# Enhanced Email System

Igor Zboran
izboran@gmail.com
September 11, 2020

## Abstract

Electronic mail (email) is the most pervasive form of business information exchange. Email is often used not only as an interpersonal communication tool, but also as the default choice to send files. In this paper the User-Managed Access (UMA) authorization framework is proposed to address data storage, access control and data transfer limitations of current email systems. Outgoing mail is typically transferred from the source system to the destination system as a single text-encoded file using Simple Mail Transfer Protocol (SMTP). SMTP is a push protocol only. The UMA framework introduces a resource server and an authorization server into the email system. The resource server is accessed generally by HTTP protocol that was designed as a pull protocol. The two-way push-pull data transfer in combination with a data storage system controlled by the standardized authorization framework significantly leverages email security, enhances email system utilization and elevates the email ecosystem to the ubiquitous Content Services platform.

## Introduction

The main components of the email system have been designed between 1971 and 1992 by many inventors. In the course of time, email has become the most commonly used application of the Internet. Nowadays the email is the only truly decentralized communication system of the Internet and the email infrastructure forms the backbone of the worldwide digital identity.

## Problem

Despite the importance of email infrastructure, the whole ecosystem still relies on over 40 year-old architecture and protocol design. There are spam and attachment issues from the very beginning. The email system, while conceptually sound as a communication means, is structurally obsolete and functionally deficient.

### Current Situation

With the rising popularity of free email providers, such as Gmail or Outlook.com, web-browsers are increasingly being used to access email server. From a user standpoint, it is easy to read and send email via web-browser on any device, from anywhere in the world. Centralized access to the mailboxes, increases the security of web-based email systems.

### Current Flaws

Even though the main email service providers claim email accounts to be safe, the fact remains that major security and functional flaws are not fixed. There is still an attachments delivery dichotomy; the bulky files are not transferred as an attachment but are shared via links. An “attachment sharing” is not natural for current email systems where each message with attachments is expected to be consistent. Shared links pose a consent phishing attack threat where attacker tricks users into granting a malicious application access to sensitive resources. This is known as an OAuth 2.0 authorization exploit. The Enhanced Email System is resistant to this security exploit as there are no direct user involvement in access granting.

# Proposed Solution

Given that email system is lagging behind modern communication and collaboration tools, we propose an OAuth-based access control management and consequently a new data exchange channel for the email ecosystem.

## Motivation

Email still the most popular communication tool is lacking an important part of today's modern systems – an authorization framework. Understanding this lead us to implement the UMA authorization framework into the email ecosystem.

## Main Concept

The Enhanced Email System is designed to follow the Identity and Access Management (IAM) best practices while keeping compatibility with current email systems. We propose to incorporate the UMA framework between the email system with standardized SMTP/POP3/IMAP interface and the proprietary RESTful web-based email (Webmail) application as it is illustrated in Figure 1.

