beame.io

Product: Secure Remote Access to a Local Server or IoT Device (beame-gatekeeper)
February 1st, 2017 | v1.01

The purpose of this paper is to describe the particular beame-gatekeeper use case as a tool for remote access to enterprise networks or IoT devices with mobile authentication. It provides an overview of possible product integration options. Other use cases are available at www.beame.io.

Introduction

Secure remote access to devices on private networks is a universal need. Private networks are typically behind a NAT, firewall, and otherwise restricted, making access difficult. The traditional virtual private network system requires network configuration and subsequent security testing, especially in the mobile context. This makes it difficult to roll out applications. Beame.io solutions can make network devices available globally with one command in solutions that deploy on-premises or in restricted networks. *beame-gatekeeper* makes applications securely available to those that possess the correct credentials.

beame-gatekeeper can be installed on any level of a Beame Network Hierarchy (BNH). This hierarchy is detailed in the document, “Understanding the Beame.io Network’s Credential Hierarchy.”

In this paper, we present an example which demonstrates how the usage of publicly trusted TLS certificates allows new workflows and a significantly improved authentication experience for the end user. The example demonstrates how to expose applications to both global and local environments. It uses TLS for end-to-end encryption and client certificates located on the end user’s mobile devices for the granting and verification of access rights.

End-to-end encryption means that the traffic is opened only on the destination device, and can’t be decrypted by a third party while in transit.

Technical Overview of beame-gatekeeper

The *beame-gatekeeper* enables a new way to access systems behind a firewall while enforcing a very strict user authentication policy. It protects the origin servers by never exposing them to incoming internet traffic.

Access is granted to known *beame-crypto-IDs* (which are tied to a user device). A local server can recognize and trust a remote user based on a unique *beame-crypto-ID*. This creates a closed-circuit cryptographic authentication system

