Universidad de Buenos Aires
Facultad de Ciencias Económicas
Escuela de Estudios de Posgrado

MAESTRÍA EN CIBERDEFENSA Y CIBERSEGURIDAD

TRABAJO FINAL

Implementación de un nodo minero-institucional en la red Ethereum-BFA

ING. JONATAN URAN ACEVEDO
MG. ING. JORGE ETEROVIC

NOVIEMBRE 2021

Índice

i. Resumen

En la actualidad es posible gracias a la tecnología Blockchain dar solución a varios problemas. En este sentido, una Blockchain brinda ventajas como por ejemplo la no necesidad de intermediarios de confianza en una operación económica o legal entre una o más partes, la inmutabilidad del registro de operaciones y sobre todo la transparencia de dichas operaciones.

El presente trabajo consiste en implementar un Nodo Minero (en adelante Sellador) perteneciente a una institución, dentro de la red Blockchain Federal Argentina (BFA).

BFA es una red permisionada montada sobre la Blockchain Ethereum.

Se detallarán los pasos procedimentales y administrativos para obtener los permisos necesarios para montar el mencionado nodo, y luego proceder a la instalación del hardware y software que lo soporte.

Para la implementación se tomará como organización pública a la Universidad Nacional de La Matanza (UNLaM).
Lo que se espera es hacer *parte* a la UNLaM de Blockchain Federal Argentina y montar un nodo sellador dentro de la red, que pertenezca a la UNLaM.

También se desarrollará una Aplicación distribuida (en adelante dApp) que se conectará a un Smart Contract que reside dentro de Blockchain Federal Argentina llamado “sello de tiempo”. Con esto se logrará garantizar la integridad de un documento.

Por otra parte, el nodo será montado en la nube de Amazon Web Services (en adelante AWS) y se utilizará el Marco de Buena Arquitectura de AWS, el mismo será explicado y se brindarán recomendaciones de ciberseguridad.

El fin fundamental del presente trabajo es mostrar de forma práctica, desde el punto de vista de la Ciberseguridad, como a través del uso de una Blockchain se puede dejar un mensaje (en este caso el hash de un documento) inmutable. Lo que da como resultado la autenticidad de un mensaje de forma indiscutible.

Blockchain Federal Argentina (BFA) es la primera plataforma multiservicios abierta y participativa de Argentina. Está pensada para integrar servicios y aplicaciones sobre la Blockchain Ethereum.

La relevancia de este trabajo radica en la importancia que tiene para una institución formar parte de una red con las características de BFA. La UNLaM, hasta el momento, no forma parte de BFA ni de otras plataformas similares.

Por lo dicho, se pretende dejar como valor agregado que la UNLaM tendrá una identidad dentro de BFA.

Esto es extrapolable a cualquier organización pública o privada que quiera formar parte de Blockchain Federal Argentina o de otra red de similares características, por lo que se pretende que este trabajo sirva como guía de implementación de un nodo minero/sellador.

palabras clave: blockchain, ethereum, dapp, smart contract, BFA, UNLaM.

ii. Abstract

Currently, it is possible to solve problems through Blockchain technology use. In this sense, a Blockchain offers advantages such as no need intermediaries in economic or legal operations between one or more parties, the immutability of operations registry and operations transparence.

The present work is about Mining Node implementation (Sealer Node) belonging to an institution, within the Blockchain Federal Argentina (BFA) network.

BFA is a permissioned network mounted on the Ethereum Blockchain.

The administrative steps will be detailed to get authorization to mount a Node, and then install hardware and software to supports it.

For the implementation, Universidad Nacional de La Matanza (UNLaM) will be taken as a public organization.
The objective is UNLaM will be part of the Blockchain Federal Argentina and set up a Sealer Node within the network, which belongs to the UNLaM.

A distributed Application (dApp) will also be developed and will be connected to Smart Contract (this is in the Blockchain Federal Argentina) called "Sello de tiempo" (Timestamp). This will ensure the integrity of documents.

On the other hand, the node will be mounted in the Amazon Web Services Cloud (AWS) and the AWS Well Architecture Framework will be used, it will be explained, and cybersecurity recommendations will be provided.

Blockchain Federal Argentina (BFA) is the first multiservice platform in Argentina. It is designed to integrate services and applications on the Ethereum Blockchain.

It is especially important for UNLaM to be part of BFA. Before this project UNLaM was not part of the BFA or other similar platforms.

iii Dedicatoria

A mi madre que me enseñó a leer y a sumar y todos los días me enseña a caminar sin caer. Algún día voy a aprender a manejar un hogar como vos.

A mi padre la persona más humilde y noble que conozco, viejo intento imitarte, pero aún no lo logro.

Viejos les debo todo lo que soy.

A mi hermanita, la persona más transparente y dulce que conozco, con tus abrazos me llenas el alma. Disfruto verte crecer.

A mi hermano con quien tengo una gran conexión y con una mirada o un silencio nos comunicamos más que con una palabra.

iv. Agradecimientos

En primer lugar, quisiera agradecer a Gendarmería Nacional Argentina por haber financiado gran parte de mi Posgrado y haberme dado la oportunidad de crecer y aprender desde mi lugar, de aquellos que dirigen la Fuerza y quienes ejecutan esas órdenes. Honor y Gloria.
Dedico este Trabajo a mi Comandante Mayor Don Héctor Molinero, quien siempre confió en mí y me enseñó a ser cada día un mejor Oficial. Vuela alto, algún día nos encontraremos.

A muchos otros que la Ley de Inteligencia Nacional me impide nombrar y aunque así no fuese, elegiría igualmente conservar el secreto debido al rol que ejercen dentro del Sistema de Inteligencia de la República Argentina. Los tengo en mi mente a todos y cada uno.

Todos me permitieron comprender qué es la Inteligencia Criminal en la práctica.

Quisiera agradecer al director de la Maestría Prof. Dr. Roberto Uzal, su sabiduría es incalculable para nuestro país, lo digo realmente convencido. Una vez dijo: "Es altamente importante la interacción entre ustedes (los alumnos de la Maestría), que son quienes de aquí en adelante seguirán trabajando en este camino". Me llevo esa frase conmigo.

También quisiera agradecer al Prof. Ing. Carlos Amaya. Mi Comandante, usted me ayudó brindándome su tiempo por fuera de las clases, a desafíos que se me presentaron como funcionario público, aclarándome el panorama para tomar decisiones con más claridad y conocimiento.

A todos los profesores de la Maestría, algunos de los cuales participaron defendiendo nuestro pabellón en el campo de las armas. A ustedes, gracias por defender a mi país. Saludo uno.

Por último, agradecer a mi director de Tesis Ing. Jorge Eterovic, con quien tengo la oportunidad de seguir trabajando en el ámbito académico, bajo sus órdenes, formando parte de su equipo.

1. INTRODUCCION

Es innegable el auge que tienen actualmente las criptomonedas en la sociedad. Su potencial se debe a la revolucionaria tecnología que las soporta denominada Blockchain.

La primera criptomoneda en aparecer fue Bitcoin. La Blockchain de bitcoin permite utilizarlo como resguardo de valor, ese es su objetivo.

Otra de las criptomonedas con mayor potencial que usan la tecnología Blockchain es Ethereum. Si bien Ethereum tiene una criptomoneda llamada Ether (ETH) con la cual se puede transaccionar económicamente, el objetivo de Ethereum es crear una Blockchain programable, que permite almacenar programas en su interior que sirven como columna vertebral de las llamadas aplicaciones descentralizadas (dApps). Estos programas se denominan Smart Contracts, en español son conocidos como contratos inteligentes.

1.1 Fundamentación

La elección del tema a tratar se debe a la posibilidad de mostrar de forma práctica cómo hacer una implementación de un nodo complejo en una Blockchain. Además, se mostrará también de forma práctica cómo funciona una dApp.

El presente trabajo está motivado por la falta de información clara y concisa al respecto, sobre todo en idioma español y aún más, procedente de ámbitos académicos.

Si existe un escenario que no está libre de riesgos y amenazas, es la Red de Redes. La Ciberseguridad y la Ciberdefensa cuentan con poderosas herramientas entre ellas el uso de Blockchain que permite la irrefutabilidad de la información debido a que es prácticamente imposible la modificación de la información contenida en ella sin el conocimiento de los miembros de la red.

Mediante el uso de una Blockchain es posible alcanzar un consenso por parte de todos los participantes de la red, sobre la integridad de los datos contenidos en la misma, sin necesidad de recurrir a una entidad de confianza que centralice la información.

Este trabajo describe el funcionamiento de una Blockchain, pero fundamentalmente explica cómo implementar uno de los nodos fundamentales que son los encargados de armar y sellar los bloques de transacciones y añadirlos a la cadena, es decir, nada más y nada menos que los nodos formadores de la Blockchain.

También se desarrollará una dApp que se conectará a un Smart Contract llamado “sello de tiempo”. Con esto se logrará garantizar la integridad de un documento.

Y todo esto se desarrollará siguiendo el Marco de Buena Arquitectura (WAF) recomendado por AWS.

Para llevar esto a cabo, se ha tomado como plataforma a Blockchain Federal Argentina que destaca por ser la primera red de nodos de la República Argentina, de alcance masivo a organismos públicos y privados.

1.2 Planteamiento del Problema

En el presente Trabajo se implementa un Nodo Sellador, dentro de Blockchain Federal Argentina, es decir, se detallan los pasos procedimentales y administrativos para obtener los permisos necesarios para montar dicho nodo, y se explica cómo se debe realizar la instalación del hardware y software que soporte el nodo.

Los nodos selladores, son los encargados de procesar las transacciones, agregarlas a un bloque y finalmente añadirlas a la cadena de bloques (Blockchain).

Cabe aclarar que en el presente Trabajo hablamos de Nodo Sellador y NO Nodo Minero como coloquialmente se suele llamar a los nodos que añaden bloques a la Blockchain. La diferencia está en el protocolo de consenso utilizado por los nodos. En BFA se utiliza PoA, en el cual los nodos no compiten por sellar, sino que se turnan para hacerlo, en este caso los nodos que se encargan de sellar se denominan Selladores (Sealers). Y para el caso en que los nodos compiten por sellar, a través de la resolución de un algoritmo, en un protocolo de consenso PoW, los nodos se denominan mineros.

El mencionado nodo sellador pertenece a una institución pública.

Para la implementación se tomó como organización pública a la Universidad Nacional de La Matanza (UNLaM). Esto significa que se hizo formar *parte* de Blockchain Federal Argentina a la UNLaM y se generó un nodo sellador dentro de la red, perteneciente a dicha institución.
Todo esto se llevó a cabo siguiendo el Marco de Buena Arquitectura (WAF – Well Architecture Framework) recomendado por Amazon Web Services (AWS).

También se desarrolló una Aplicación distribuida (en adelante dApp) que se conecta a un Smart Contract dentro de Blockchain Federal Argentina llamado “sello de tiempo”. Con esto se logró garantizar la integridad de distintos documentos.

El Trabajo Final intenta crear una guía detallada que explique cómo montar un Nodo complejo en una Blockchain; desde los pasos administrativos y procedimentales hasta la instalación, montaje y programación del nodo y seguido a esto, el desarrollo de una dApp que muestra cómo se comunica con la red.

Está motivado por la falta de información detallada al respecto sobre todo en idioma español y más aún, procedente de ámbitos académicos.

2. MARCO TEÓRICO

En principio me gustaría contextualizar el problema que pretendo resolver. Por ello primero voy a hablar acerca de las Organizaciones que estarán involucradas y cuál será su rol. Seguido a esto, voy a hablar sobre la terminología existente dentro del dominio del problema que resolverá este trabajo.

2.1 Contextualización

¿Qué es BFA?

Blockchain Federal Argentina es una plataforma multiservicios abierta y participativa pensada para integrar servicios y aplicaciones sobre Blockchain. Una iniciativa confiable y completamente auditable. (Blockchain Federal Argentina, 2017)

Blockchain Federal Argentina fue diseñada para potenciarse a través de los aportes de sectores públicos, privados, académicos y de la sociedad civil.
BFA opta por una estrategia donde la participación de toda la comunidad es esencial, desde la ingeniería organizacional hasta el despliegue de la infraestructura.

Siguiendo el modelo de Múltiples Partes Interesadas, Blockchain Federal Argentina mantiene un modelo de gobernanza que asegura la representación de todos los sectores en la toma de decisiones. Pero al ser una plataforma pública, su uso no estará restringido a las organizaciones que participan del consorcio. Toda la comunidad tiene las puertas abiertas para participar en BFA.

Individuos, organismos, instituciones o empresas de cualquier sector interesados en desplegar aplicaciones y servicios aprovechando todas las características de la plataforma, o simplemente en contribuir al primer desarrollo de esta índole en el país, pueden sumarse.

Hay dos formas de integrarse a Blockchain Federal Argentina: como *usuarios* de servicios o como *parte* de la organización.

Los *usuarios* de servicios, pueden ser organizaciones o individuos, que utilizan Blockchain Federal Argentina para desplegar sus aplicaciones sobre esta plataforma, aprovechando todas las ventajas que implica el funcionamiento sobre una Blockchain y en particular sobre la BFA.

Las organizaciones (tanto públicas como privadas) pueden formar *parte* de Blockchain Federal Argentina, ejerciendo distintos roles de control dentro de la organización. También pueden desplegar aplicaciones sobre la plataforma.

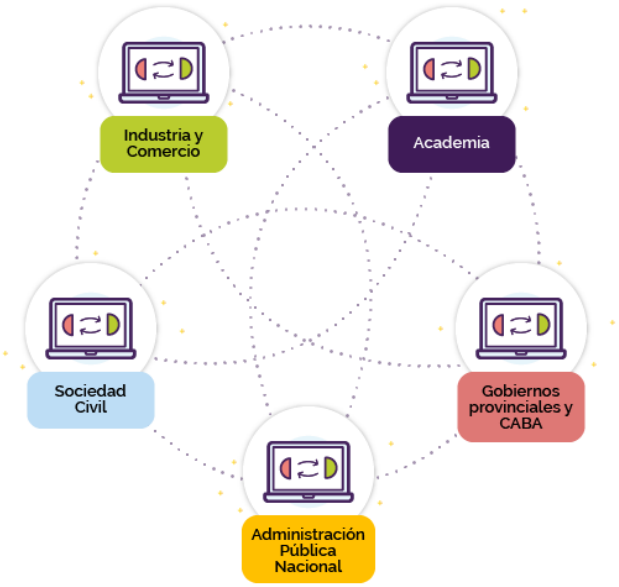

Ilustración 1. Composición de BFA

El diseño tanto técnico como de gestión de Blockchain Federal Argentina no solo fue pensado para garantizar que la iniciativa fuera escalable gracias a la incorporación de nuevos participantes, sino también para asegurar su continuidad en el tiempo: que perdure más allá de las personas e instituciones que lo gestaron gracias a un modelo de trabajo horizontal y colaborativo.

¿Qué es la UNLaM?

La Universidad Nacional de La Matanza (UNLaM) es una universidad pública Argentina. Fue fundada el 29 de septiembre de 1989 en San Justo, partido de La Matanza, provincia de Buenos Aires.

Ilustración 2. Universidad Nacional de La Matanza, calle principal.

Es la segunda universidad más grande de la provincia de Buenos Aires, detrás de la Universidad Nacional de La Plata, en cuanto a cantidad de alumnos, ya que tiene más de 50 mil alumnos.

Depende financieramente del Estado nacional, pero es autónoma, como toda Universidad Nacional.

La división académica de la UNLaM es a través de Departamentos, donde se agrupan las diferentes disciplinas afines y las áreas de investigación, la Escuela de Formación Continua

que permite completar estudios terciarios y la Escuela de Posgrado que permite profundizar los conocimientos a los profesionales de grado.

La Universidad Nacional de La Matanza está compuesta por los departamentos:
Departamento de Ingeniería e Investigaciones Tecnológicas.
Departamento Humanidades y Ciencias Sociales.
Departamento Ciencias Económicas.
Departamento Derecho y Ciencia Política.
Departamento De Ciencias de la Salud.

Yo tengo el honor de ser un graduado de la UNLaM. Cursé mis estudios en el Departamento de Ingeniería e Investigaciones Tecnológicas.

¿Qué es Amazon Web Services?

Amazon Web Services (AWS) es una colección de servicios de cómputo en la nube ofrecidas por la empresa Amazon.
Es usado en aplicaciones populares como Dropbox, Foursquare, HootSuite. Es una de las ofertas internacionales más importantes de la computación en la nube y compite directamente contra servicios como Microsoft Azure, Google Cloud e IBM Cloud. Es considerado como un pionero en este campo

Informática en la nube es la entrega bajo demanda de potencia de cómputo, bases de datos, almacenamiento, aplicaciones y otros recursos de TI, a través de internet con un sistema de precios de pago por uso

2.2 Conceptos Fundamentales

¿Qué es Blockchain?

Blockchain, en español, cadena de bloques, es una tecnología que permite administrar un registro de datos en la nube. Tiene como característica la transparencia y es prácticamente incorruptible.

Una Blockchain puede ser vista como un gran libro contable. Allí sólo pueden ingresar nuevas entradas y las entradas anteriores no pueden ser modificadas ni eliminadas.
Esas entradas se llaman transacciones, las cuales se van agrupando en bloques que se agregan sucesivamente a una cadena.
Cada uno de los bloques hace referencia al bloque inmediatamente anterior de modo que, si alguna transacción pudiera ser modificada, esa referencia cambiaría y ese bloque y todos los posteriormente agregados serían inválidos.

Por lo anteriormente dicho, si queremos corregir información ya registrada, solo lo podemos hacer mediante el agregado de nueva información. Los datos originales siempre van a permanecer en la cadena y pueden ser inspeccionados en cualquier momento.

Una Blockchain puede ser vista también como una base de datos distribuida que contiene un histórico irrefutable de información.

La Blockchain (esa cadena de bloques compuesta por transacciones) no se encuentra almacenada en un solo servidor centralizado, sino que se encuentra replicada en un gran conjunto de dispositivos conocidos como nodos que conforman lo que se conoce como red de pares (P2P).