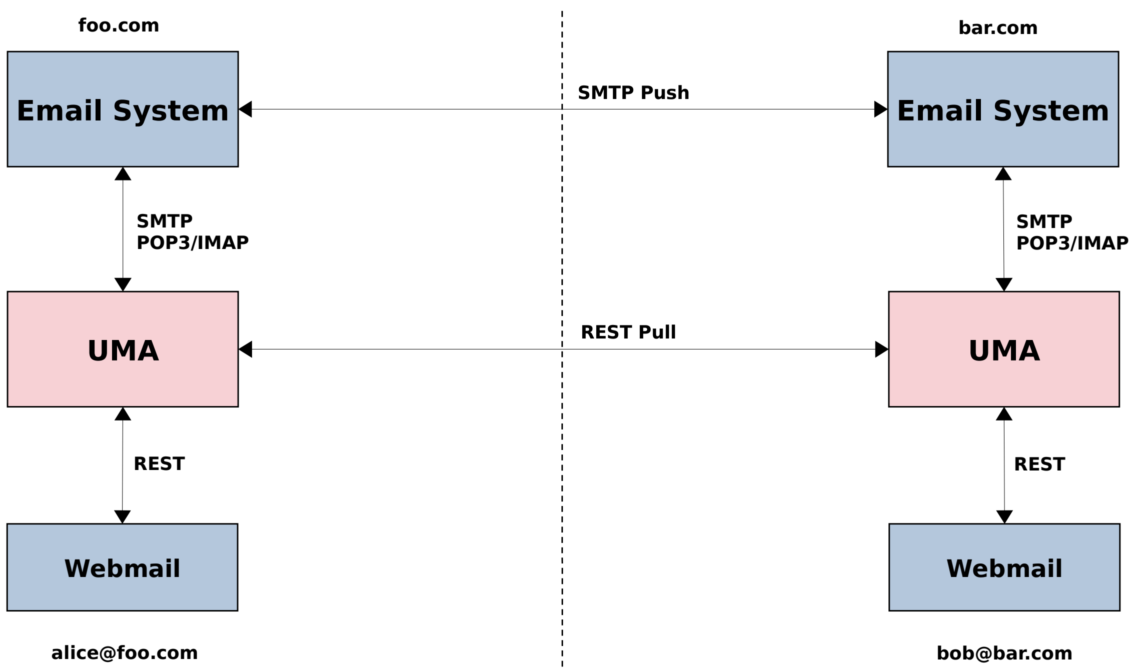


Figure 1. Main concept

Key points:

1. The actual contents of email messages and attachments are stored separately in the email repository. Data at rest are not typically protected by the UMA framework.
2. The contents of the email in the email repository are temporarily shared with the recipient. Following a successful sharing process, links to content are sent to the recipient via the email infrastructure. Data in the transfer process falls under UMA framework protection.
3. The recipient’s mail agent receives the email with the links to content, authenticates against the sender’s UMA framework, gets authorized access and downloads the contents of the email from the sender’s repository. The agent then creates copies of the downloaded data and stores them in the recipient's repository.

UMA uses a special jargon. For the sake of brevity of this proposal, the following list of acronyms will be used:

- AS Authorization Server
- RS Resource Server
- RO Resource Owner
- RqP Requesting Party
- PAT Protection API (access) Token
- RPT Requesting Party Token
- PCT Persisted Claims Token

We introduce some new acronyms:

- EES (pronounced "ease") Enhanced Email System
- MFA Mail Fetch Agent
- OAT OAuth (access) Token

The UMA framework plays its role during the data exchange process between mailboxes. The Webmail application gives the sender a user-centric approach to manage and protect his/her ready-to-send resources while the MFA that acts on behalf of the recipient is used to access and download sender's resources as it is illustrated in Figure 2.