Figure 1: Mobile Device Authenticating Browser Access

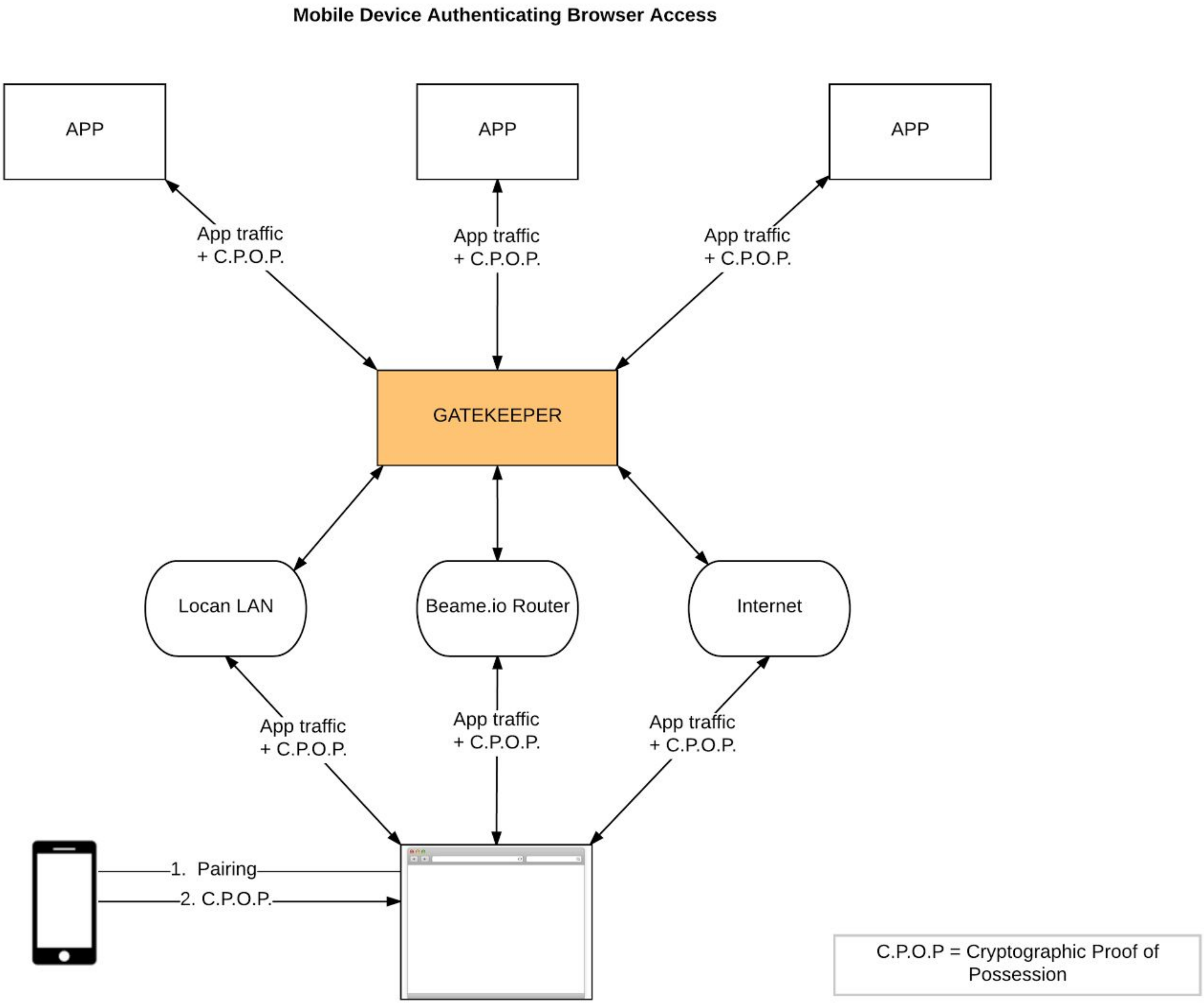


Figure 2: Access to