Cada vez que se agrega una nueva transacción, ésta se agrega a un bloque y posteriormente este bloque se agrega a la cadena y por último ésta es actualizada en todas las réplicas que guardan cada uno de los nodos.

Una Blockchain no solo está protegida por este modelo de red descentralizada, sino que también está atravesada por métodos criptográficos que garantizan que nada pueda ser borrado o alterado sin que todos los nodos que la componen puedan darse cuenta de ello.

Una Blockchain permite garantizar la identidad de las partes involucradas, ya que todas las transacciones son firmadas criptográficamente.

Se puede certificar la fecha y hora de cada transacción. La información es inalterable. Además, toda la información almacenada en la cadena es completamente auditable.

Una Blockchain funciona sin intermediarios, esto es, no hace falta una persona, empresa o institución que legitime la información guardada en la cadena.

¿Qué es Ethereum?

Ethereum es una plataforma descentralizada de código abierto (Open Source), que permite la creación de contratos inteligentes sobre una Blockchain. (Ethereum, 2016)

En diciembre de 2013, Vitalik Buterin comenzó el desarrollo de Ethereum, con la primera prueba de concepto (PdC).

Existe una bifurcación de la cadena de bloques Ethereum a partir de julio de 2016, que dio como resultado dos líneas de Ethereum activas: Ethereum y Ethereum Clásico.

Ethereum funciona a través de una máquina virtual llamada Ethereum Virtual Machine (EVM). Esta máquina ejecuta un código intermedio o bytecode el cual es una mezcla de LISP, ensamblador y bitcoin script.

Los programas que realizan contratos inteligentes son escritos en lenguajes de programación de alto nivel como Solidity.

Ethereum utiliza como criptomoneda al Ether, esta sirve como intercambio de valor y también sirve como "combustible" para ejecutar los contratos inteligentes.

Smart Contracts

Un Smart Contract (Contrato Inteligente en español) es simplemente un bloque de código (software) que reside en una Blockchain y que se ejecutará cuando algo lo invoque.

Por ejemplo 'Sello de Tiempo' es un Smart Contract que reside en la Blockchain de BFA y que se ejecuta cuando nosotros lo invocamos para sellar documentos. En verdad, lo que hace es enviar un hash a un Nodo Transaccional y éste último arma una transacción con esto y lo envía a la red, para que luego un Nodo Sellador lo coloque dentro de un bloque y lo selle, formando así parte de la Blockchain.

Los Smart Contracts fueron propuestos en 1996 por Nick Szabo, un científico informático, jurista y criptógrafo muy ligado a Bitcoin. Buscaba ofrecer una herramienta que permita formalizar acuerdos entre partes para entornos de redes de computadoras como por ejemplo internet. De ahí el origen del nombre, poro hoy en día no necesariamente representan acuerdos.

Los Smart Contracts al estar dentro de la Blockchain se encuentran distribuidos en cada copia que almacenen los nodos que forman parte de la red, por esto tienen la característica de ser distribuidos también.

En un principio fueron pensados para que dos partes que no se conocen establezcan una serie de compromisos mediante una Blockchain sin necesidad de un tercero confiable. Si las condiciones no se cumplen por las partes, el contrato no se liquidará. Un proceso mucho más sencillo, sin necesidad de terceros confiables, que reduce los costos y tiempo de constitución.

Como dije anteriormente los Smart Contract requieren de una Blockchain donde almacenarse y ejecutarse, ya que realmente son fragmentos de código. En Ethereum los Smart Contracts se ejecutan y administran operaciones que se producen cuando direcciones (usuarios u otros Smart Contracts) interactúan entre ellos.

Todas las direcciones (de usuarios) en una Blockchain que no son un Smart Contract reciben el nombre de cuenta externamente controlada (Externally Owned Account; EOA).

Un Smart Contract en cualquier Blockchain se basa en un código que realiza determinadas acciones y un conjunto de claves públicas. Se deben dar al menos dos claves públicas, la del creador del contrato inteligente y la del propio contrato, que hace el papel de identificador único.

Se realiza luego una ejecución de este mediante una transacción en la Blockchain. El Smart Contract solo se ejecutará cuando sea llamado por una EOA o por otros contratos inteligentes.

Sería bueno en este punto enumerar las principales características de los Smart Contracts:

Distribuidos: Cualquier Smart Contract es replicado y distribuido por todos los nodos conectados a la red, ya que los mismos una vez que son subidos, residen en la Blockchain. Se garantiza que todos tengan una copia de las condiciones establecidas y no se puedan cambiar a voluntad de ningún usuario, incluso aquel que lo subió a la Blockchain. Esto es porque ninguna información presente en una Blockchain puede ser modificada.

Determinísticos: Únicamente pueden realizar acciones para las que ha sido diseñado, pero solo cuando las condiciones dadas se cumplen. El resultado final no variará nunca, sin importar quien lo ejecute.

Autónomo: Tienen la capacidad de automatizar cualquier tipo de tarea, funcionando como un programa con autoejecutables. Si un Smart Contract no recibe una activación (una llamada), éste permanecerá en espera y no hará nada.

Inmutable: Cuando son lanzados a la red estos ya no pueden ser modificados. Una vez desplegados solo pueden ser eliminados, siempre y cuando se haya implementado esta función particular previamente.

Trustless: Estos no requieren de terceras partes de confianza que verifiquen la integridad del proceso y que se cumplan las condiciones marcadas.

Transparentes: Se almacenan siempre en una Blockchain, así que el código puede ser visto por todos, sean o no participantes en el Smart Contract

Ahora veamos algunas características adicionales de los Smart Contracts

Los Smart Contract tienen la capacidad de ser eliminados, si así se ha programado. Dentro de la Blockchain de Ethereum los Smart Contract pueden añadir una función en el código que es: SELFDESTRUCT. Esta implementación en el código permite borrar el contrato inteligente en el futuro si se detecta un error o sustituirlo por uno nuevo. Si no se añade este elemento, el Smart Contract jamás se podrá borrar.

Un Smart Contract NO se puede modificar. Una vez que el Smart Contract está lanzado no puede ser modificado. Normalmente lo que se hace es que un contrato inteligente realice llamadas a otros contratos inteligentes con funciones determinadas. Esto nos permite borrar un Smart Contract (si hemos habilitado la opción de borrado) concreto para sustituir un Smart Contract por otro nuevo más completo o mejorado.

Quizás surja la pregunta ¿Qué se puede crear con un Smart Contract? Y la respuesta sería que el límite de las capacidades de un Smart Contract lo pone la capacidad de creación de las personas y la habilidad de un/unos programadores para crear el código.

Bitcoin presenta algunas dificultades en cuanto a la creación y despliegue de Smart Contracts. Ethereum por su parte permite la creación, despliegue y ejecución de Smart Contracts sin límites técnicos gracias a su EMV de tipo Turing completo, que a grandes rasgos funciona como una gran computadora descentralizada.

Ethereum permite crear Smart Contracts muy ricos y que permiten hacer una enorme cantidad de cosas. Se pueden crear organizaciones autónomas descentralizadas (DAO), aplicaciones descentralizadas (DApps), tokens fungibles y no fungibles (NFT, por ejemplo), aplicaciones de finanzas descentralizadas (DeFi), exchange descentralizadas (DEX) y todo lo que podamos imaginar.

Veamos también algunos *problemas* que traen los Smart Contracts
Son código. Cualquier software, juego o sistema operativo tienen código y es fácil que presenten diferentes tipos de problemas. Los Smart Contracts no son más que código y son desarrollados por personas, así que pueden tener fallos que pueden terminar siendo catastróficos.

Un caso emblemático fue The DAO, una organización autónoma descentralizada creada basándose en un Smart Contract que fue hackeada en 2016 debido a un fallo en el contrato inteligente desarrollado para su creación. Esto permitió a un atacante robar nada más y nada menos que 3.6 millones de ether (moneda de la Blockchain de Ethereum).

Precisamente que estos contratos inteligentes sean inmutables hace que un fallo en el código no se pueda corregir. Es por esto por lo que normalmente si se quiere crear un Smart Contract muy complejo se divida en una cantidad de estos, por si alguno presentara algún fallo de seguridad poder suprimirlo y lanzar uno nuevo que funcione correctamente.

Debe quedar claro que, pese a su nombre, no son ni un contrato (en el sentido convencional del término), ni son inteligentes. Son simplemente código que tiene una serie de condiciones establecidas por el desarrollador de este.

La particularidad que los hace tan atractivos es que son inmutables, por lo que una vez lanzados no se pueden modificar las variables que lo rigen. Aunque se pueden eliminar, siempre que se habilite esta función y si se detecta algún problema en el código o se quieran añadir funcionalidades. (Solé, Profesional Review, 2021)

Red p2p

Las redes P2P o Peer-to-Peer son un tipo de redes de computadoras descentralizadas. Estas son redes que están formadas por cientos, miles e incluso millones de computadoras ubicadas en todo el mundo. Todas ellas funcionando bajo un mismo protocolo de comunicaciones, con el objetivo de crear una enorme red para compartir información de cualquier índole. Si hoy podemos conectarnos a una Blockchain es gracias a este tipo de red.

Una red P2P, es una red donde individuos o máquinas participan de forma completamente descentralizada. Es decir, es una red donde no hay un punto central de conexión o control, y donde las partes actúan de forma autónoma respondiendo a un protocolo de comunicaciones y consenso común. De esta forma, los integrantes de la red pueden intercambiar información de forma directa y sin intermediarios.

Para esto, las redes P2P se construyen sobre protocolos que se ejecutan sobre Internet (también conocido como TCP/IP). De allí viene que a los protocolos P2P se los catalogue como protocolos de aplicación o de Capa 7, según el modelo Open Systems Interconnection (OSI). Esto significa que los protocolos P2P necesitan para su funcionamiento, el uso de otros protocolos a los fines de poder funcionar, pero que, al mismo tiempo, los hace más sencillos.

Los protocolos P2P, han sido ampliamente utilizados desde su creación para distintos usos. Los protocolos P2P son muy potentes y permiten la creación de estructuras descentralizadas, difícilmente censurables y de uso libre. Por esa razón, criptomonedas como Bitcoin fueron construidas sobre la base de protocolos P2P.

Sería bueno en este punto, conocer un poco más sobre estos protocolos y su evolución.

Origen del P2P

El origen de la primera red P2P lo podemos encontrar en la creación del protocolo UUCP o Unix to Unix Copy Protocol, en el año 1980. Este protocolo dio origen a la conocida red USENET y, a los BBS, redes que hoy en día aún se encuentran activas y funcionando.

El principio de funcionamiento de estos sistemas es sencillo: la máquina realiza una llamada de conexión (conexión por marcación) usando un modem, se comunica con la maquina destino, y pueden compartir información punto-a-punto sin intermediarios. Al terminar la llamada de conexión, el usuario puede iniciar otra conexión con otra máquina comenzando el proceso nuevamente. Cabe destacar que, todo esto es posible sin una estructura como la del Internet que conocemos hoy, una red y tecnología que para ese entonces (1980) aún estaba en desarrollo.

Más tarde, en 1983, llegó al mundo el protocolo TCP/IP. Básicamente, este nuevo protocolo buscaba flexibilizar la creación de grandes redes globales, de hecho, es la base de construcción de lo que hoy conocemos como Internet. Esto último, se hizo realmente posible con la llegada del protocolo WWW (World Wide Web) en 1990.

Tipos de redes P2P

Entre los tipos de redes P2P existentes podemos encontrar:

Red descentralizada y estructurada. Este tipo de redes son conocidas como redes P2P híbridas. En este tipo de redes no existe un servidor central, sino que en su lugar existen una serie de nodos o peers, que tienen la capacidad de recibir peticiones de información y responder a las mismas para facilitar el acceso a los recursos. Para evitar la centralización de esta funcionalidad, los nodos o peer especiales pueden ser instalados y configurados por cualquier persona, buscando con ello que la misma comunidad de usuarios extienda la funcionalidad de la red y permita su correcto funcionamiento. Un buen ejemplo de este tipo de redes es la red federada Mastodon.

Red descentralizada y no estructurada. En este tipo de redes P2P no existen computadoras o nodos que funcionen como controladores centrales de peticiones. Por el contrario, cada

nodo dentro de la red tiene las mismas funciones que el resto de los nodos, por lo que cada nuevo nodo ejerce la misma autoridad que el resto. En este punto, redes como Bitcoin cumplen con esta característica, puesto que cada nodo conectado tiene las mismas capacidades que el resto.

¿Cómo funciona una red P2P?

El funcionamiento de una red P2P es sencillo. Básicamente lo que se hace es construir un protocolo (conjunto de pasos preestablecidos) de comunicaciones que permita a las computadoras que usan dicho software comunicarse de forma directa y sin intermediarios. Sin embargo, el mayor problema frente a la construcción de estos sistemas es; ¿Cómo diseñar un sistema que no necesite una computadora que dirija la comunicación entre computadoras que ejecuten el mismo software?

El problema es complejo, pero la situación se puede solventar de una forma efectiva con dos medidas bien definidas:

En primer lugar, hacer que el software sea capaz de compartir información de conexión sobre quienes ejecutan el mismo. Así, cada computadora que ejecuta el software es capaz de tener una carpeta compartida y permitir la conexión del nodo que lo solicita.
En segundo lugar, incentivar la mayor descentralización posible de la red. Es decir, hacer que muchas personas ejecuten el software creando sus propios nodos, aumentando el tamaño de la red. De esta manera, se mejora su alcance y las posibilidades de esta.
Es decir, mientras más pares o peers (computadoras ejecutando el software P2P) tenga la red, más posibilidades hay de que la red no pueda ser censurada, su funcionamiento será más resistente y, la misma tendrá mejores capacidades. En los primeros sistemas P2P, los sistemas y sus conexiones se hacían conocer por medios escritos, llamadas, o el mismo sistema que tenía un tablero de pares a los cuales poder conectarse. Así, cada nuevo integrante de la red tenía acceso a la lista de peers y se auto añadía para que otros pudieran establecer comunicación con ellos en caso de requerirlo. No solo eso, ese nuevo peer podía ser la puerta de entrada a la información de peers que pudieran bloquearse.

Pero la creación de redes más grandes como el IRC, DCC, DC++, Napster, Gnutella, BitTorrent e incluso Bitcoin cambió drásticamente esto. Ahora cada nodo se conecta a un punto, obtiene una lista de peer iniciales y a partir de allí, cada nodo es capaz de recrear una lista propia de nodos que pertenecen a la red. Como resultado se obtiene una mejor resistencia a la censura y la red puede crecer más rápidamente.

Por supuesto, el funcionamiento de cada protocolo es distinto. IRC, por ejemplo, es un sistema distribuido (casi centralizado) de servidores que pueden otorgar la capacidad de conectarse punto a punto con una persona. Pero DC++ y Gnutella, son completamente descentralizados, sus redes están pensadas para que, de forma automática, la red se ajuste con la entrada y salida de nuevos nodos a la red.

Lo mismo pasa en Bitcoin, donde la red comenzó con una sola semilla, la iniciada por Satoshi Nakamoto, y desde entonces, la red ha ido creciendo paulatinamente para convertirse en una red con un tamaño superior a los 10 mil nodos activos. Por supuesto, el objetivo de Bitcoin es distinto al de una red como Gnutella, pero los principios del protocolo se mantienen: comunicar a dos partes sin intermediarios.

Es interesante profundizar un poco más sobre P2P, haciendo hincapié en las ventajas y desventajas de esta tecnología.

Ventajas

Una red P2P es resistente a la censura. Una red P2P altamente descentralizada es prácticamente imposible de censurar.

Ofrecen una resiliencia inigualable. Si un nodo cae, otro nodo puede tomar su lugar. Por eso se dice que las redes P2P pueden sobrevivir a una catástrofe nuclear, porque estas pueden destruir muchos nodos, pero si solo uno sobrevive, la red puede reconstruirse por completo.

Las redes P2P pueden llevar a soluciones de escalabilidad potentes para presentar servicios únicos con alcance global.

Al no depender de entidades centrales, las P2P generan más confianza en sus usuarios.

Ofrecen un alto nivel de ancho de banda. Esto gracias a que aprovechan el ancho de banda de cada participante, para transformarlo en propio de la red.

Sirven para transmitir información digital de cualquier tipo. Desde un archivo a cientos de millones de dólares, en segundos.

Desventajas

Una red P2P es resistente a la censura, pero realmente no es anónima a menos que esté diseñada para ello, incluso, si esa red usa cifrado. El mejor ejemplo es BitTorrent, donde los ISP pueden detectar el uso del protocolo, y con ello advertir a las autoridades de la descarga ilegal por parte de un usuario.

El diseño de las redes P2P generan que a mayor tamaño aumente la latencia. Es decir, para que una información llegue a todas las partes que forman la red, se tomará más tiempo en una red P2P de gran tamaño que en una de menor tamaño. De allí que se busquen nuevos algoritmos y protocolos que ayuden a superar este problema.

Los protocolos P2P tienen una serie de problemas estructurales conocidos. Casos como los ataques MITM para tomar el control de nodos, debido a que estos deben estar conectados todo el tiempo de forma pública. También los protocolos son susceptibles a ataques de enrutamiento o cosas tan sigilosas como un Eclipse attack o un Erebus attack.

Por último, me gustaría hablar sobre Bitcoin como una red P2P para manejar valor.

Bitcoin es una de las redes P2P más grandes que existen en la actualidad, con sus más de 10 mil nodos activos, Bitcoin es una red global que permite a sus usuarios manejar valor sin intermediarios. Lo único que debe hacerse es descargar un software que permita interactuar con esta red.

La construcción de Bitcoin como una red P2P responde a la necesidad de descentralizar sus capacidades. De nada vale crear una moneda con criptografía, si esta luego es manejada por una entidad central. Eso sería simplemente crear un nuevo banco central. En su lugar, Satoshi Nakamoto deseaba una red global, incensurable, segura y privada que permitiera manejar valor. Así que, para ello, Nakamoto diseño Bitcoin sobre la base de una red P2P usando un protocolo propio diseñado bajo los principios del protocolo Kademlia y protocolo Gossip.