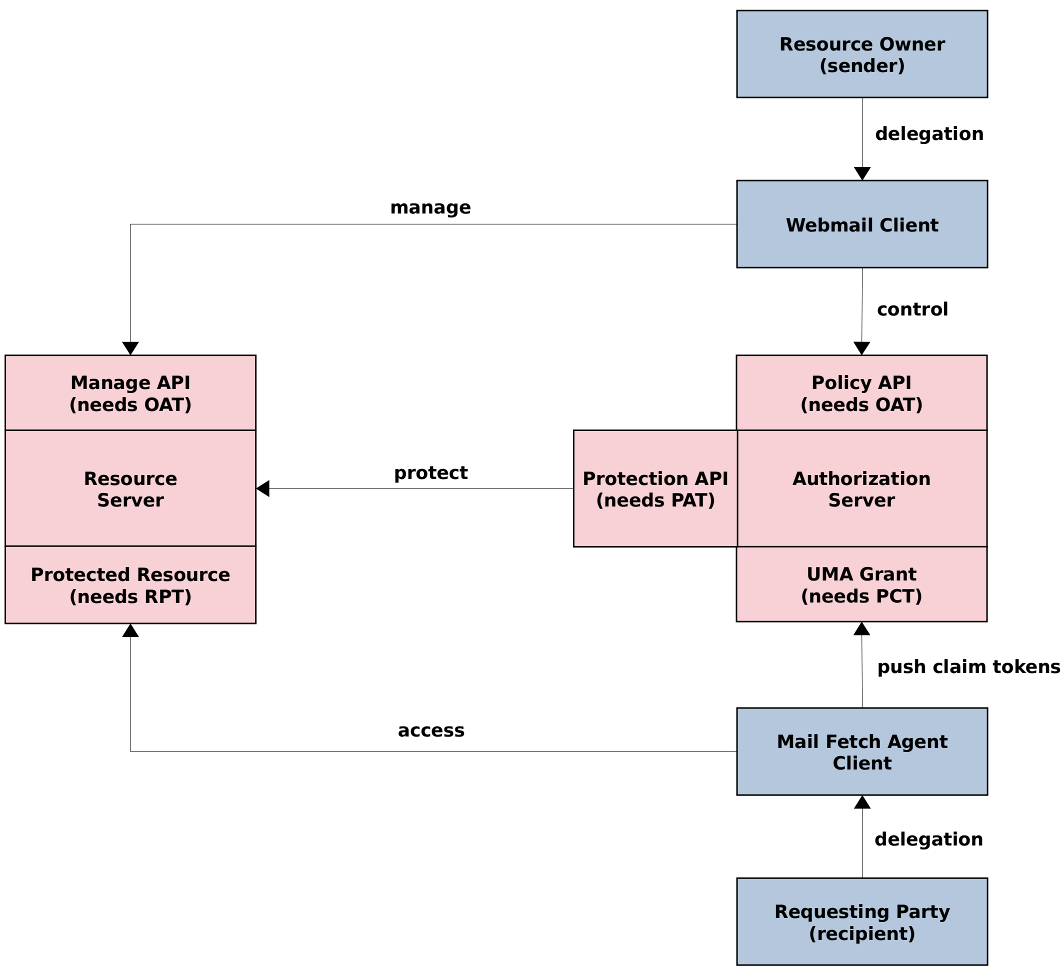


Figure 2. EES/UMA abstract flow

Figure 3 provides the sequence diagram for the EES/UMA Grant when the MFA client pushes a recipient's email address claim.

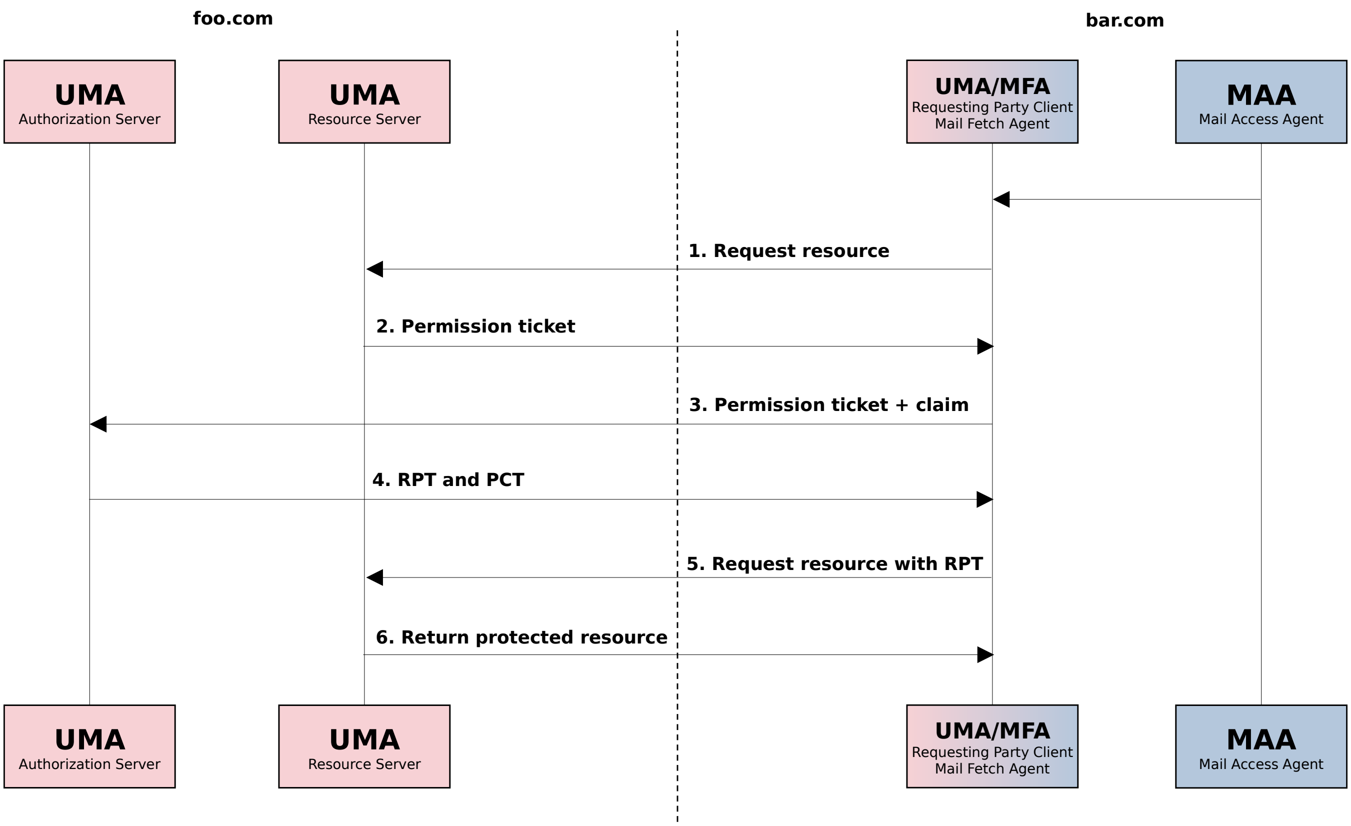


Figure 3. EES/UMA Grant sequence diagram (with pushed claim)

The following is a description of steps to get access to a protected resource:

1. The RqP client (aka MFA) is trying to access an UMA protected resource at the RS.
2. Without an access token, the RS returns a permission ticket.
3. The client presents the permission ticket and claim along with client credentials at the AS token endpoint (the claim value contains recipient's email address).
4. The AS returns the access token (RPT) and PCT on successful evaluation of the policies.
5. The client again tries to obtain the resource.
6. The RS returns the protected resource after validating the access token.

