario Rossi” e, cliccando sull'icona corrispondente, verremo magicamente indirizzati verso la console AWS dove, come potete vedere, risultiamo loggati con il nostro account federato, m.rossi@besharp.net.



Soddisfatti? 😊

Nel prossimo articolo vedremo come abbiamo utilizzato, in maniera decisamente creativa, lo stesso approccio basato su G Suite e SSO per utilizzare ai servizi AWS che prevedono l'autenticazione tramite la coppia key/secret, come la [AWS CLI](#), [CodeCommit](#) e l'accesso alle API da client extra VPC.

Negli articoli successivi, invece, approfondiremo l'utilizzo di G Suite come Identity Provider per tutti gli altri servizi che normalmente richiederebbero una federazione con Active Directory o LDAP.

Avete dubbi o domande? Commentate l'articolo, vi risponderemo ASAP! E se volete implementare una soluzione come questa ma vi manca il tempo o la voglia... [chiamateci!](#)



Simone Merlini

CEO e co-fondatore di beSharp, Cloud Ninja ed early adopter di qualsiasi tipo di soluzione *aaS. Mi divido tra la tastiera del PC e quella a tasti bianchi e neri; sono specializzato nel deploy di cene pantagrueliche e nel test di bottiglie d'annata.

Get in touch

beSharp.it
proud2becloud@besharp.it

Copyright © 2011-2021 by beSharp srl - P.IVA IT02415160189