El resultado, es que Bitcoin es una red P2P que, prácticamente, no se puede detener. Incluso con sus fallas estructurales, el protocolo P2P fue la mejor decisión que Nakamoto pudo tomar en el diseño de Bitcoin. Con ello se aseguró de crear un dinero digital que sirviera a los intereses del mundo y de sus usuarios.

Lo mejor de todo, es que Bitcoin con su sistema P2P ha logrado crear un sistema de contabilidad distribuido en el que problemas como el doble gasto son cosas del pasado. El doble gasto, era uno de los problemas principales del dinero digital. La posibilidad de duplicar el dinero y falsificarlo, era algo que no había tenido solución hasta que Satoshi Nakamoto diseño la Blockchain y su protocolo P2P para Bitcoin.

Desde entonces es posible usar criptomonedas con total seguridad sabiendo que el dinero no solo está en un protocolo abierto, transparente y libre, sino que también jamás podrá ser falsificado. Y todo ello porque cada nodo tiene un historial de transacciones de la red, siendo testigo de cada operación en la misma. Esta enorme red de testigos permanece como un registro inmodificable de todo lo que sucede y nos da la seguridad de que el sistema no es manipulable.

Algo realmente útil y que permite transformar a Bitcoin en un dinero digital seguro, el más seguro y transparente de todos.

Wallet

Muchos expertos señalan la elección de la Wallet como un aspecto fundamental del mundo Blockchain. Elegir una Wallet acorde a las necesidades del usuario es casi tan importante como saber elegir en qué momento lanzarse a comprar criptomonedas. (Cobarro, 2019)

"Una Wallet es un programa que almacena criptomonedas".

Una Wallet es un software que almacena pares de claves públicas y privadas (siempre van juntas) y permite enviar y recibir criptomonedas a través de una Blockchain, almacenarlas y controlar el saldo asociado a una cuenta. En verdad lo que hace es firmar las transacciones con la clave privada, donde dichas transacciones implican el envío de un token de una cuenta a otra.

Las claves privadas son como el número PIN de una tarjeta de crédito y sirven para acceder a una cuenta.

Cuando alguien envía por ejemplo un Bitcoin, está enviado un valor en forma de transacción, transfiriendo una propiedad al destinatario.

La propiedad y la conservación de la clave privada otorga un control absoluto sobre los fondos que estén asociados a la clave pública.

Existen varios tipos de Wallets para cubrir distintos tipos de necesidades.

Paper Wallet

Este tipo de Wallet es totalmente inmune a cualquier ataque informático ya que se encuentra permanentemente offline.

Básicamente una Wallet de este tipo es un documento impreso que contiene una dirección pública donde se pueden enviar criptomonedas y una clave privada. Eso te permite operar con ella de manera sencilla.

Muchas veces la mejor manera de operar con este tipo de Wallets es mediante códigos QR para poder escanearlos y realizar una transacción.

Se puede generar una Paper Wallet con servicios como BitAddress que permite crear una dirección aleatoria con su clave privada. Después se debe imprimir el documento generado y configurar las medidas de seguridad necesarias. (Bit Address, 2021)

La ventaja de una Wallet en papel es que las claves no están almacenadas en ninguna red, lo que imposibilita a los atacantes informáticos realizar cualquier robo digital.

Una vez configurada la cartera, la web que genera las llaves funcionará sin conexión así que se debe asegurarse de desconectar de internet el dispositivo antes de obtener las claves. La impresión del documento también es recomendable hacerla en una impresora offline.

Lo más básico y lógico es no olvidar que se está imprimiendo una información privada en un trozo de papel. Por esto se debe tomar las medidas necesarias para guardarlo y protegerlo de cualquier pérdida robo o deterioro.

Wallet Móvil

Este tipo de Wallet es la más recomendada para los usuarios que realizan transacciones con frecuencia.

Estas, se ejecutan igual que cualquier otra aplicación permitiendo pagar directamente con un smartphone. Hay algunas incluso que permiten emplear funciones mediante cercanía, algo parecido al "Contact Less" de las tarjetas de crédito o débito, con el que se puede pagar simplemente tocando el teléfono.

Realmente cualquier software que requiera tener acceso a toda una Blockchain requerirá un espacio de almacenamiento que un smartphone de momento no puede dar. Es por este motivo que hay que simplificar las verificaciones de pago.

Este tipo de Wallet aprovecha la tecnología de verificación de pagos simplificada (SPV) de tal manera que puede trabajar sin problemas con conjuntos muy reducidos de una Blockchain, para conseguir aligerar el peso del almacenamiento.

Toda esa practicidad para almacenar y dar uso a las criptomonedas tiene la contra de que las Wallets Móviles son muy vulnerables a ataques informáticos. Se pierde el control sobre ellas simplemente si alguien obtiene acceso al dispositivo.

Wallet Online
Una Wallet Online o Web almacena las claves privadas en un servidor controlado por la empresa que proporciona los servicios.

Existen muchos servicios que ofrecen características diferentes. Algunos se pueden vincular a monederos móviles o de escritorio.

Permite que los usuarios accedan a sus activos desde cualquier parte en la que el dispositivo tenga acceso a la red.

Es importante elegir bien la Wallet Online ya que las administradoras de cada una pueden tener acceso a las claves privadas y de esa manera potencialmente, acceder a las criptomonedas.

Wallet de escritorio
Las Wallet de escritorio se instalan en una computadora de escritorio o notebook, almacenando las claves privadas en el disco duro. Por esa misma razón son más seguras que las Wallets Online o móviles.

Al estar conectadas a la red cada vez que se enciende el ordenador no son 100% seguras. Estas billeteras son muy recomendables para gente que quiere negociar pequeñas cantidades de criptomonedas desde su computadora.

Hay Wallets de escritorio de muchísimos tipos. Algunas brindan seguridad, otras, anonimato, etc.

Wallet Física / Hardware
Estos dispositivos están diseñados para proporcionar una seguridad adicional a las opciones de almacenamiento en frío (Offline) como las Paper Wallet.

Con una Paper Wallet tus fondos estarán totalmente seguros hasta que utilices un dispositivo para operar con ellos. Si resulta que ese dispositivo está comprometido, la Paper Wallet puede sufrir un ataque.

Las Wallet físicas tienen con ellas un chip de seguridad que impide que las claves privadas se ingresen en una computadora. Para operar con criptomonedas simplemente se deberá ingresar un código PIN en la Wallet.

Además, si por algún motivo la Wallet se perdiera o rompiera se podría restaurar el acceso a la cuenta con un nuevo dispositivo mediante las palabras clave o semilla que se generaron durante la creación de dicha cuenta.

Estas Wallet son la manera más segura de almacenar las criptodivisas, pero tienen un inconveniente para algunos inversores, su precio.

Una debilidad de estas Wallets es la necesidad de almacenar la semilla en un lugar seguro ya que si se pierde esa serie de palabras se perderá también el acceso a la cuenta para siempre.

Token

Un token es un objeto digital que tiene valor en cierto contexto o para determinada comunidad, aunque su propia materialidad no contenga ese valor en sí. (Ripio, 2020)

Por ejemplo, las fichas de casino son solo pedazos de plástico de distintos colores, pero representan cantidades de dinero. Algunas, hasta millones de dólares, aunque fabricar una de ellas cueste apenas centavos.

Eso hacen los tokens, representan otra cosa. Hay muchos motivos para su utilización; la comodidad, la seguridad, la facilidad de transferirlos, etc.

En el mundo cripto, los tokens se generan a partir de piezas de código de programación, en formato de contratos inteligentes que corren sobre una Blockchain. El Smart Contract describe cómo funciona cada token. La base de datos lleva el registro de cuántos tiene cada uno. Y los usuarios pueden enviárselos entre sí como forma de transferirse valor.

Tokens y criptomonedas
En 2009, cuando fue creada por Satoshi Nakamoto, Bitcoin se convirtió en la primera criptomoneda. De inmediato, muchos otros entusiastas de la seguridad digital, las nuevas economías y los desarrollos tecnológicos se lanzaron a crear las suyas propias.

Una de ellas fue Ethereum, creación del joven programador ruso Vitalik Buterin. Fue recién entonces que los tokens se convirtieron en un concepto popular en el mundo cripto.

Actualmente, token y criptomoneda son utilizados como sinónimos. En definitiva, ETH es el token de Ethereum, BTC el de Bitcoin, DAI el de MakerDAO, etc. Y todos estos casos son criptomonedas.

Podemos decir que toda criptomoneda es un token, y que una criptomoneda es un token con valor comercial. Por el contrario, no todos los tokens son criptomonedas.

Tokens, tecnología y economías digitales
En el campo de las economías y tecnologías digitales, los tokens tienen diversos usos más allá de las criptomonedas.

Como monedas: son los tokens usados para transferir valor entre usuarios de una red. Por ejemplo, ETH, el token de Ethereum.

- *Como bienes:* cada vez que se compran assets digitales, en realidad se está comprando un token. Por ejemplo, los Cryptokitties y otros coleccionables.
- *Como acciones:* son los tokens que representan acciones de una empresa tradicional, y que permiten facilitar su trading.
- *Como recompensas:* existen servicios y redes que premian a sus usuarios con tokens. Es el caso de BAT, el token del navegador Brave, browser que bloquea por defecto las publicidades online, pero entrega BATs a quienes eligen verlas voluntariamente.

Por último, repasemos los diferentes tipos de tokens existentes actualmente.

Currency Token
Mientras que Bitcoin (con mayúscula) refiere a la red, bitcoin (en minúsculas) -BTC- es la moneda de esa red. Y esos bitcoins ya no tienen un uso restringido a la red Bitcoin: se pueden usar para cualquier transacción cotidiana, como reemplazo del dinero tradicional. Hoy se pueden comprar tickets de avión, zapatillas, o pagar suscripciones a servicios online. En el mundo cripto, a los currency tokens se los llama directamente criptomonedas y en general el concepto de token se usa mayormente para designar a otros tipos.

Utility Token
Es un tipo especial de token que sirve de ayuda en la capitalización o financiación de proyectos para startups, empresas o grupos de desarrollo de proyectos.
Los Utility Tokens resultan de ayuda cuando una empresa desea crear un cupón que puede canjearse en el futuro por un acceso a sus servicios. (bit2me Academy, 2019)

Security Token
Funcionan como un contrato de inversión, y quienes los compran lo hacen esperando una ganancia (en forma de dividendos de una empresa). Sirven como garantía de propiedad de una porción de la criptomoneda emitida, y ganan o pierden valor acorde a las fluctuaciones de precio de esa moneda. También pueden funcionar como acciones. Son instrumentos diseñados para obtener una ganancia financiera y por eso suelen estar sometidos a regulaciones más estrictas.

Asset Token
Está pensado para representar objetos del mundo real, para facilitar la compra y venta de artículos físicos sin la necesidad de moverlos de un lado al otro. Con un asset token se puede comprar y vender por ejemplo oro, sin necesidad de transportarlo. Otros ejemplos son cabezas de ganado, propiedades, automóviles, juguetes, libros.

Marco de Buena Arquitectura de AWS

La arquitectura es el arte y la ciencia de diseñar y construir grandes estructuras.
Tener aplicaciones con una buena arquitectura aumenta considerablemente la probabilidad de éxito en cualquier proyecto.

El Marco de Buena Arquitectura de AWS es una guía para el diseño de infraestructuras y les otorga las características siguientes:

- Seguridad
- Alto desempeño
- Resiliencia
- Eficacia

El Marco de Buena Arquitectura de AWS puede ser visto como un enfoque para evaluar e implementar arquitecturas en la nube, que brinda prácticas recomendadas que fueron

recopiladas a partir de lecciones aprendidas a través de la revisión de arquitecturas de clientes.

El Marco de Buena Arquitectura de AWS se compone de cinco pilares

1. Excelencia operativa
2. Seguridad
3. Fiabilidad
4. Eficacia del rendimiento
5. Optimización de costos

Cada pilar incluye un conjunto de principios de diseño y una lista de prácticas recomendadas.

En el presente Trabajo voy a poner especial énfasis en la seguridad, pero sin dejar de lado la fiabilidad, eficacia del rendimiento y la excelencia operativa. No será prioritaria en estos momentos la optimización de costos.

Pilar de EXCELENCIA OPERATIVA.

El pilar de Excelencia Operativa se centra en la habilidad de Ejecutar y monitorear las aplicaciones con el objetivo de ofrecer valor de negocio y mejorar de forma continua los procesos y procedimientos

Entre los temas clave se incluyen la administración y automatización de los cambios, la respuesta a eventos y la definición de estándares para administrar correctamente las operaciones diarias.

Principios de Diseño para la Excelencia Operativa:

Realizar operaciones como código: para ello se debe definir toda la carga de trabajo (aplicaciones e infraestructura) como código y actualizarla como código.
Implementar procedimientos de operaciones como código y configurarlos para que se desencadenen automáticamente en respuesta a eventos. Cuando las operaciones se realizan como código, se limita el error humano y se producen respuestas coherentes a los eventos.

Comenzar sobre la documentación: para ellos se debe automatizar la creación de documentación comentada después de cada compilación. La documentación comentada puede ser usada tanto por personas como por aplicaciones. Los comentarios se pueden utilizar como un aporte clarificador para el código desarrollado.

Realizar cambios frecuentes, pequeños y reversibles: diseñar cargas de trabajo para permitir la actualización periódica de los componentes. Realizar grupos pequeños de cambios que puedan revertirse en caso de error (sin que ello afecte el normal funcionamiento de la aplicación, siempre que sea posible).

Refinar los procedimientos de las operaciones con frecuencia: buscar oportunidades para mejorar los procedimientos de las operaciones. Hacer que los procedimientos evolucionen

de forma adecuada a la medida que evolucionan las cargas de trabajo. Establecer días de prueba periódicos para revisar todos los procedimientos, validar se eficacia y asegurarse de que los equipos de trabajo estén familiarizados con ellos.

Anticiparse a errores: identificar las posibles fuentes de error para que se puedan eliminar o mitigar. Probar las situaciones de error y validar su compresión del impacto de estas. Probar los procedimientos de respuesta para asegurarse de que son eficaces y para asegurarse de que los equipos están familiarizados con su ejecución. Establecer días de prueba periódicos para probar las cargas de trabajo y las respuestas del equipo a los eventos simulados.

Aprender de los errores y eventos operativos: impulsar mejoras a partir de las lecciones aprendidas de todos los eventos y errores operativos. Compartir lo aprendido con todos los equipos y en toda la organización.

Pilar de SEGURIDAD.

El Pilar de Seguridad se centra en la capacidad de proteger la información, las aplicaciones y los recursos, a la vez que se aporta valor de negocio a través de evaluaciones de riesgo y estrategias de mitigación.
Entre los temas principales se incluyen la protección, la confidencialidad y la integridad de los datos. Se debe identificar y administrar quien puede hacer determinada actividad.
También configurar controles para detectar eventos de seguridad (administración de privilegios) y proteger aplicaciones y servicios.

Principios de diseño para la Seguridad:

Implementar una base sólida de identidades: implementar el principio de mínimo privilegio y aplicar la separación de obligaciones con la autorización apropiada para cada interacción con los recursos de AWS. Centralizar la administración de privilegios o reducir e incluso eliminar la dependencia de credenciales a largo plazo.

Habilitar la trazabilidad: monitorear, alertar y auditar acciones y cambios en tiempo real. Integrar registros y métricas con sistemas para responder y tomar medidas de manera automática.

Aplicar la seguridad en todas las capas: aplicar defensa en profundidad y controles de seguridad en todas las capas de la arquitectura (por ejemplo, a la red de borde, a la nube virtual privada, a la subred, al balanceador de carga y a cada instancia, sistema operativo y aplicación).

Automatizar las prácticas recomendadas de seguridad: automatizar los mecanismos de seguridad para mejorar la capacidad de escalar de forma segura con mayor rapidez y rentabilidad. Crear arquitecturas seguras e implementar controles definidos y administrados como código en plantillas con control de versiones.

Proteger los datos en tránsito y en reposo: clasificar los datos en niveles de confidencialidad y utilizar mecanismos como cifrado, uso de tokens y control de acceso cuando corresponda.

Mantener a las personas alejadas de los datos: para reducir el riesgo de pérdida o modificación de información de los datos confidenciales debido a errores humanos. Crear mecanismos y herramientas para reducir o eliminar el acceso directo o el procesamiento manual de datos.

Prepararse para eventos de seguridad: Contar con mecanismos de administración de incidentes que se ajusten a los requisitos de la organización. Ejecutar simulaciones de respuesta a incidentes y usar herramientas con automatización a fin de aumentar la velocidad de detección, investigación y recuperación.

Pilar de FIABILIDAD

El Pilar de fiabilidad se centra en la capacidad de un sistema de recuperarse rápidamente después de que se produzcan errores, para satisfacer la demanda del negocio y los clientes. Se centra en la capacidad de una aplicación de recuperarse de interrupciones en la infraestructura y el servicio, incorporar dinámicamente recursos informáticos que satisfagan la demanda y mitigar las interrupciones como errores de configuración o problemas de red temporales.

Principios para el Pilar de Fiabilidad

Probar los procedimientos de recuperación: probar de qué manera fallan los sistemas y validar los procedimientos de recuperación. Utilizar la automatización para simular diferentes errores o para volver a crear situaciones que hayan dado lugar a errores antes. Esta práctica puede exponer rutas de error que usted puede probar y rectificar antes de que se produzca un error real.

Recuperarse automáticamente de los errores: monitorear las aplicaciones en busca de indicadores clave de rendimiento y configurar las aplicaciones para desencadenar una recuperación automatizada cuando se supere un límite. Esta práctica permite la notificación automática y el seguimiento de los errores, así como procesos de recuperación automatizados que solucionan provisoriamente o reparan el error.

Escalar horizontalmente para aumentar la disponibilidad total del sistema: reemplazar un recurso grande por varios recursos más pequeños y distribuir las solicitudes entre estos últimos para reducir el impacto de un único punto de error en la aplicación.