Apps Directly from the Mobile Device

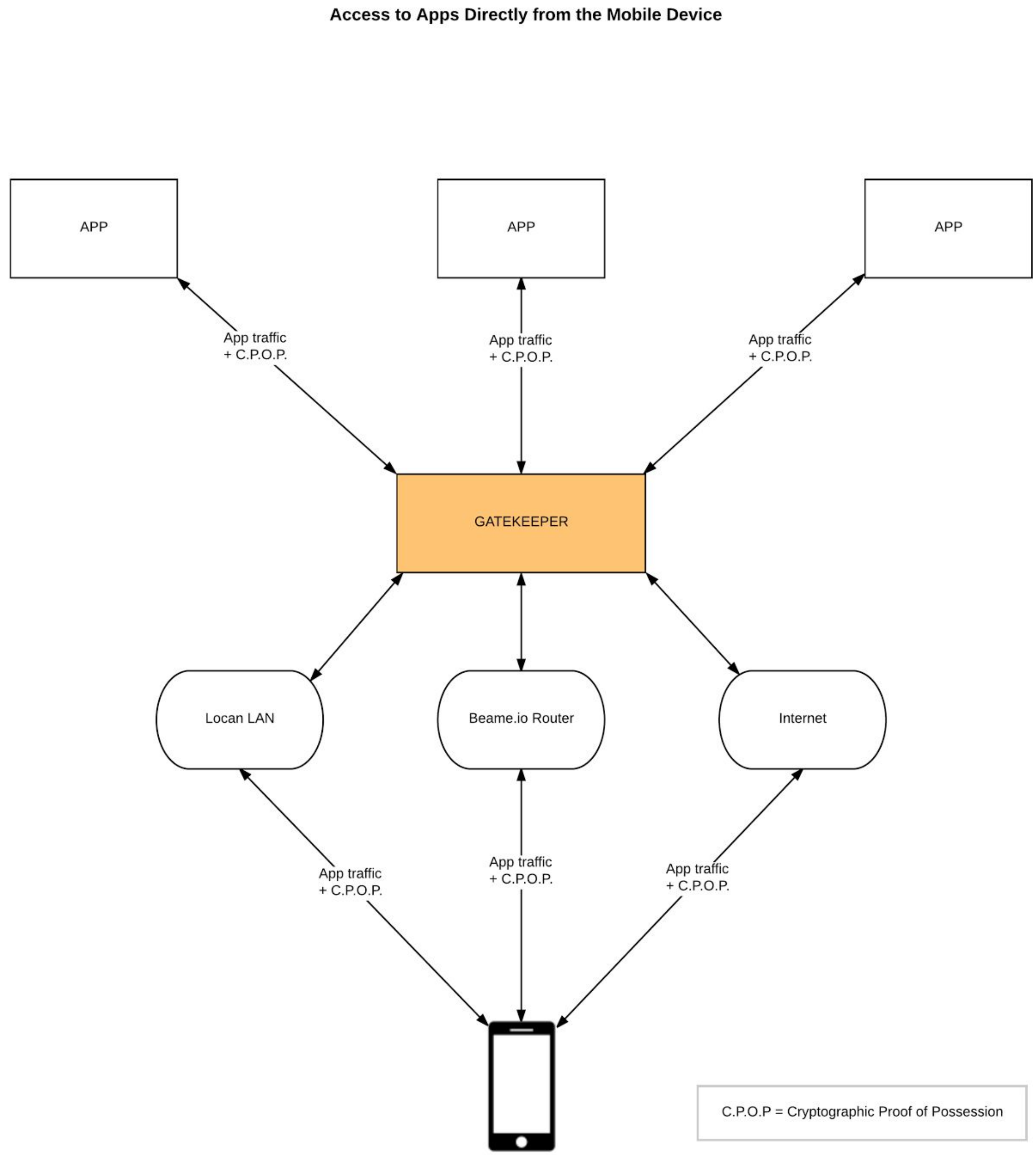
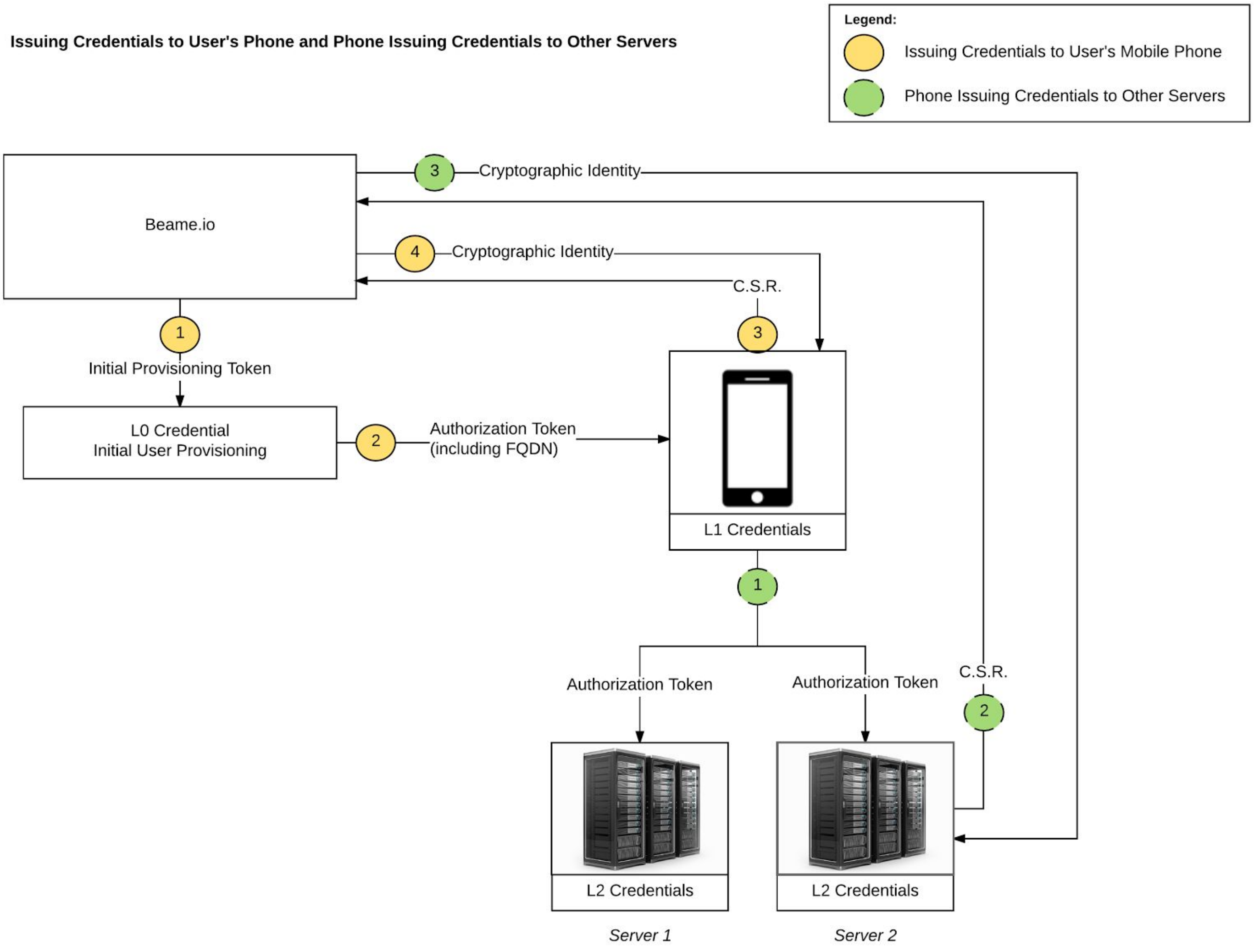