## Trust Model

The UMA authorization framework was originally designed for Business-to-Consumer (B2C) scenarios. In UMA the roles of the AS, RS, RO and RqP client are co-located, they are all under the realm of a single trust domain. Fortunately during the development of UMA 2.0, the working group also considered a wide ecosystem where you can access a previously unknown UMA-protected RS. EES combines a decentralized email ecosystem with the UMA wide ecosystem to satisfy both the B2C and Business-to-Business (B2B) scenarios.

Figure 4 illustrates the decentralized three-way trust relationship model:

- Mail Trust – SMTP to SMTP trust (the most vulnerable).
- Mail to UMA Trust – a trust delegation from the email system to the UMA framework.
- UMA Trust – a trust between UMA components.

There is no contract between authorization servers and UMA roles remains co-located.

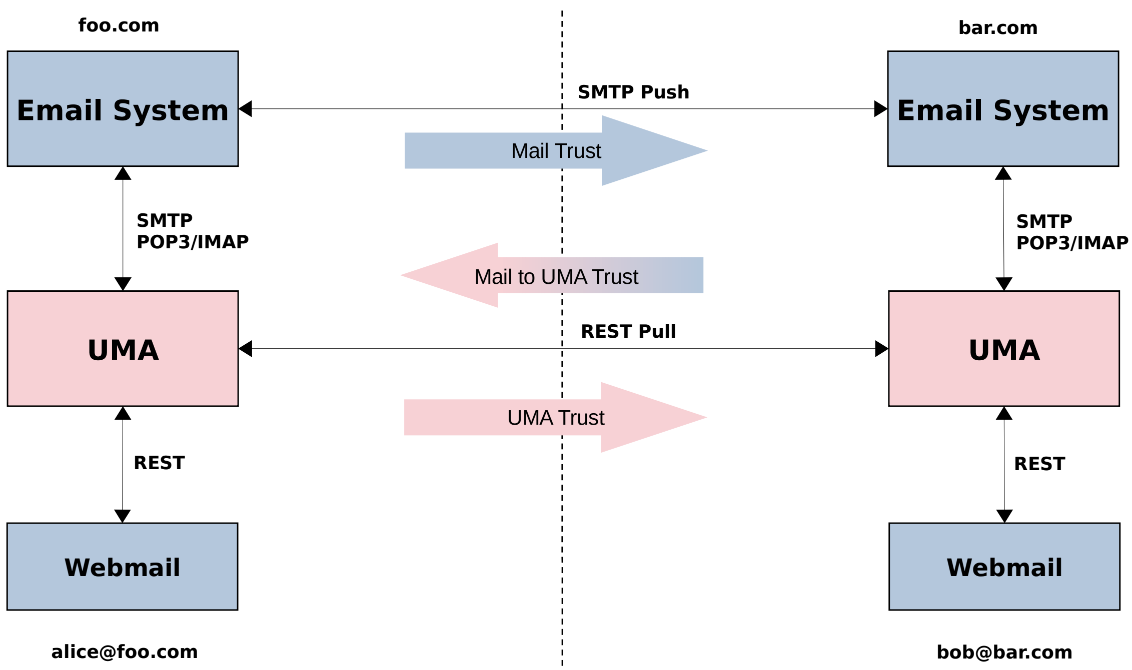


Figure 4. alice@foo.com to bob@bar.com trust model

## Scenarios and Flows

The following scenario represents an email sent from alice.foo.com address to bob.bar.com address assuming that this is the first ever communication between the foo.com and bar.com security domains and no previous trusted relationships was established between them. Trusted communication between users in the two security domains can be divided into an one-time initial registration action and three main phases.