Evitar asumir estimaciones sobre capacidad: monitorear la demanda y el uso de la aplicación y automatizar la incorporación o eliminación de recursos para mantener el nivel óptimo a finde satisfacer la demandada.

Administrar cambios en la automatización: utilizar la automatización para realizar cambios en la infraestructura y administrar cambios en la automatización.

Pilar de EFICACIA DEL RENDIMIENTO

El Pilar de Eficacia del Rendimiento se basa en la capacidad de utilizar la TI y los recursos informáticos de forma eficaz para cumplir los requisitos del sistema y mantener esta eficacia a medida que cambia la demanda y evolucionan las tecnologías.
Entre los temas principales se incluyen la selección de los tipos y los tamaños adecuados de los recursos.
Permite tomar decisiones fundamentadas para mantener la eficacia a medida que evolucionan las necesidades del negocio.

Principios de diseño para la EFICACIA DEL RENDIMIENTO

Democratizar las tecnologías avanzadas: utilizar tecnologías como servicio. Por ejemplo, las bases de datos NoSQL, la transcodificación multimedia y el aprendizaje automático requieren de una experiencia que no está muy extendida en la comunidad técnica. En la nube, estas tecnologías se convierten en servicios que los equipos pueden utilizar. El uso de tecnologías permite que los equipos se centren en el desarrollo de productos en lugar de en el aprovisionamiento y la administración de recursos.

Adquirir escala mundial en cuestión de minutos: implementar aplicaciones en varias regiones de AWS para ofrecer una menor latencia y una mejor experiencia al cliente con un costo mínimo.

Utilizar arquitecturas sin servidor: la arquitectura sin servidor elimina la carga operativa que supone ejecutar y mantener servidores para llevar a cabo actividades informáticas tradicionales. Las arquitecturas sin servidor también pueden reducir los costos transaccionales, ya que los servicios administrados operan a escala de la nube.

Experimentar con más frecuencia: realizar pruebas comparativas de diferentes tipos de instancias, almacenamiento o configuraciones.

Disponer de compatibilidad mecánica: utilizar el enfoque tecnológico que se ajuste mejor a lo que se intenta conseguir. Por ejemplo, tener en cuenta los patrones de acceso a datos cuando se seleccionan enfoques para bases de datos o almacenamiento.

El pilar de optimización de costos habla de ejecutar aplicaciones para aportar valor de negocio al menor precio. El mismo queda fuera del alcance de este trabajo.

3. DESCRIPCION DE LA SOLUCION

En el siguiente apartado se detallarán los pasos llevados a cabo para cumplir con los objetivos del presente Trabajo.

3.1 Solución propuesta (Hipótesis)

Utilizando la infraestructura de Blockchain Federal Argentina (BFA), se implementó un nodo sellador perteneciente a un organismo público.

El proceso abarcó desde la realización de los pasos procedimentales y administrativos para obtener los permisos necesarios para montar dicho nodo en la red de producción de BFA, hasta la instalación del hardware y software que soporta dicho nodo.

Todo esto se llevó a cabo siguiendo el Marco de Buena Arquitectura recomendado por Amazon Web Services (AWS).

También se desarrolló una Aplicación distribuida (dApp) que se conecta a un Smart Contract dentro de Blockchain Federal Argentina llamado “sello de tiempo”. Con esto se logró garantizar la integridad distintos documentos.

Se pretende que este Trabajo Final pueda ser tomado como una guía detallada que explique cómo montar un Nodo complejo en una Blockchain; desde los pasos administrativos y procedimentales hasta la instalación, montaje y programación del nodo y seguido a esto, el desarrollo de una dApp que muestra cómo se comunica con la red.

3.2 Objetivos

Objetivo principal: implementación de un Nodo Sellador dentro de Blockchain Federal Argentina (BFA).

Objetivo secundario: creación de una dApp, que se conecta a un Smart Contract llamado “Sello de Tiempo”.

El objetivo principal incluye el detalle de los pasos procedimentales y administrativos para la obtención de los permisos necesarios por parte de Blockchain Federal Argentina para montar un nodo Sellador sobre su red, perteneciente a una institución pública. También incluye una explicación sobre cómo se debe realizar la instalación del hardware y software que soporta dicho nodo.

El objetivo secundario incluye el desarrollo de una dApp, que se conecta a un Smart Contract llamado “Sello de Tiempo”. El mencionado contrato está presente dentro de la BFA

3.3 Metodología y Técnicas utilizadas

Para el presente trabajo se utilizó un enfoque explicativo, siguiendo un claro diseño experimental.

Lo primero que se hizo desde la Universidad Nacional de La Matanza, fue establecer contacto con Blockchain Federal Argentina y solicitar ser parte del consorcio representando a una institución pública.

Una vez aceptada la incorporación de la Universidad Nacional de La Matanza por parte de autoridades de Blockchain Federal Argentina, se llevó a cabo la instalación del nodo sellador.

Todo esto se llevó a cabo siguiendo el **Marco de Buena Arquitectura** (WAF - Well Architectured Framework) recomendado por Amazon Web Services (AWS).
El WAF de AWS fue explicado en detalle en la sección anterior.

Con respecto al objetivo secundario del Trabajo Final se desarrolló una Aplicación distribuida (dApp). La misma se conectó a un Smart Contract dentro de Blockchain Federal Argentina llamado “sello de tiempo”. Con esto se logró garantizar la integridad distintos documentos.

Para el desarrollo del software se utilizó una un enfoque iterativo-incremental.

3.4 Desarrollo de la solución

Celebración del Contrato
Lo primero que se hizo fue establecer contacto con Blockchain Federal Argentina y solicitar ser parte del consorcio representando a una institución pública.

Una vez establecido un canal de comunicación con BFA se acordó la entrega de documentación respaldatoria de quienes representan a la institución y la entrega del Formulario de Solicitud de Incorporación.

También se indicó que la institución solicita la Incorporación al CONTRATO DE COLABORACIÓN PÚBLICO – PRIVADA BLOCKCHAIN FEDERAL en calidad de parte.

Ver Anexo Contrato de Colaboración Público-Privada.

Luego de un tiempo, el comité directivo de BFA se expidió haciendo un acuse de notificación indicando que acepta la incorporación de la institución como parte de BFA.

Una vez aceptada la incorporación puso en contacto a su sector de IT con un representante de la institución.

Luego se procedió a instalar el nodo sellador.

Instalación del Nodo

Para llevar a cabo la instalación del nodo sellador primeramente se instaló y configuró una máquina virtual perteneciente a la UNLaM. El mencionado servidor está dentro de la nube proporcionada por AWS.

A continuación, se darán las indicaciones en formato TO-DO, las mismas fueron probadas exitosamente en la realización del presente trabajo. Esto es con el objetivo de que el mismo sirva como guía ante implementaciones similares.

1. Registrarse en AWS. (Ver anexo registrarse en AWS)
2. Configurar MFA para la cuenta. Este punto es opcional pero recomendado por AWS. (Ver anexo registrarse en AWS)
3. iniciar sesión en AWS.
4. instalar una máquina virtual donde luego se configurará el nodo (Ver anexo Instalación de una Máquina Virtual en AWS).
5. conectarse a la máquina virtual a través de SSH. (Ver anexo Conectarse a una Máquina Virtual en AWS).
6. instalar el Nodo Sellador (Ver anexo Instalación de un Nodo Sellador en BFA).
(GitLab Nodo Docker, 2018)

Una vez instalado y probado el nodo sellador, se procedió con la segunda parte de este trabajo que es el desarrollo de una aplicación web que se conecta con un Smart Contract llamado “Sello de Tiempo” que reside en la BFA (Renzo Ontivero, 2019). Esta aplicación permite enviar hashes a la BFA a través de este contrato de modo tal que el resultado es garantizar la integridad de distintos documentos.

Desarrollo de la dApp

Con respecto al objetivo secundario del Trabajo Final se desarrolló una Aplicación distribuida (dApp). La misma se conectó a un Smart Contract dentro de Blockchain Federal Argentina llamado “sello de tiempo”. Con esto se logró garantizar la integridad distintos documentos.

La aplicación fue desarrollada siguiendo un enfoque iterativo-incremental.
Se utilizó PHP Laravel Framework para el desarrollo del backend.
Para la comunicación con el Smart Contract “Sello de Tiempo” se utilizó la biblioteca curl.
He liberado el código bajo licencia GPL v3.
(UNLaM-BFA, 2021)

La aplicación permite que un usuario asignado a personal de la UNLaM tome un documento emitido por la institución y lo procese mediante la acción SELLAR.
Lo que la aplicación hace es tomar el documento y en base a su contenido, generar un hash utilizando el algoritmo de cifrado SHA256 y posteriormente entregar el mismo a un nodo transaccional público. Éste último, toma ese hash y en comunicación con el Smart Contract Sello de Tiempo, arma una transacción. Una vez armada la transacción, la envía a la red

para que mediante el algoritmo de sellado PoA, ésta sea tomada por un Nodo Sellador y sea agregada a un bloque y posteriormente sellada y agregada a la Blockchain.

Las páginas que forman la aplicación son las siguientes:

Inicio
Esta página brinda una breve explicación de la app y posee un enlace a la página que permite verificar si un documento fue sellado con anterioridad en BFA. En el caso de que el usuario logueado sea un administrador también mostrará un enlace a la página que permite sellar un documento.
Ver Ilustración 37. Pantalla dApp. Inicio

Sellar
Esta página solo se muestra en el caso de que el usuario logueado sea un administrador. Permite sellar un nuevo documento, subiendo su hash a la Blockchain BFA.
Ver Ilustración 38. Pantalla dApp. Sellar.

Verificar
Esta página se muestra incluso si no existe un usuario logueado, por lo cual es visible para cualquier persona que visite el sitio. Permite verificar si un documento fue sellado con anterioridad, es decir, calcula el hash de un documento y verifica si éste está presente en la Blockchain BFA.
Ver Ilustración 39. Pantalla dApp. Verificar.

Quienes somos
Esta página brinda una breve explicación de quien es el grupo que soporta la app y posee un enlace a la página que permite verificar si un documento fue sellado con anterioridad en BFA. En el caso de que el usuario logueado sea un administrador también mostrará un enlace a la página que permite sellar un documento.
Ver Ilustración 40. Pantalla dApp. Quienes Somos.

Contacto
Esta página se muestra incluso si no existe un usuario logueado, por lo cual es visible para cualquier persona que visite el sitio. Permite enviar un mensaje a un equipo de personas de la UNLaM que se encargan de dar soporte a la app.
Ver Ilustración 41. Pantalla dApp. Contacto.

Login
Esta pantalla permite que una persona con los permisos adecuados ingrese a la app con un usuario.
Ver Ilustración 42. Pantalla dApp. Login.

4. FUTUROS TRABAJOS Y LO QUE QUEDA POR INVESTIGAR

Como trabajo a futuro sería muy bueno poder implementar dentro de la red un Nodo Transaccional propio y dejar de utilizar un nodo público. Esto traerá como ventaja, poder administrar dicho nodo transaccional. Actualmente ante una caída del nodo público, se debería esperar que el área de IT (que no corresponde a la UNLaM) lo levante nuevamente. Esto no sucede a menudo, pero en el caso de suceder, se pierde el control de la situación.

También sería interesante escribir un nuevo contrato propio, que permita generar una trazabilidad de un documento sellado. Recordemos que el Smart Contract Sello de Tiempo, permite que se cree una transacción con un hash correspondiente a un archivo dado y en el caso de que este documento se modifique emitirá un hash totalmente diferente. Para el caso de uso propuesto (sellar un documento emitido por una institución) esto es correcto y suficiente. Pero este contrato no permite establecer una relación entre modificaciones sucesivas de un mismo documento, sino que para él (Sello del Tiempo), se trata de "otro documento".

Con respecto a la dApp desarrollada debería crearse un Backend dedicado a la administración de la aplicación, con la posibilidad de crear diferentes usuarios con distintos tipos de roles. En principio crearía el rol de administrador, que sería un usuario con los permisos necesarios para crear nuevos usuarios y administrarlos. También crearía el rol de sellador que sería asignado a los usuarios que tengan permisos para sellar los documentos emitidos por la institución. Y por último crearía el rol visualizador que pudiera ver en detalle todos los documentos sellados a lo largo del tiempo, con el correspondiente usuario que lo selló, fecha y hora, etc.

Por último, debería crearse un módulo con distintos tipos de estadísticas que permitan realimentar a la aplicación y mejorarla.

5. CONCLUSIONES

Como conclusiones puedo decir que la concreción del primer objetivo de este trabajo, Implementar un Nodo Sellador en una red Ethereum sirvió para entender en profundidad cómo funciona una Blockchain, con el agregado fundamental de permitir a una institución formar parte de esta, con todo el valor que eso trae consigo.

En la segunda parte, creación de una dApp, se demostró de forma práctica, como a través del uso de una Blockchain se puede enviar un mensaje a la misma (en este caso el hash de un documento), y este mensaje es inmutable e incuestionable. Lo que da como resultado la autenticidad de un mensaje de forma indiscutible.

Desde el punto de vista de la Ciberseguridad uno de los temas más discutidos es justamente la integridad de los mensajes enviados entre diferentes autores.

Blockchain vino a cambiar este escenario, y es una tecnología que trae consigo muchos otros conceptos y tecnologías asociadas y posibilita un sinfín de aplicaciones.

Para este caso, nos permite subir públicamente un hash que representa un documento y solo ese documento. Si alguien cambiara al menos una coma de ese documento, ya no sería el mismo, por lo cual, al compararlo con el hash subido del documento original, el resultado sería que son documentos diferentes.

Esto nos permite emitir documentación desde una Entidad, en este caso una Universidad Nacional, sellarla y enviarla a quien corresponda. Si el receptor o incluso cualquier otra persona quisiera verificar la integridad de la documentación (esto es, que sea ORIGINAL) no tiene más que consultar a la Blockchain.

Cabe destacar que ese hash que se genera al procesar el documento se sube a la Blockchain de Ethereum y está disponible para todo el mundo sin restricciones.

“La mejor forma de ocultar algo es ponerlo a la vista de todos”.

Esto trae consigo la ventaja de que no se necesitan subir los documentos originales a la Blockchain sino un hash de estos.

Mas allá de la ventaja de no tener que colocar grandes conjuntos de datos en la nube, sino solo un hash, existe otra ventaja aun mayor y es que en el caso de tener información sensible, no es necesario subirla, sino que la misma puede ser resguardada y solo se necesita del hash para verificar su integridad.

6. BIBLIOGRAFÍA

Andreas M. Antonopoulos, Gavin Wood Ph. D. (2018) Mastering Ethereum: Building Smart Contracts and DApps 1st Edition. Editorial O'Reilly.

Andreas M. Antonopoulos. (2017) Mastering Bitcoin: Programming the Open Blockchain 2nd Edition. Editorial O'Reilly.

EscuelaCryptoEs (2019). Participar en cripto y vivir para contarlo. Google Play Books.

EscuelaCryptoEs (2020). Programando Ethereum. Google Play Books.

Mark Smith. (10 Julio 2019). Blockchain: Conociendo la Revolución del Blockchain y la Tecnología detrás de su Estructura. Editor Guy Saloniki, 2019

Mayukh Mukhopadhyay (23 febrero 2018). Ethereum Smart Contract Development: Build Blockchain-based Decentralized Applications Using Solidity. Packt Editorial.

Ritesh Modi (2018). Solidity Program Essentials. Packt Editorial.

Matt Stauffer (2019). Laravel Up & Running. 2nd Edition. Editorial O'Reilly.

7. REFERENCIAS

Bit Address. (2021). *BitAddress*. Obtenido de https://www.bitaddress.org/

bit2me Academy. (1 de 2019). *bit2me Academy*. Obtenido de https://academy.bit2me.com/que-es-utility-token/

Blockchain Federal Argentina. (2017). *BFA*. Obtenido de https://www.bfa.org

Cobarro, Á. (2019). *bitcobie*. Obtenido de https://www.bitcobie.com/wallet/

Ethereum. (2016). *Ethereum.org*. Obtenido de https://www.ethereum.org

García, R. J. (2018). *rubenjgarcia*. Obtenido de https://rubenjgarcia.cloud/crear-cuenta-aws-gratis/

GitLab Nodo Docker. (2018). Obtenido de https://gitlab.bfa.ar/greykoda/nodo-api-for-docker

Renzo Ontivero. (2019). *Sello de Tiempo 2.0*. Obtenido de https://gitlab.bfa.ar/blockchain/tsa2

Ripio. (2 de 6 de 2020). *Ripio*. Obtenido de https://launchpad.ripio.com/blog/que-es-un-token-y-como-funciona

Solé, R. (20 de Junio de 2021). *Profesional Review*. Obtenido de https://www.profesionalreview.com/2021/06/20/que-son-smart-contract/

UNLaM-BFA. (2021). *GitHub*. Obtenido de https://github.com/jonuran/aws_unlam

8. ANEXOS

Contrato de Colaboración Público-Privado

FORMULARIO DE SOLICITUD DE INCORPORACIÓN

En la Ciudad Autónoma de Buenos Aires, a los ____ días del mes de _____ de ____, _____, identificada con CUIT N°_____, representada en éste acto por _____, con DNI N°_____, en su carácter de _____, con domicilio legal en la calle _____, _____, _____, solicita su incorporación al CONTRATO DE COLABORACIÓN PÚBLICO – PRIVADA BLOCKCHAIN FEDERAL, suscripto con fecha 18 de junio de 2018 y sus Addendas suscriptas el 28 de junio de 2018, el 21 de noviembre de 2018 y el 11 de abril de 2019, en adelante el CONTRATO, en calidad de PARTE, en el sector _____, en los términos de su Cláusula NOVENA.
Se designa como representante ante la BFA a _____, y como suplente a _____.
Por el presente, el suscripto declara conocer y aceptar todos y cada uno los términos del CONTRATO, aceptando expresamente sus objetivos, derechos y obligaciones establecidos en el mismo y en el REGLAMENTO INTERNO y en las POLÍTICAS DE USO DE LA BFA.