Figure 3: How credentials are issued to the user's phone and how the phone issues credentials to other servers.



beame-gatekeeper enables the on-demand deployment of a powerful remote access tool without exposing the target servers. The tool allows remote access to local servers with an additional authentication feature which requires cryptographic proof of possession of a *beame-crypto-ID* located on the mobile device.

The *beame-crypto-ID* enables unique cryptographic keys to be generated on the recipient device, an FQDN, a publicly trusted SSL certificate, and cryptographic authorization verification.

Mobile Authentication Application

The *beame-authenticator* is the required mobile application that instantly identifies users to *beame-gatekeeper*. There are several integration options: using the application as-is from the app stores,
with a customized skin, or as a dedicated framework for another app (SDK of the mobile application plus source code).

It allows the user to be recognized by providing proof of possession of a device's cryptographic key without revealing it. The app manages secure sessions and log out, multiple sets of credentials and their secure servers. The app is locked with a PIN code which is chosen by the user.

Provisioning

In a process called provisioning, the recipient device first generates private and public keys. Next, the authorization is delivered (matching the keys to the recipient device). The authorization is submitted to the Beame.io CA and the recipient device receives its X.509. Within the scope of *beame-gatekeeper*, this process happens one time on the local target servers to be accessed (origin servers), and on the mobile device of the authorized user using the beame-authenticator application.

Login and Connection Process and Access

There will be one dedicated master URL such as *login.yourcompany.com*, or alternatively *login.beame.io*, and this should be the only URL visible to your customers. This is a company-hosted public login page which provides two ways to pair with the authorized mobile device through the Beame.io APIs: ultrasonic pairing, or by scanning a QR. The use of ultrasonic pairing requires the use of a Beame.io matching server. Upon pairing, the browser is redirected to the FQDN of the mobile device. The mobile device provides proof of possession and authorization of the device private key without ever revealing the key itself. The *beame-gatekeeper* verifies the proof of possession before directing traffic to the origin servers. Optionally, the origin servers may receive the proof of possession, verify it, and map it to a real user identity. This provides a seamless single-sign-on functionality.

Beame Auth Token

The unique token allows the system to control the validity of a session based on both a pre-determined length of time and continuous proof of possession of the proof of identity. It is used throughout the workflow, from initial registration to session establishment and control.

Session Control and Logout

The *beame-gatekeeper* maintains contact with the mobile device and upon expiration of the secure token or inability to renew it, the *beame-gatekeeper* kills the session. Alternatively, a user may log out manually as a mobile-controlled session request.

Blockchain-Based Cryptographic Verification of Authorization

Beame.io allows blockchain-based cryptographic verification that specific private keys were present at the time that an SSL certificate was authorized by such as a server, a person, or a combination. The first machine or device in a network to generate a beame-crypto-ID gets administrative privileges. Upon provisioning, the administrator device (parent node) receives a Beame.io-generated Fully Qualified Domain Name (FQDN) which it uses to fill in the Common Name (CN) field when it generates a certificate signing request to Beame.io. The CN becomes a part of the SSL certificate issued to the device through Beame.io services.

Subsequent *beame-crypto-ID*'s for users (child nodes) under the administrator will contain part of the administrator's CN and their creation will be attributable to the administrator without knowing the administrator's *beame-crypto-ID*. The relationship between the parent and child nodes can be checked because the SSL certificates on the devices are used to sign data.