**Registration action - get authorization, register, set up a relationship**

Before the communication itself, a trust relationship governed by a contract must be established between the bar.com RqP client (aka MFA) and the foo.com AS. To set up a relationship a registration message in an email must be sent from foo.com domain to the bar.com domain. The sending of the registration message must be authorized by the foo.com AS. To make this process streamlined OAuth 2.0 Dynamic Client Registration Management Protocol (RFC 7592) is used to avoid manual registration workflow as it is illustrated in Figure 5. The Initial Access Token and the Software Statement are sent in the registration message. This registration message is generated by the foo.com RS and is typically bundled with the main email message sent by the user.

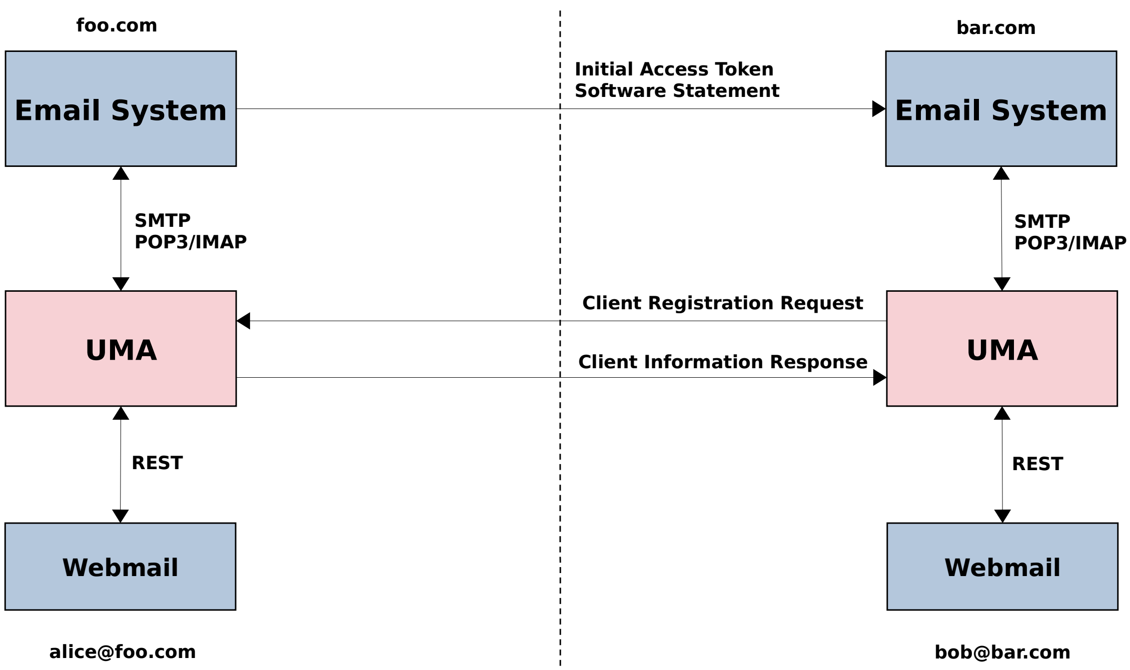


Figure 5. alice@foo.com to bob@bar.com dynamic client registration

After registration a trust relationship is set up between bar.com RqP client and the foo.com AS.

**Phase I - put resources under protection, create policy, send resources links to the recipient(s)**

Let us assume the sender Alice – the resource owner (RO) – has prepared a draft email with an attachment. The draft message and the attachment are stored separately in the email repository – the UMA protected RS. The draft resources are not protected by UMA framework at this stage; data are considered inactive “data at rest”.

Before Alice presses the "Send" button, she fills in the "To" field with the recipient’s email address – bob@bar.com. This value will be used to set up an access to the email resources.

After Alice presses the "Send" button mutable draft resources become immutable email resources protected by UMA framework. Resources are registered at the AS resource registration endpoint. Next, a policy that gives access to email resources to Bob is created. Finally an email with links to email resources – in the ESS email format – is composed and sent to the recipient Bob. The data flow is illustrated in Figure 6.