_____

SOLICITANTE

- Este formulario debe estar acompañado de: (1) el Estatuto, Contrato Social u otra forma de documento del cual surja la existencia de la Entidad solicitante, (2) el acto de designación o personería de quien firma (ambos documentos se pueden enviar escaneados).
- El formulario debe ser firmado (hológrafa o digital si fuera posible) y enviado escaneado a CdA@bfa.ar.

PARA USO DEL CONSEJO DE ADMINISTRACIÓN ↓

En la Ciudad Autónoma de Buenos Aires, a los ____ días del mes de _____ de _____, el CONSEJO DE ADMINISTRACIÓN del CONTRATO DE COLABORACIÓN PÚBLICO – PRIVADA BLOCKCHAIN FEDERAL, representado por los señores _____, Acepta ( ) / No acepta ( ) en este acto la incorporación como PARTE, en el sector _____, de _____ identificada con CUIT N°_____.

_____ _____

CONSEJO DE ADMINISTRACIÓN

PARA SER DILIGENCIADO POR LA PARTE UNA VEZ ACEPTADA SU INCORPORACIÓN POR EL CDA ↓

ACUSE DE NOTIFICACIÓN

En la Ciudad Autónoma de Buenos Aires, a los ____ días del mes de _____ de _____, _____, identificada con CUIT N°_____, representada en éste acto por _____, con DNI N°_____, en su carácter de _____ , con domicilio legal en la calle _____, _____, _____, se notifica de la resolución del CdA-BFA por la que se Acepta ( ) / No acepta ( ) su incorporación al CONTRATO DE COLABORACIÓN PÚBLICO – PRIVADA BLOCKCHAIN FEDERAL, suscripto con fecha 18 de junio de 2018 y sus Addendas suscriptas el 28 de junio de 2018, el 21 de noviembre de 2018 y el 11 de abril de 2019, en adelante el CONTRATO, en calidad de PARTE, en el sector _____, en los términos de su Cláusula NOVENA.

_____

_____

PARTE

Registrarse en AWS.

Abrir un navegador como Mozilla Firefox, Google Chrome, etc. desde allí crearemos nuestra nueva cuenta en AWS. (García, 2018)

Ir a la página https://aws.amazon.com y hacer clic en “Cree una cuenta de AWS”.

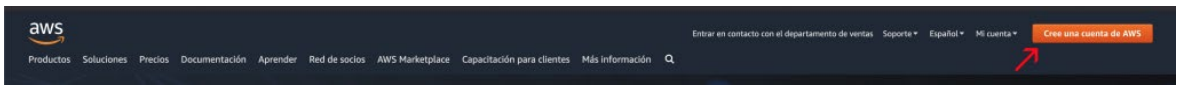

Ilustración 3. Creación de Cuenta en AWS.

En la siguiente pantalla colocar el email, la contraseña (se debe inventar una) y confirmarla, luego se debe elegir un nombre para la cuenta de AWS.

Ilustración 4. Registrarse en AWS. Paso 1.

Hacer clic en Continuar y luego elegir si la cuenta será utilizada para negocios o de forma personal (en este trabajo se eligió negocios).

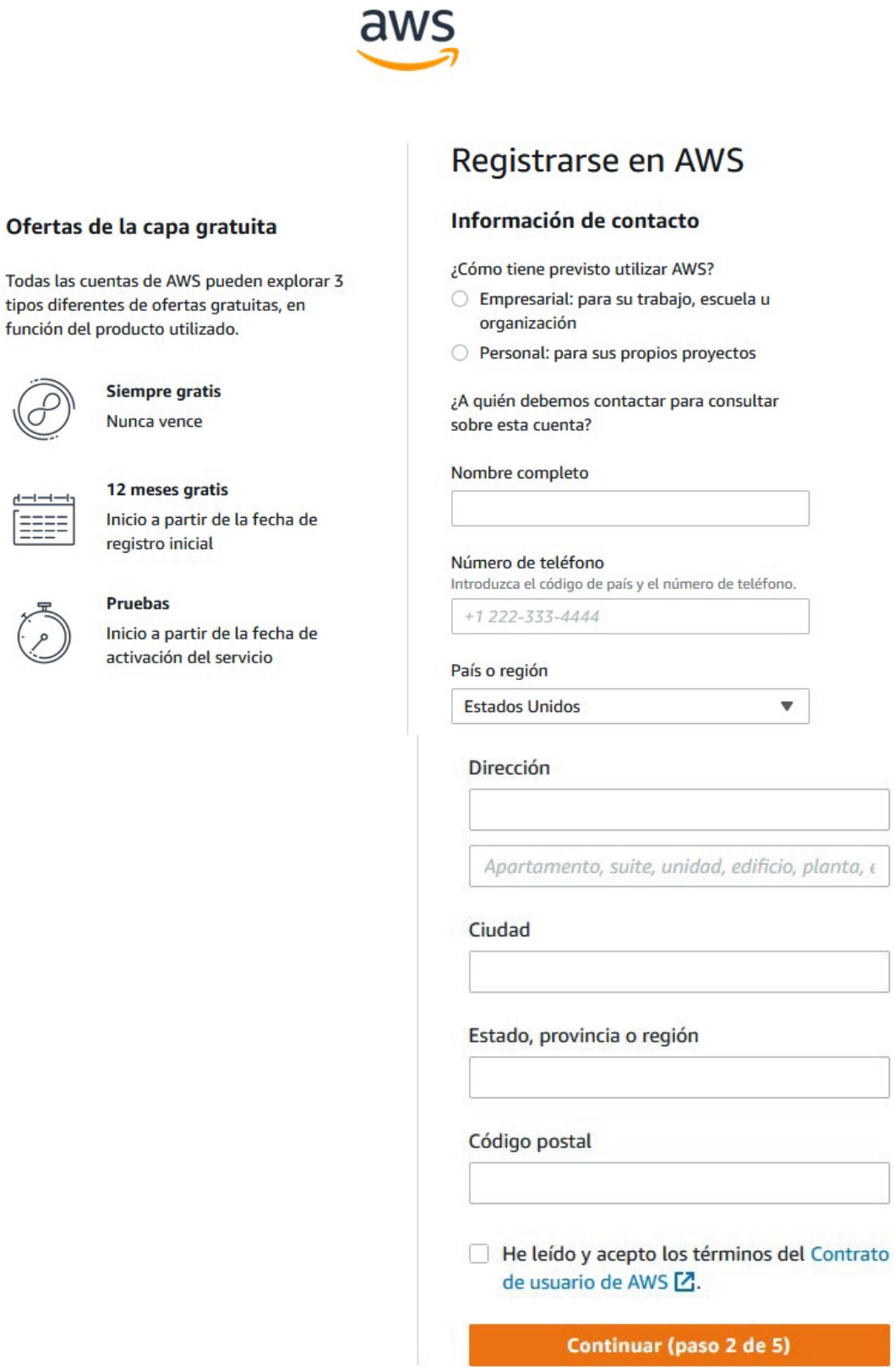

Ilustración 5. Registrarse en AWS. Paso 2.

Además, se debe completar Nombre completo, número de teléfono, se debe elegir país o región, dirección, ciudad, Estado, provincia o región, código postal y aceptar los términos del contrato de Usuario de AWS. Luego hacer clic en continuar.

A continuación, se debe ingresar el número de tarjeta de crédito, y se cobrará un cargo de un dólar que será retenido en la cuenta unos 3 / 5 días, pero luego será devuelto. Esto es sólo para comprobar que esa tarjeta de crédito es real y tiene fondos.

Ilustración 6. Registrarse en AWS. Paso 3.

Una vez verificada la tarjeta de crédito se debe verificar un número de teléfono. Para eso se debe introducir el número y se recibirá un SMS o llamada de verificación.

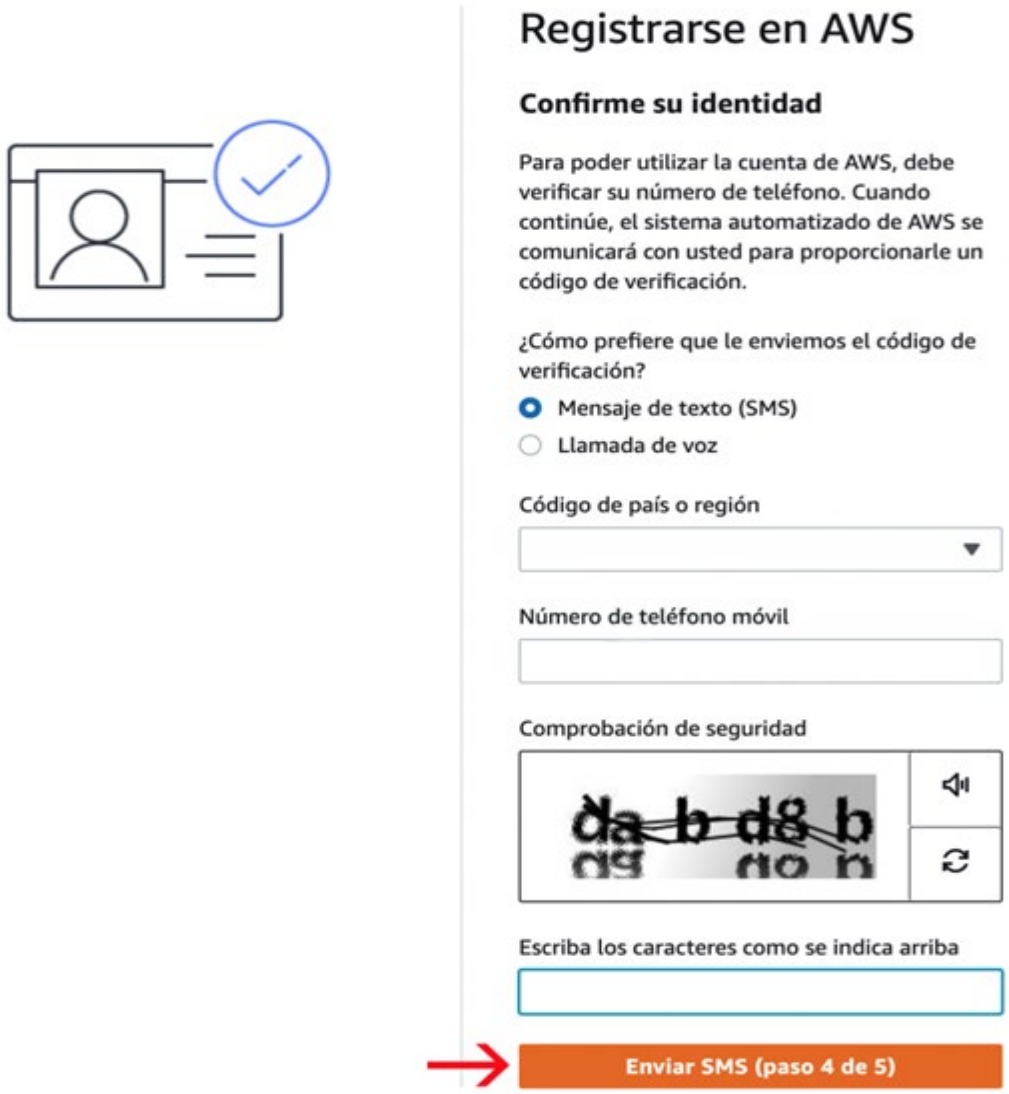

Ilustración 7. Registrarse en AWS. Paso 4.

Luego de recibido el código debe ingresarlo.

Ilustración 8. Registrarse en AWS. Paso 5.

Luego se debe elegir qué tipo de soporte se desea. Aquí hay tres tipos:

Basic: es gratuito.
Developer: desde 29$. El precio va en función de lo que se facture en AWS. Cuanto más se factura más se nos va a cobrar.
Business Support: Empieza en 100$ dólares y también va en función de la facturación en AWS.

Nosotros usamos el Basic.

Ilustración 9. Registrarse en AWS. Pantalla final.

Hacer clic en Finalizar Registro y la cuenta se creará.

Felicitaciones

Gracias por registrarse en AWS.

Estamos en proceso de activar su cuenta. Esto solo debe tardar unos pocos minutos. Recibirá un correo electrónico cuando se haya completado el proceso.

Ir a la consola de administración de AWS

Regístrese para obtener otra cuenta or entre en contacto con el departamento de ventas.

Ilustración 10. Pantalla felicitaciones.

Iniciar sesión en AWS

Para acceder a la consola de AWS, primero se debe iniciar sesión. Existen dos opciones: entrando como root user o entrando como IAM user.
La primera vez se debe ingresar como root user y posteriormente debe guardarse este usuario y crear nuevos usuarios de IAM. Solo debe usarse el usuario root para casos especiales y para todo lo relativo a la facturación de los servicios de AWS.

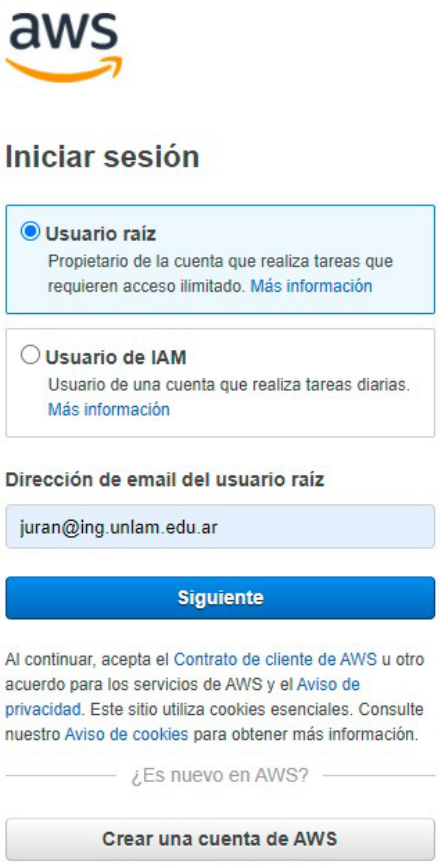

Ilustración 11. Iniciar Sesión.

Se debe ingresar los caracteres de la siguiente pantalla como comprobación de seguridad.

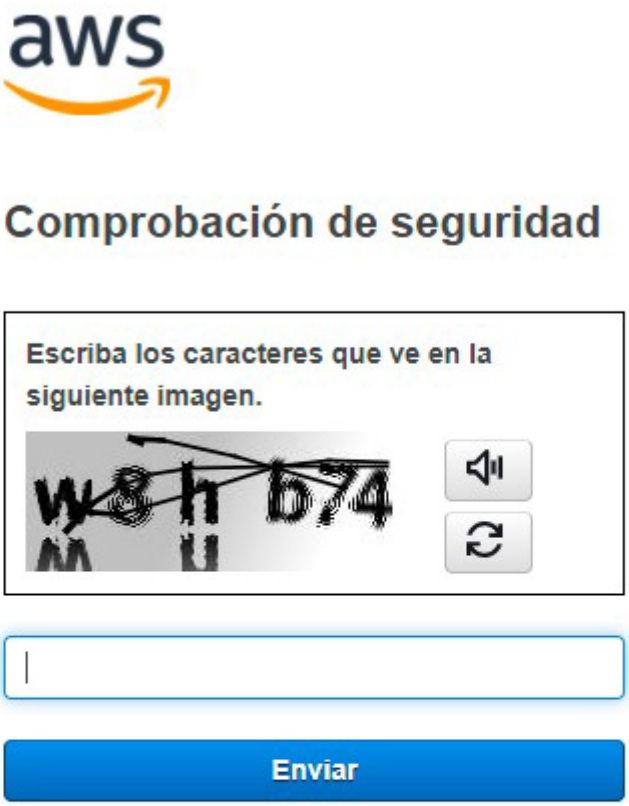

Ilustración 12. Iniciar Sesión. Comprobación de seguridad.

Luego se debe hacer clic en enviar

Ilustración 13. Login.

Por último, debe colocarse la contraseña y hacer clic en iniciar sesión.

Activar MFA en AWS

El MFA es una protección extra a la hora de entrar a la cuenta y es sumamente recomendable tenerlo activado, Para ello una vez dentro de la consola buscar el servicio IAM.

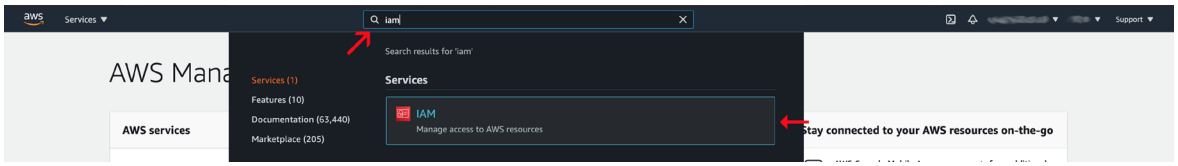

Ilustración 14. Activar MFA en AWS. Buscar IAM.

Luego si el usuario no tiene el MFA activado, hacemos clic en Habilitar MFA.

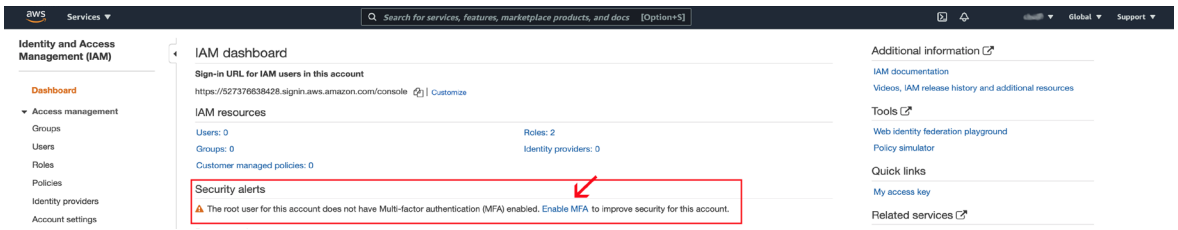

Ilustración 15. Activar MFA en AWS. Verificar MFA.

A continuación, hacer clic en Activar MFA.

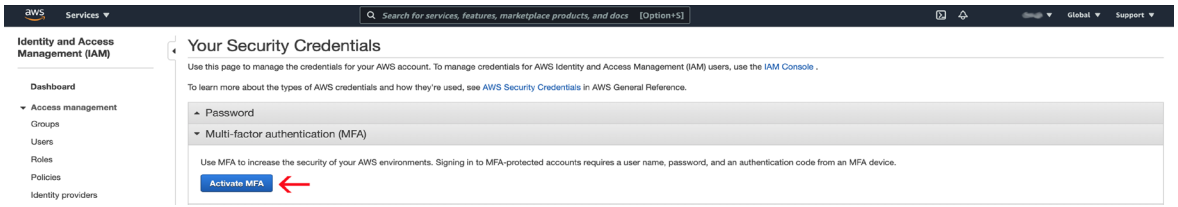

Ilustración 16. Activar MFA en AWS. Activar.

Utilizaremos un teléfono móvil para esto, por lo cual, seleccionamos la primera opción Virtual MFA device.

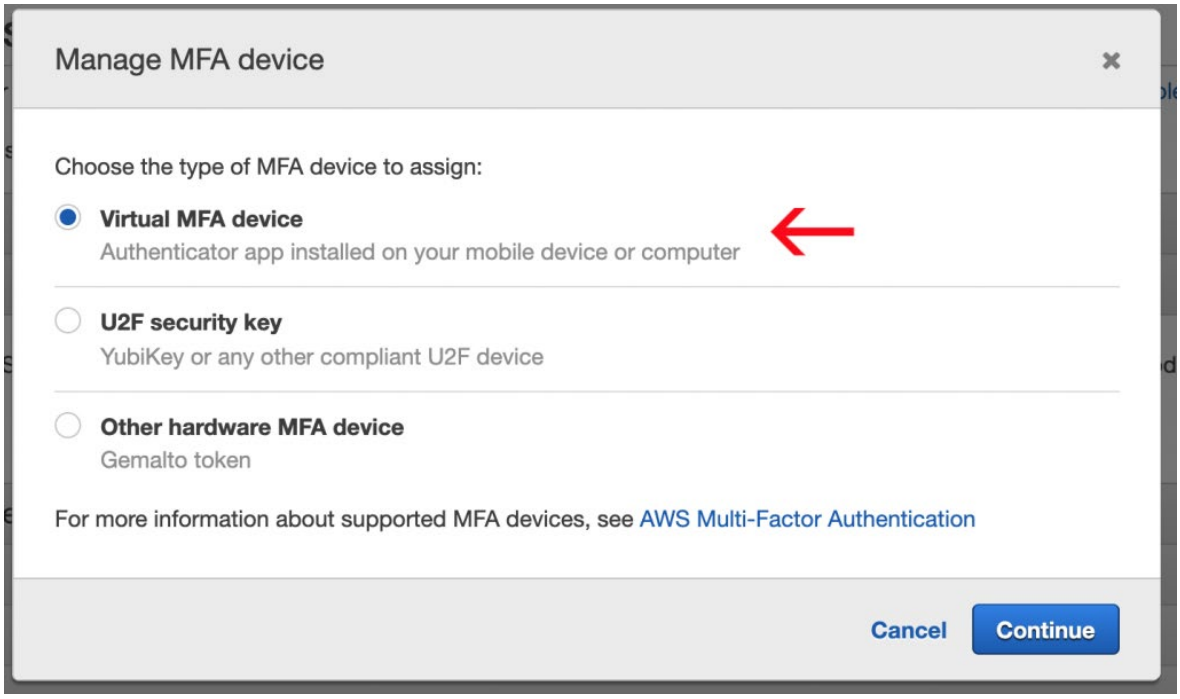

Ilustración 17. Activar MFA en AWS. Virtual device.

Para el siguiente paso debemos instalar una aplicación compatible con nuestro teléfono. Recomiendo usar Google Authenticator (Android / iOS), que es la utilizada para el presente trabajo.

Una vez instalada hacemos clic en Mostrar código QR.

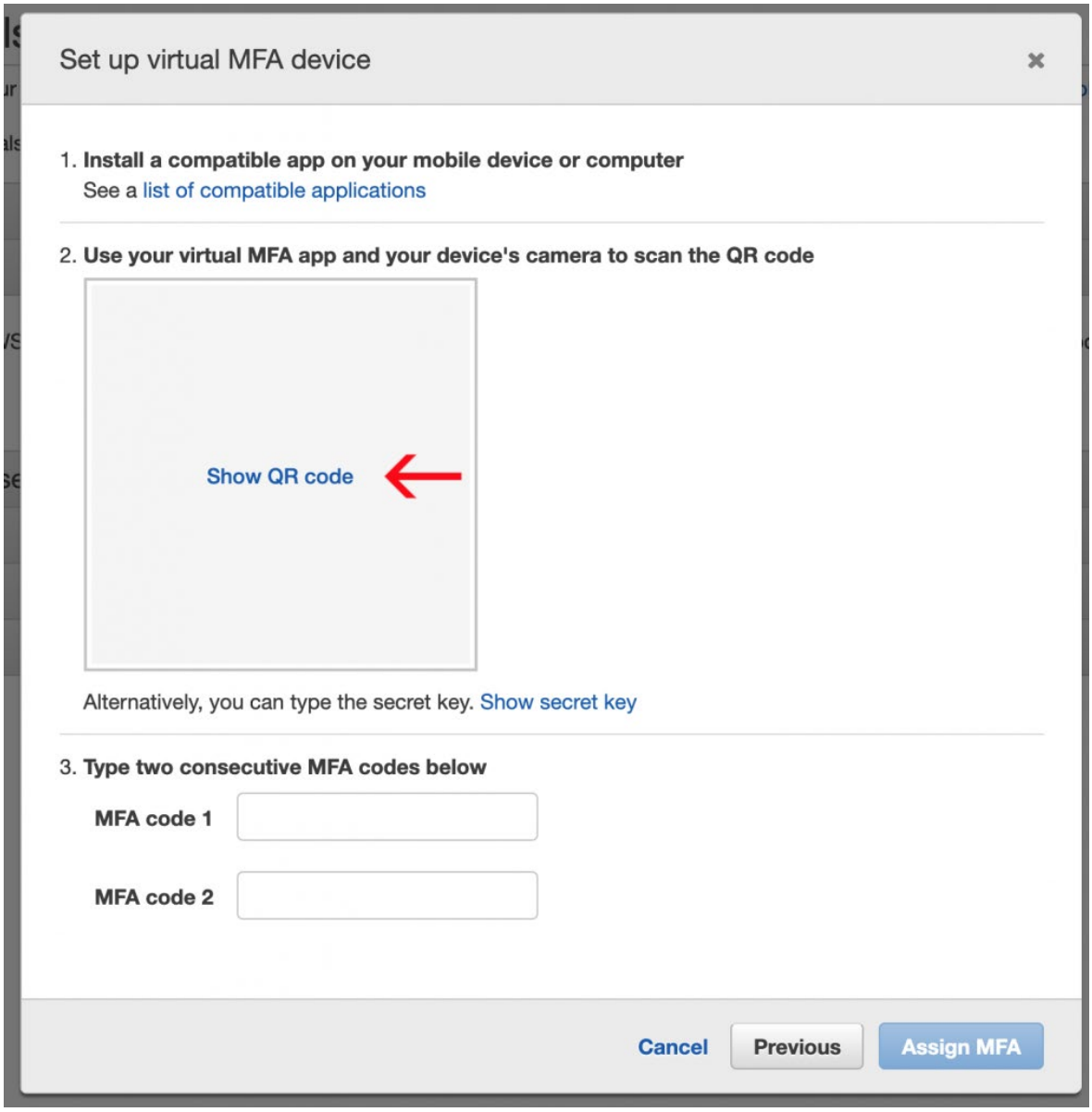

Ilustración 18. Activar MFA en AWS. QR code.

Esto mostrará un QR que se debe leer con la aplicación instalada anteriormente y después se debe escribir dos códigos MFA correctos.

Ilustración 19. Activar MFA en AWS. Validar código.

Una vez configurado la próxima vez que se ingrese se nos pedirá introducir el código para el ingreso a la cuenta.

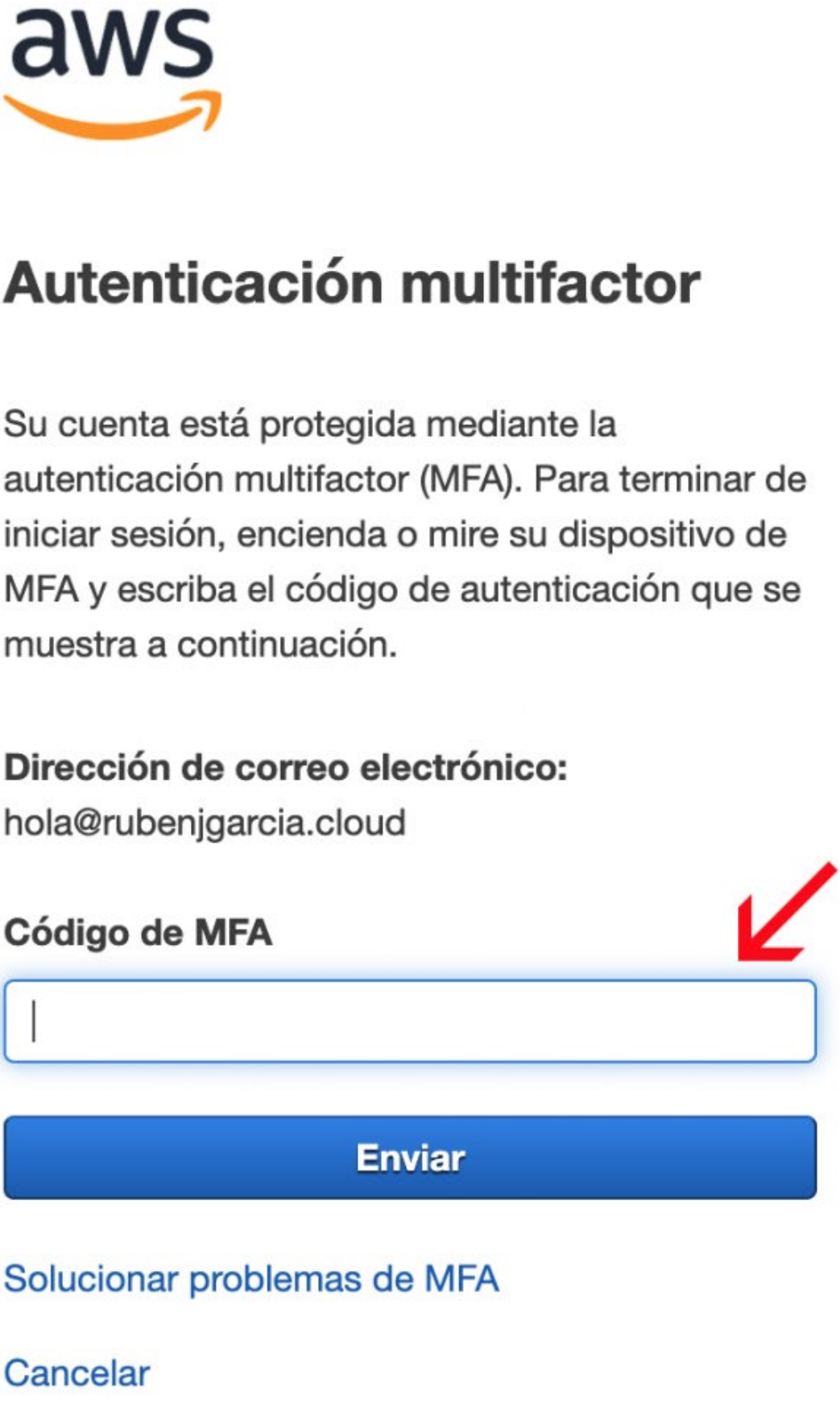

Ilustración 20. Activar MFA en AWS. Introducir MFA.

Y con esto se termina de configurar el MFA para esta cuenta.
Es recomendación de AWS utilizarlo, ya que forma parte de las Buenas Prácticas dentro de su WAF.

Instalación de una Máquina Virtual en AWS

Estando en la consola de AWS buscar y seleccionar el servicio EC2.

Ilustración 21. Instalación de una VM en AWS. Consola.

En este momento se visualizará todas las máquinas virtuales que posee nuestra cuenta con sus correspondientes estados. Como se quiere lanzar una nueva, hacer clic en Lanzar instancias.

Ilustración 22. Instalación de una VM en AWS. Lanzamiento.

A continuación, se verá una lista de los posibles sistemas operativos que pueden elegirse.

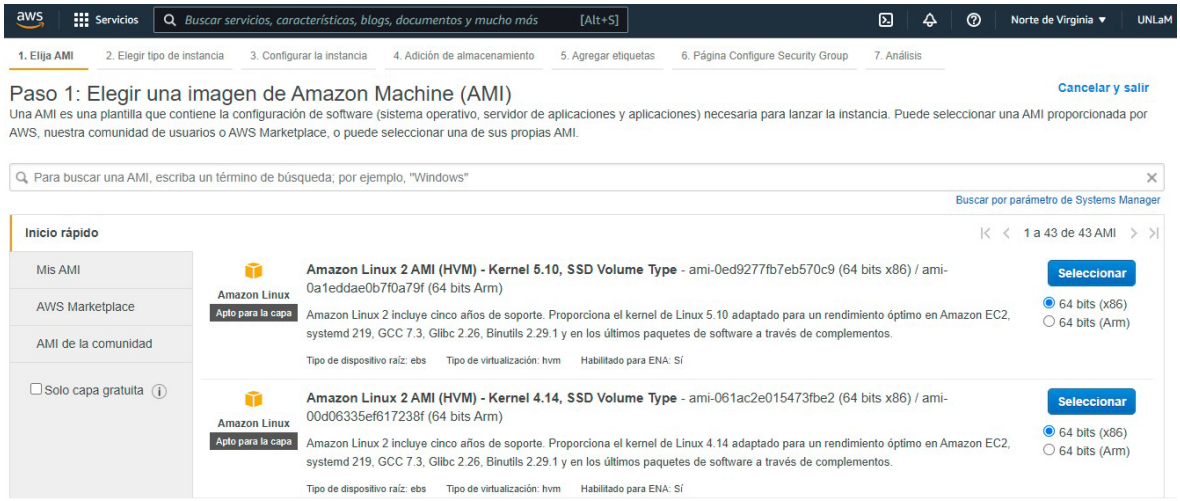

Ilustración 23. Instalación de una VM en AWS. Lanzamiento Paso 1.

Seleccionamos el sistema operativo que necesitamos. En este trabajo utilizamos Ubuntu Server 20.04.

Ilustración 24. Instalación de una VM en AWS. Lanzamiento. Elección SO.

A continuación, se debe seleccionar un tipo de instancia. Esto es la CPU, memoria, disco, etc.

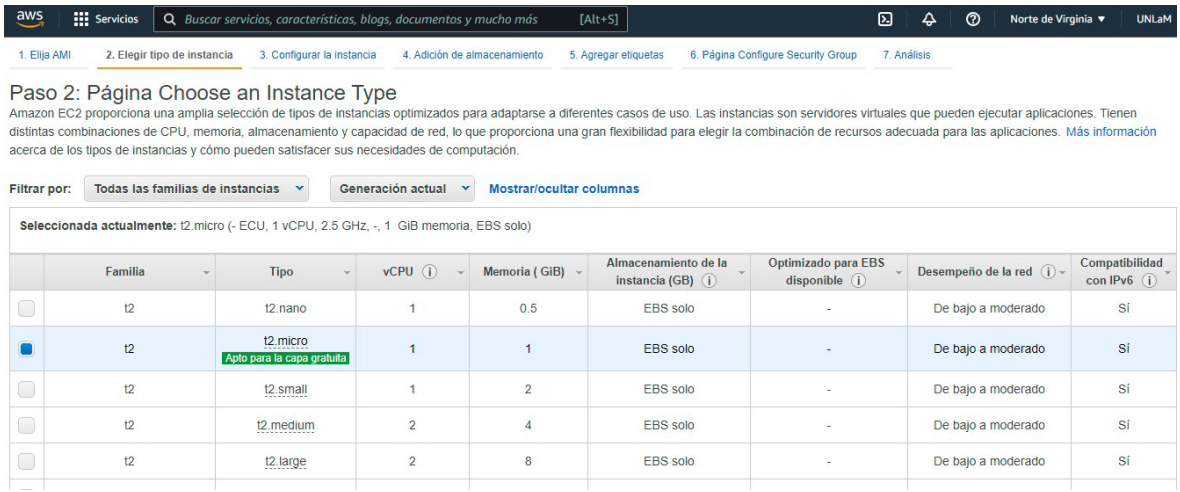

Ilustración 25. Instalación de una VM en AWS. Lanzamiento Paso 2.

A continuación, se debe configurar lo relativo a la red donde estará la instancia. Puede dejarse los valores por defecto y configurarla luego.

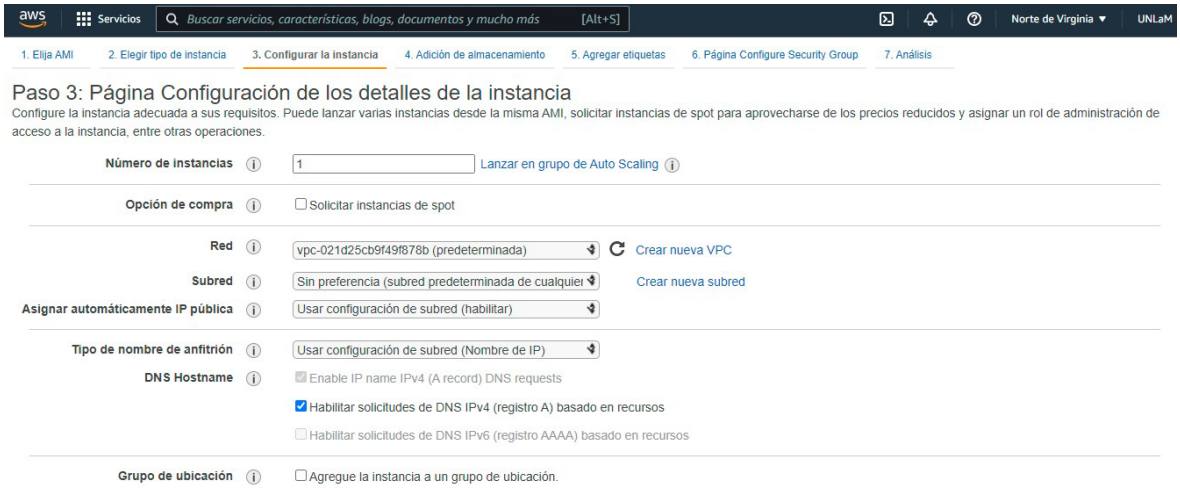

Ilustración 26. Instalación de una VM en AWS. Lanzamiento Paso 3.