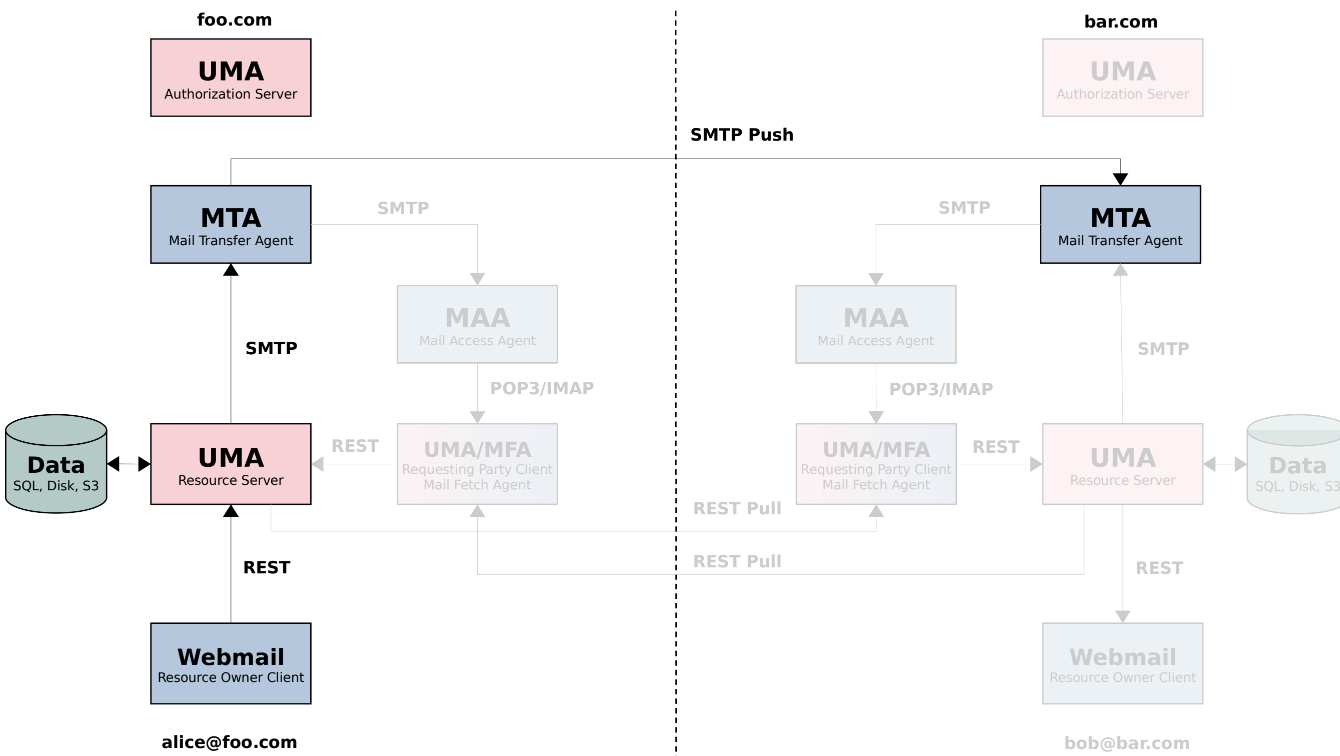


Figure 6. alice@foo.com to bob@bar.com – send resources links

Note: The specification of the ESS email format is outside the scope of this document.

**Phase II – get email with resources links, push claims, get authorization**

An email from Alice with links to her repository has just arrived at Bob's email provider. The mail access agent (MAA) notifies RqP client (aka MFA) – authenticated via IMAP/OAuth2 – of incoming email. The RqP client loads the email and checks its format. The data flow is illustrated in Figure 7. If it is the EES format, links to the Alice's repository are extracted and authorization process – using the EES/UMA Grant protocol – will proceed to get access to her resources. The Bob's email address is used in the pushed claim value. The sequence diagram is illustrated in Figure 3.