Luego debe configurarse el disco. Para este trabajo se utilizó un disco SSD de 30 GB, lo cual nos mantiene en la capa gratuita de AWS.

Ilustración 27. Instalación de una VM en AWS. Lanzamiento Paso 4.

Seguido a esto pueden agregarse algunas etiquetas a la instancia, del tipo clave-valor. En este trabajo no se han utilizado etiquetas.

Ilustración 28. Instalación de una VM en AWS. Lanzamiento Paso 5.

Luego se debe configurar el Security Group, esto es, distintas opciones de Firewall. Debemos abrir el puerto para la conexión SSH, para poder conectarnos a la Máquina Virtual

e interactuar con ella a través de la consola. También se debe abrir los puertos para las conexiones HTTP y HTTPS

Ilustración 29. Instalación de una VM en AWS. Lanzamiento Paso 6.

Por último, se verá un resumen de todas las opciones elegidas, las cuales se deben verificar y posteriormente si todo está como se desea, continuar.

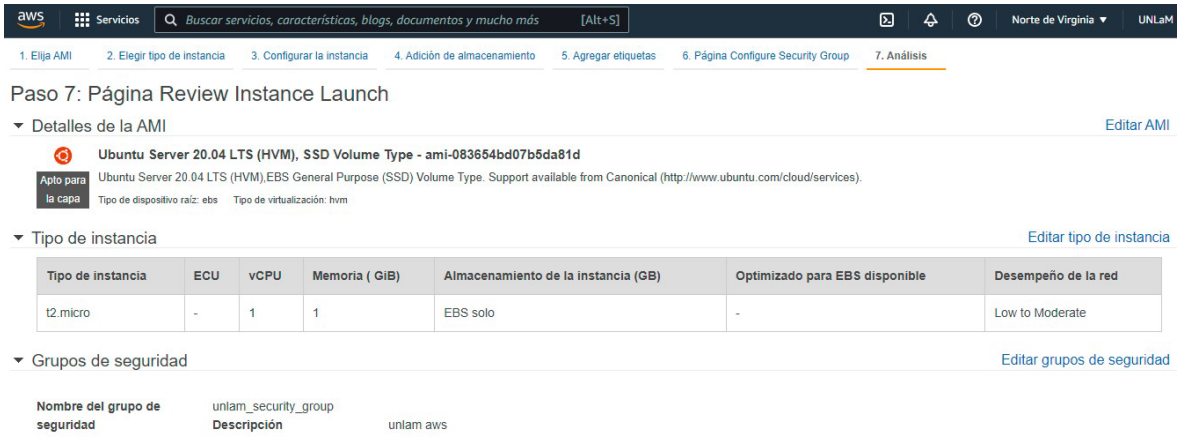

Ilustración 30. Instalación de una VM en AWS. Lanzamiento Paso 7.

Seguido a esto, se solicitará seleccionar un par de claves existente o crear uno nuevo. Para este trabajo creamos uno nuevo ya que no teníamos con anterioridad uno creado. Este par de claves sirve para conectarse a la instancia, nosotros lo haremos mediante SSH, como se configuró anteriormente.
Elegimos el tipo de par de claves, esto es, con que algoritmo se cifrarán, nosotros elegimos RSA y colocamos un nombre al par de claves, luego hacer clic en Descargar par de claves. Por último, hacemos clic en Lanzar instancia.

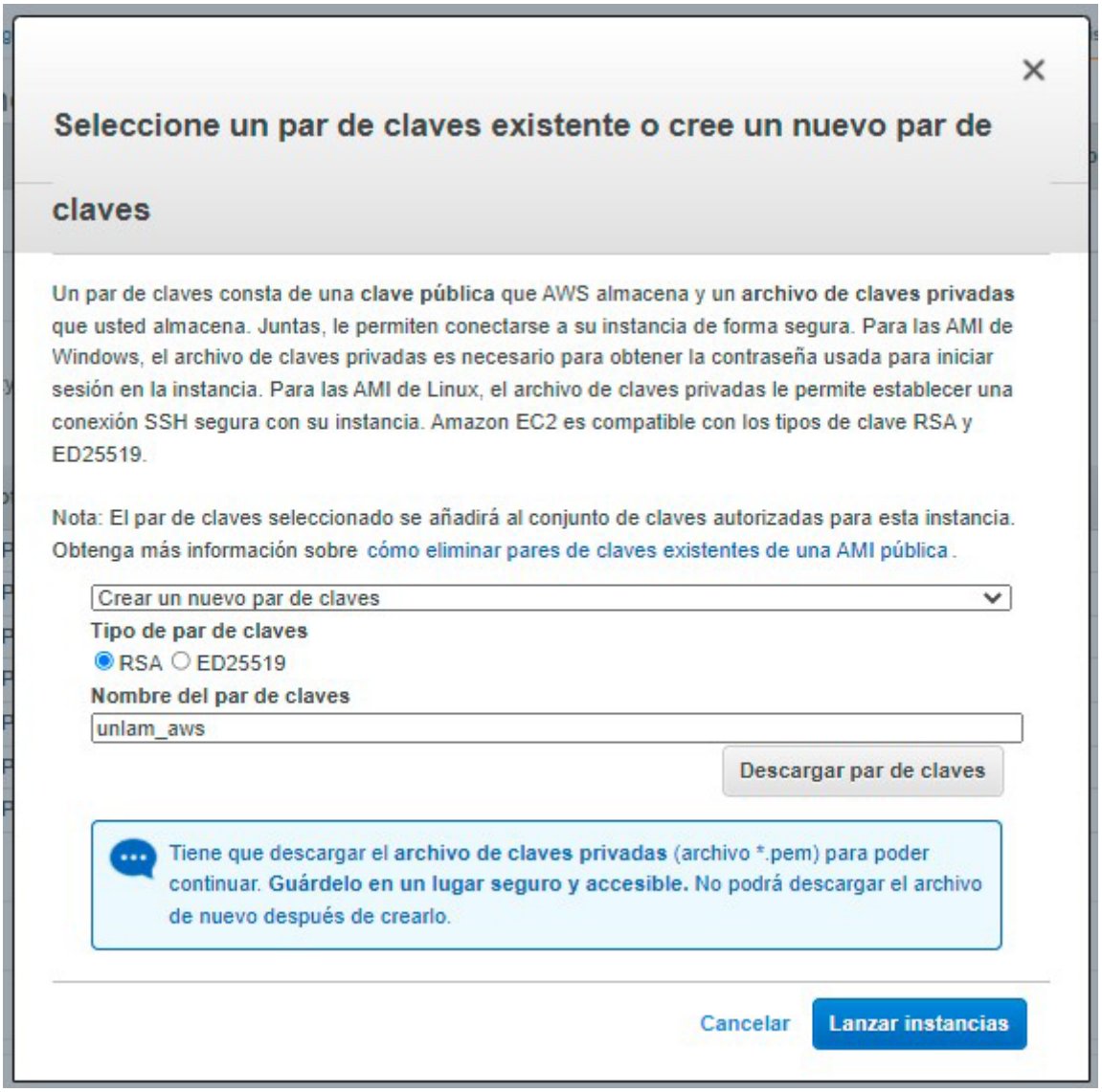

Ilustración 31. Instalación de una VM en AWS. Lanzamiento, creación de claves.

Se mostrará una pantalla que indica que la instancia se está lanzando. Esto tomará algunos minutos.

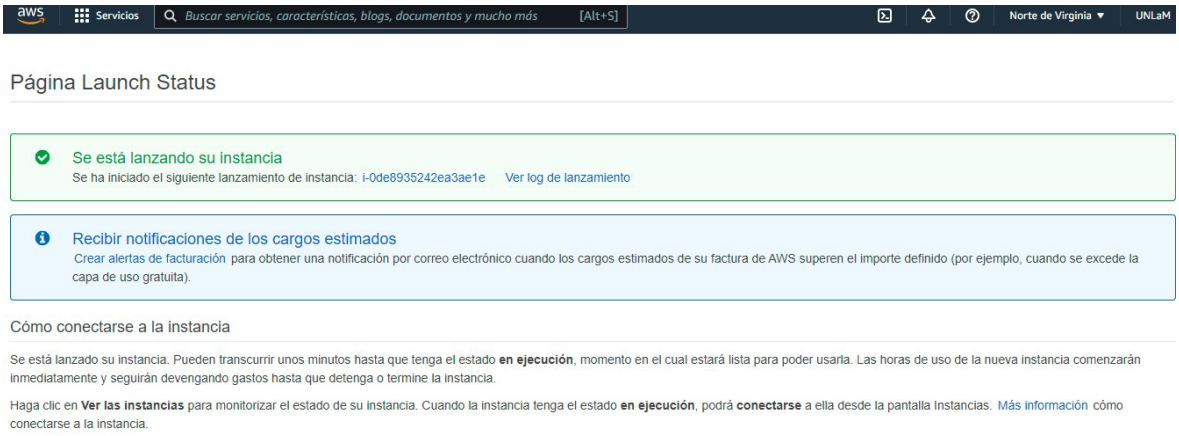

Ilustración 32. Instalación de una VM en AWS. Lanzamiento Paso final.

Por último, debemos hacer clic en el nombre de la instancia. Esto nos llevará nuevamente al listado de instancias y podrá visualizarse la instancia recientemente creada

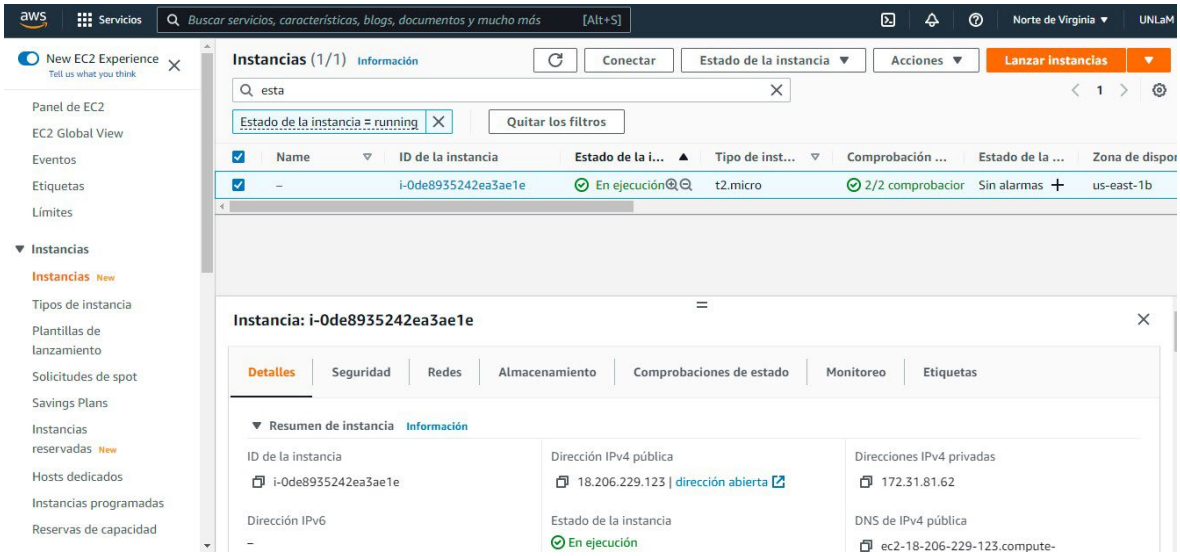

Ilustración 33. Instalación de una VM en AWS. Lanzamiento, visualización.

Conectarse a una Máquina Virtual en AWS

Se debe seleccionar la instancia a la que se desea conectar través del panel de EC2 y luego seleccionar la opción conectarse a la instancia. Seguido a esto, ir a la opción Cliente SSH, esto nos dará una breve, pero precisa indicación de como conectar con la instancia. Seguimos estos pasos y nos conectamos.

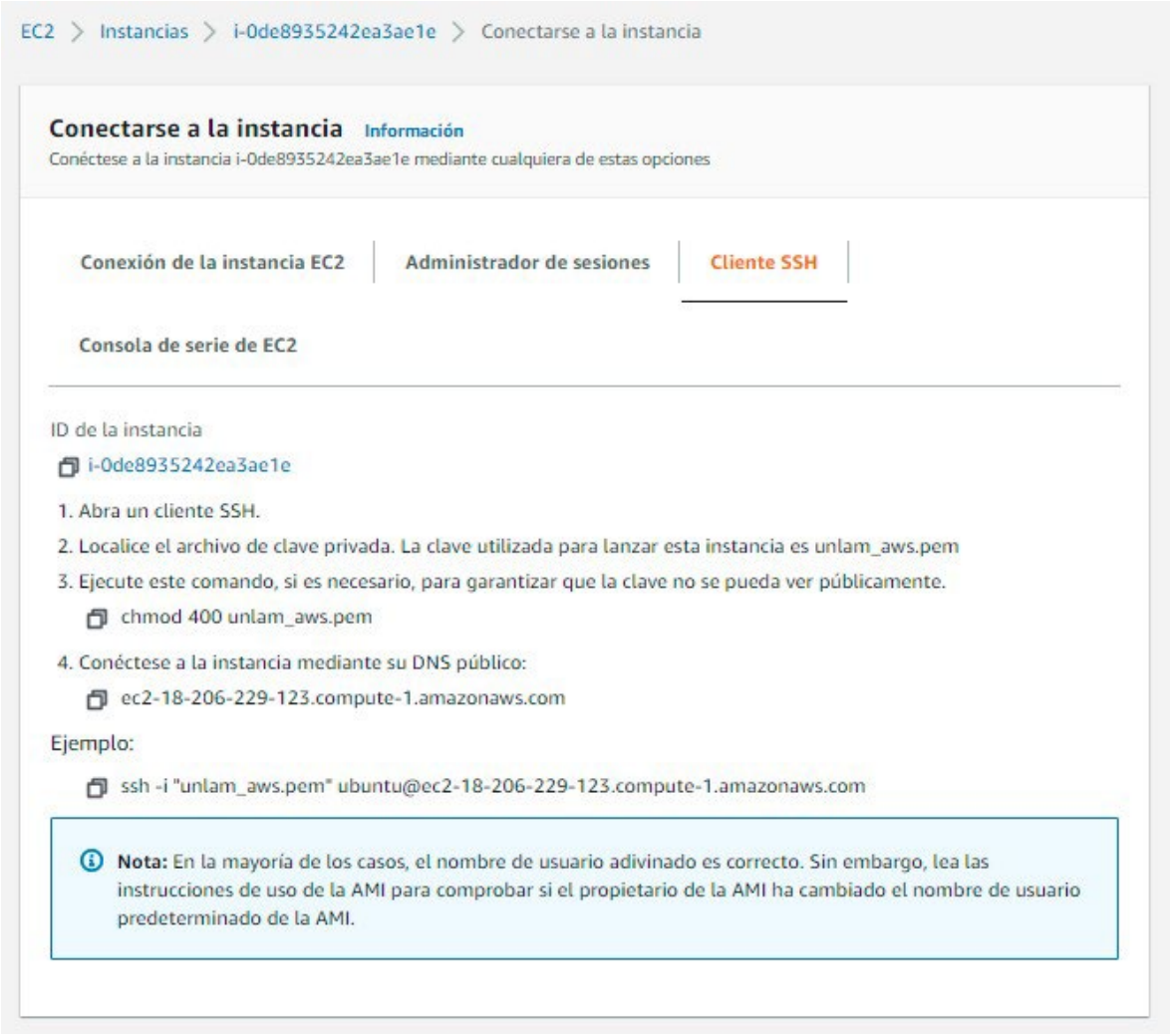

Ilustración 34. Conectarse a una VM en AWS.

Luego de esto podemos ver la conexión realizada a través de la línea de comandos.