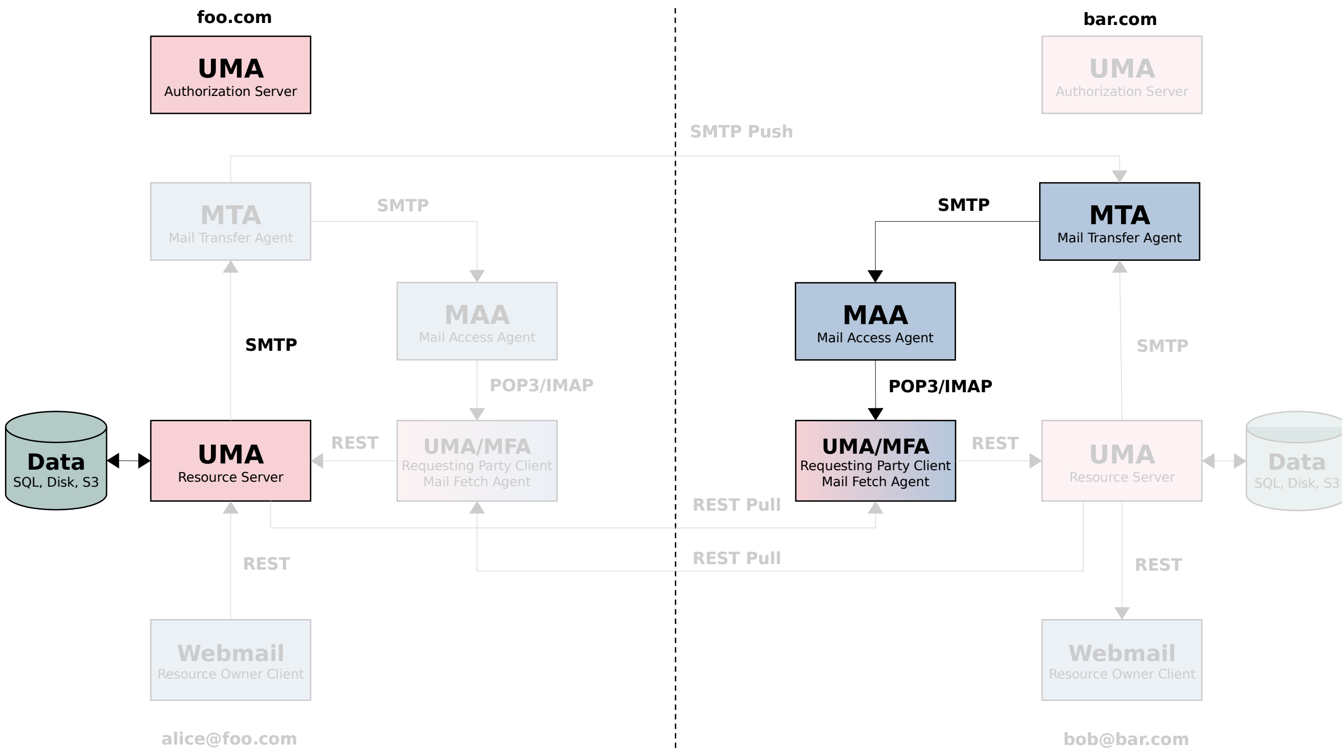


Figure 7. alice@foo.com to bob@bar.com – get resources links

**Phase III - get data, notify the Webmail client**

After authorization using the EES/UMA Grant protocol flow, Alice's email resources are downloaded as it is illustrated in the last section of sequence diagram in Figure 3. The RqP client (aka MFA) should store the downloaded data in the Bob's email repository as it is illustrated in Figure 8. Bob's Webmail application should be notified of Alice's incoming email.

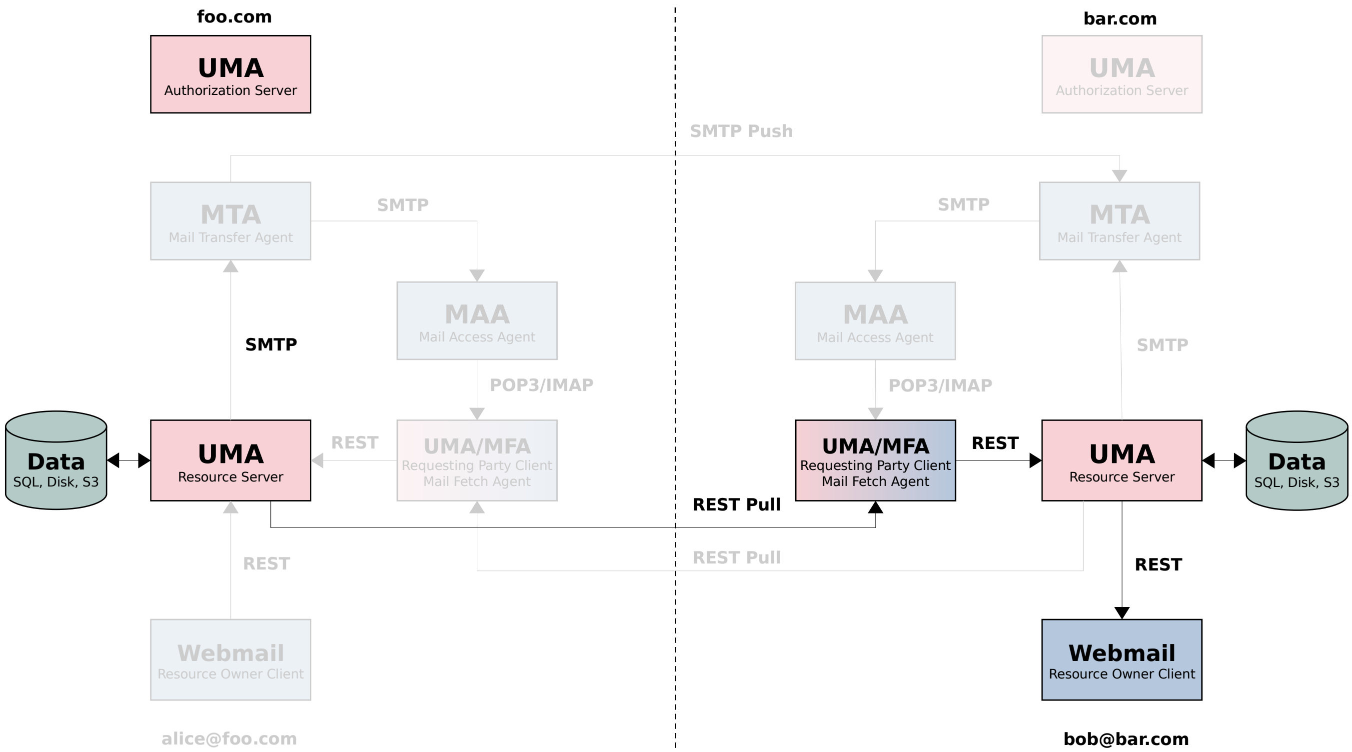


Figure 8. alice@foo.com to bob@bar.com – get data

# Features and Comparison with Current Email System

The novelty of the proposed solution approach can be assessed by comparison with the current email system.

## New Features

The proposed EES solution provides several new features that are lacking in current email systems:

- Intrinsic privacy-preserving properties. Each user can have their own separate RS as an email repository. The user can run its own RS, even its own AS.
- Built-in cross-domain autonomic (without conscious user intervention) access control using the standardized UMA framework.
- Autonomous (without interfering with the email system) data exchange channel.
- No attachments size limit. Attachments are transferred as separate files without size limit.
- Linked content using a clickable hyperlinks.
- Instant messages. Messages and attachments are transferred separately, there is no need to wait for incoming bulky message-with-attachments file. Attachments-stripped bare messages are transferred with a higher priority.

## Comparison with Current Email System

From the users point of view, the use of EES has many advantages over the standard email system. In the following we highlight the advantages of the proposed solution compared to the current email system.

1. Security and Privacy:

The architecture of EES guaranties more control over potential security and privacy issues such as leakage of intellectual property or loss of confidential content and makes this system compatible with enterprise security policies.

2. Usability:

At the core of proposed solution is an attempt to improve the usability of email – not only as an interpersonal communication tool, but also as the default choice to send and store files. With the ability to store, locate, send and receive any content including documents, images, audios and videos the proposed solution can be considered a promising platform for Content Services.

3. Integrations:

EES provides a standardized Restful API interface to ease the integrations with external marketing, sales, Enterprise Content Management (ECM) or Customer Relationship Management (CRM) systems.

# Conclusion

EES can play an important role in communication across various industries in the public and private sectors. The combination of repository, communication and identity represents a central point for information exchange and collaboration throughout any organization.

## Overall Summary

The email system technology in combination with the UMA framework creates a composite architecture that meets the needs of the modern communication tool. This architecture increases robustness and performance of existing email ecosystem. The proposed solution can be used as a Content Services platform to provide the storage repository protected by the standardized authorization framework utilized by users via the Webmail application.

A consolidated access control and a new data exchange mechanism leverages email security and enhances email system utilization. The question arrives whether a standard implementation of UMA can be integrated into the current email ecosystem.

## Future Work

The combination of the UMA framework with email system outcomes in a new data exchange technology that predestine email system to become more than a bare messaging tool.

The following are potential future R&D areas:

- Explore the upcoming standardized UMA Relationship Manager (aka Wallet) Policy API vs. the proprietary Policy API.
- Consider a Consent mechanism extension design.
- Explore linked content using a clickable hyperlinks – linking content across the business.
- Design an extension for exchanging tagged messages and attachments – grouping content across the business.
- Design an attachment versioning extension – the attachments with the same content are versioned.
- Explore health information exchange between healthcare professionals and inspect use of email communication between patients and healthcare professionals.
- Employ regular email clients and applications using JMAP as a standardized email API.

A prototype implementation of the proposed solution, working as a proof of concept, would be interesting to build.

## About the Author

Igor Zboran is a mechanical engineer by education with professional experience as a software engineer and solutions architect. He'd like to transform his knowledge into a useful system or service that people would love to use.

Igor received Ing. degree in Mechanical Engineering from the University of Žilina in 1988. After graduating, he worked in several small private companies as a software developer. From 2008 to 2009, he provided expert advice to Prague City Hall IT department as an external consultant. He invented a new decentralized Identity-Based Privacy (IBP) trusted model built around OAuth2 and OpenID Connect standards. Igor is a strong proponent of open source software and open standards.