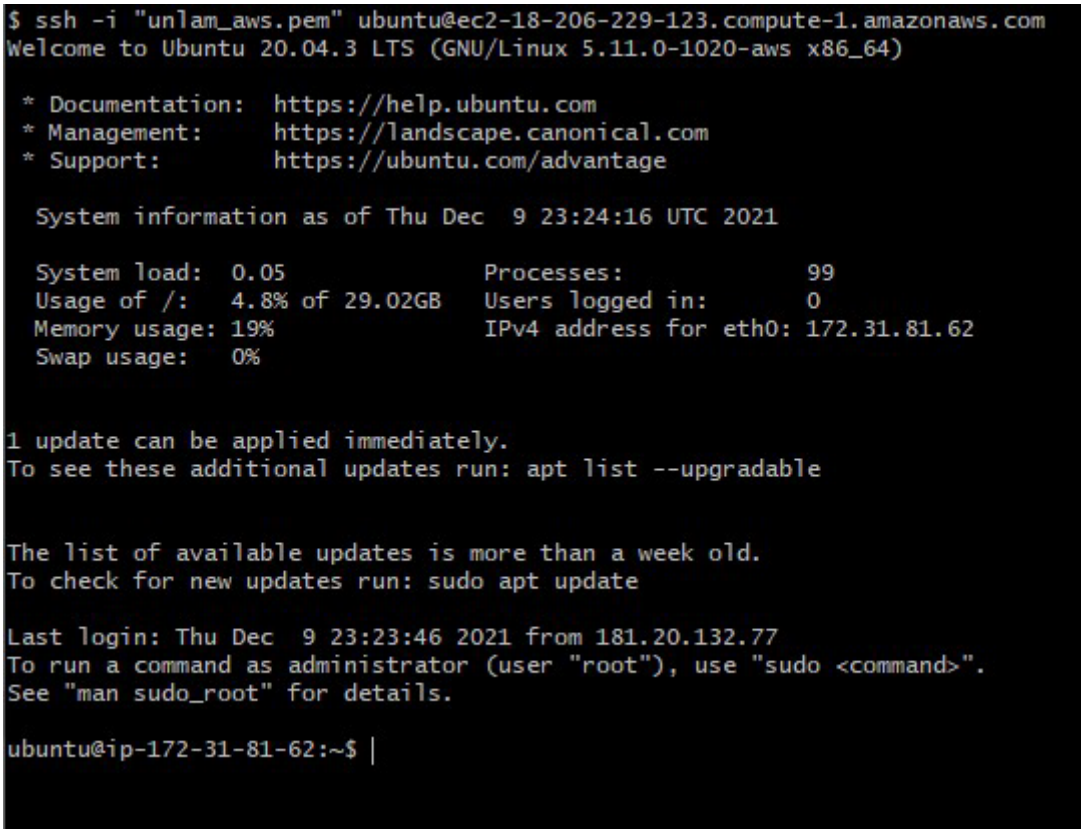

Ilustración 35. Conectarse a una VM en AWS. Consola

Instalación de un nodo Sellador en BFA

Verificaciones iniciales:

1. Verificar los requerimientos técnicos para un nodo sellador (Ver Anexo Requerimientos Técnicos para un nodo sellador).
2. El firewall del servidor debería:
    a. permitir conexiones entrantes a los puertos 22 (ssh), 3000 (API). Esta configuración responde a que tanto la API como el Nodo están en el mismo servidor. El puerto que queda público es el de la API.
    b. permitir conexiones salientes ICMP, UDP y TCP.
3. Instalar dependencias
    a. sudo apt-get install unzip zip wget git

Establecer configuración inicial

timedatectl set-timezone America/Argentina/Buenos_Aires
timedatectl set-ntp on
timedatectl
locale -k LC_TIME
locale -a
localectl set-locale LANG=es_AR.utf8
locale-gen es_AR.utf8
update-locale LANG=es_AR.utf8

localectl set-locale LANG=es_AR.utf8
locale -a

Instalar Nodo

Clonar el repositorio del nodo
cd /
sudo git clone https://gitlab.bfa.ar/blockchain/nucleo.git bfa

Iniciar la instalación del nodo
cd /bfa/bin/
./installbfa.sh

Elegir el path por defecto (/home/bfa/bfa/) y seleccionar 1. Producción en red.

Iniciar nodo
su - bfa
start.sh

El comando inicia la sincronización y puede demorar. El proceso es en background.

Para verificar el estado
su - bfa
bfalog.sh

Se pueden ejecutar comandos RPC luego de ejecutar
su - bfa
attach.sh

Crear cuenta en Nodo

Crear la cuenta BFA
su - bfa
admin.sh account

Verificar la configuración del nodo
su - bfa
attach.sh
web3.eth.accounts

Con ese address se crea la cuenta en BFA.
Para verificar la cuenta es necesario seguir los pasos de la cuenta: https://registro.bfa.ar/accounts/list/

Tener en cuenta que no es necesario tener la cuenta verificada para poder enviar transacciones.

La PASSPHRASE tiene que ir vacía ya que al crear con admin.sh account se crea sin passphrase.

Verificar servicio

Ejecutar en el servidor un comando cURL para verificar si hay respuesta

curl --location --request POST 'localhost:8545/' --header 'Content-Type: application/json' --data-raw '{
"jsonrpc":"2.0",
"method":"web3_sha3",
"params":["0x68656c6c6f20776f726c64"],
"id":64
}'

Si la respuesta es exitosa, el nodo se encuentra instalado correctamente.

Comandos útiles
eth.chainId()
net.version
web3.eth.accounts[0]
web3.eth.getBalance(web3.eth.accounts[0])
para salir: exit

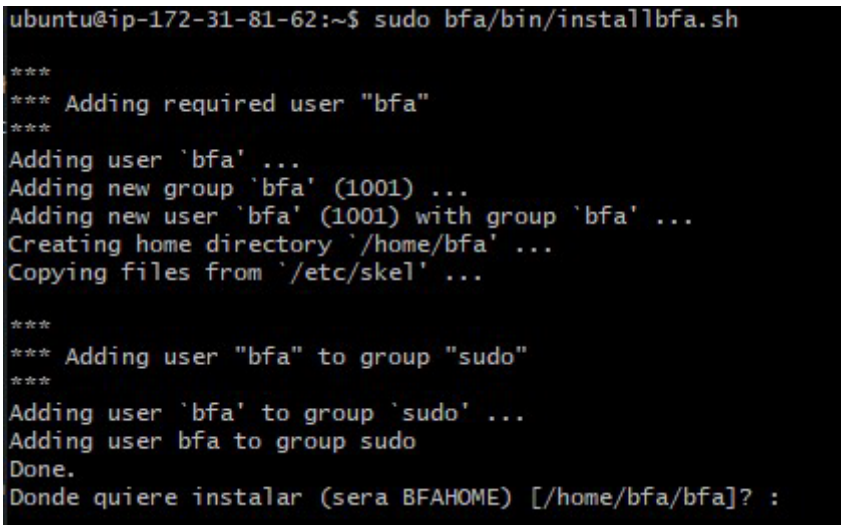

Ilustración 36. Instalación Nodo Sellador. Consola.

Requerimientos técnicos para un nodo sellador:

2 vCPU

4GB de RAM
1TB de espacio en disco (se puede comenzar con menos, pero se debe tener la posibilidad de aumentar el tamaño del volumen en el futuro)

20 GB para SO
El resto en "/home" o "/home/bfa" en un LVM2 (soporta snapshot y lvresize)

Placa de red de 1Gbps
Recomendado Debian o Ubuntu Server.

Direcciones IP:
IPv4 (fija, pública, sin NAT).
(opcional, pero fuertemente recomendado) IPv6 (fija, pública, sin NAT).

Firewall (iptables + ip6tables)
30303/tcp y 30303/udp abierto a todos los selladores y todos los gateways.
Tal vez SSH para administración de su propia red.
El resto cerrado para que Geth no se conecte a "todo Internet".

Políticas de Seguridad aplicadas al Nodo Sellador

Siguiendo el Marco de Buena Arquitectura de AWS, se aplicaron las siguientes Políticas de Seguridad durante la implementación e instalación del nodo sellador.

Administración de identidad y acceso.

- Se protegió al usuario raíz de la cuenta de AWS.
- Se creó un usuario raíz para el alta de la cuenta y el tratamiento de las cuestiones de facturación, pero luego se resguardó esta cuenta, y se creó una nueva cuenta del tipo administrador para el resto de las cuestiones relacionadas con la administración de las cuentas. Posteriormente también se creó otra cuenta para interactuar con el nodo.

- Se utilizó la autenticación multifactor sin ningún tipo de excepción al respecto.

- Se cumplieron los requisitos para la generación de contraseñas recomendadas por AWS.
- Las contraseñas se rotan con una regularidad establecida en 30 días sin excepción.

Pantallas de la dApp

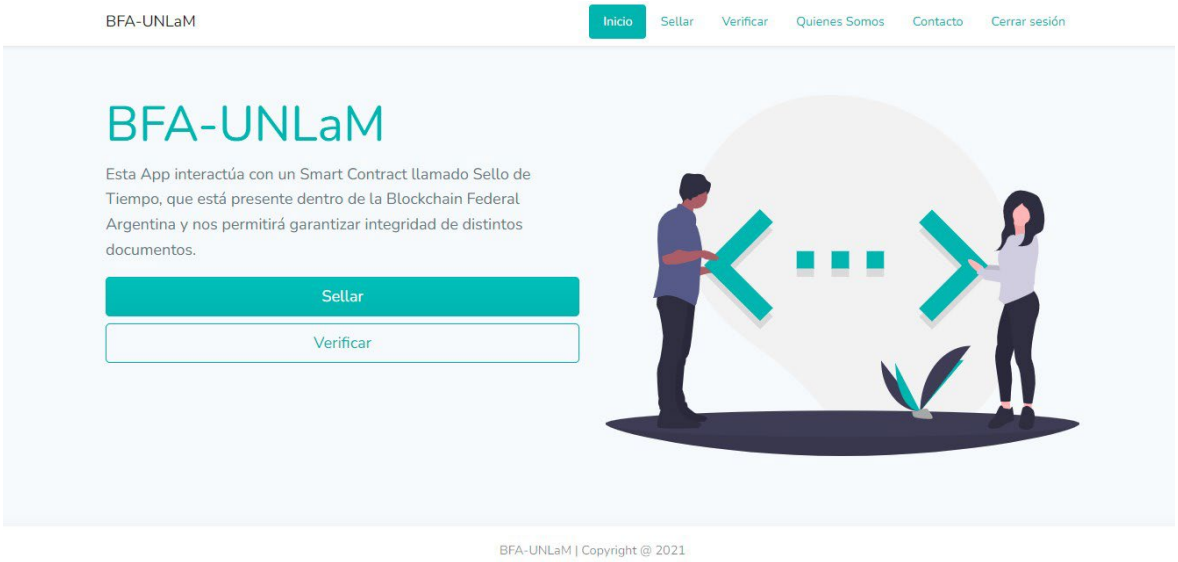

Ilustración 37. Pantalla dApp. Inicio

Ilustración 38. Pantalla dApp. Sellar

Ilustración 39. Pantalla dApp. Verificar

Ilustración 40. Pantalla dApp. Quienes Somos

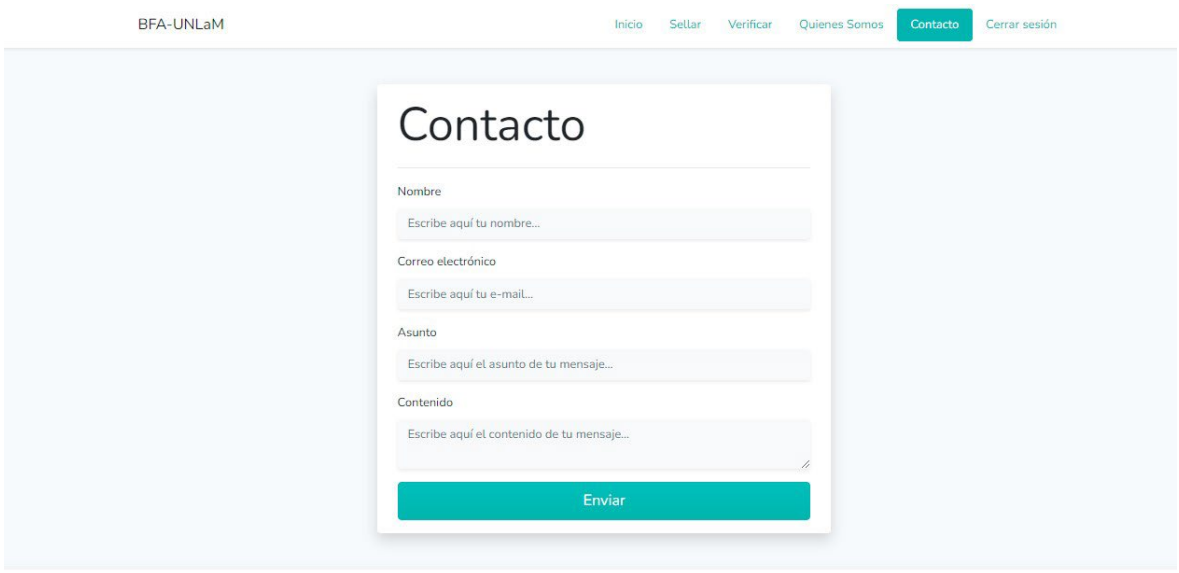

Ilustración 41. Pantalla dApp. Contacto

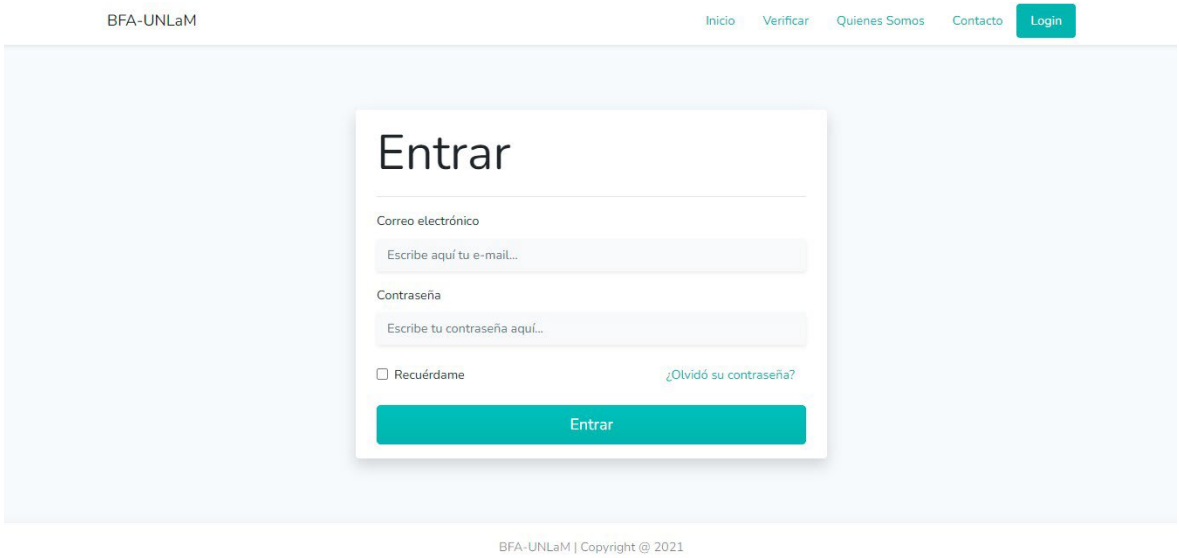

Ilustración 42. Pantalla dApp. Login

9. TERMINOLOGÍA Y ABREVIATURAS

Address: Dirección de un usuario o un Smart Contract dentro de una Blockchain.

AWS: Amazon Web Services. Es un conjunto de servicios de cómputo en la nube, administrados por la empresa Amazon.

BAT: Es un token de Ethereum que utiliza la plataforma de publicidad digital basada en una cadena de bloques del software de Brave.

BBS: (Bulletin Board System) software que permitía a computadoras intercambiar mensajes, archivos o jugar en línea entre sí, conectados a la línea telefónica.

BFA: Blockchain Federal Argentina. Es la primera red multiservicios de Argentina. Administra una red permisionada montada sobre la Blockchain Ethereum.

Bitcoin: Es la primera criptomoneda creada. Funciona como sistema de pago entre los usuarios que lo utilizan, sin la necesidad de un Canco Central

Bitcoin Script: Lenguaje de programación simple y limitado (no es Turing Complete) empleado en Bitcoin para el procesamiento de las transacciones.

BitTorrent: protocolo de intercambio de datos de forma descentralizada. Permite descargar un archivo desde diversos puntos al mismo tiempo.

Blockchain: Cadena de Bloques. Es una estructura de datos cuya información se agrupa en conjuntos (bloques) a los que se le añade información relativa a un bloque inmediatamente anterior de la cadena.

Bloque: Es una estructura de datos que contiene un conjunto de transacciones y otra información adicional. Éstos se enlazan con otros bloques formando una Blockchain.

BTC: Es la criptomoneda (token) propiamente dicha de la red Bitcoin.

Cloud Computing: es una tecnología que permite acceso remoto a software, almacenamiento de archivos y procesamiento de datos por medio de Internet.

Contact Less: es un sistema de pago que permite pagar una compra mediante tecnologías de identificación por radiofrecuencia incorporadas en tarjetas de crédito o débito, llaveros, tarjetas inteligentes, teléfonos móviles u otros dispositivos.

Criptografía: es una disciplina que se ocupa de las técnicas de cifrado destinadas a alterar las representaciones de ciertos mensajes con el fin de hacerlos ininteligibles a receptores no autorizados.

Cryptokitties: es un juego de Blockchain en Ethereum desarrollado por Axiom Zen que permite a los jugadores comprar, recolectar, criar y vender gatos virtuales únicos.

DAO: Organización Autónoma Descentralizada. Es una organización que está dirigida a través de reglas codificadas en Smart Contracts.

dApp: Una aplicación descentralizada es una aplicación informática que funciona en un sistema de computación distribuido.

DC++: es un software del tipo cliente peer-to-peer libre. Fue desarrollado inicialmente por Jacek Sieka.

DCC: Direct Client-to-Client, es un protocolo de Internet Relay Chat (IRC) que permite interconectar dos peers o puntos usando un servidor IRC para intercambiar archivos.

DeFi: Finanzas descentralizadas. Son una forma experimental de finanzas que no dependen de intermediarios centrales para ofrecer instrumentos financieros tradicionales, en cambio utilizan Smart Contracts.

DEX: Decentralized Exchanges. Son plataformas online donde se pueden realizar intercambios de criptoactivos sin la participación de intermediarios que cobren comisiones ni de una autoridad reguladora.

Dropbox: es un servicio de alojamiento de archivos multiplataforma en la nube, operado por la compañía estadounidense Dropbox.

Eclipse Attack: es un tipo de ataque que busca aislar y atacar un usuario específico que forme parte de una red Blockchain. El objetivo es poder manipular los datos que recibe el usuario desde la red.

EOA: Externally Owned Account. Es una dirección que representa a un usuario en una Blockchain. Es decir, no representa a un Smart Contract.

Erebus Attack: es un tipo de ataque de denegación de servicios distribuido o DDoS, que tiene la capacidad de tomar el control de toda una red de criptomoneda hasta el punto de volverla inutilizable.

ETH: Es la criptomoneda (token) propiamente dicha de la red Ethereum.

Ethereum: es una plataforma descentralizada y open source, que sirve para programar contratos inteligentes.

EVM: máquina virtual que se coloca entre el sistema operativo y la capa de aplicación para mitigar la dependencia del sistema operativo.

EVM bytecode: Lenguaje de programación de bajo nivel que se compila a partir de un lenguaje de programación de alto nivel como Solidity. Es un código intermedio más abstracto que el código máquina.

Foursquare: es un servicio basado en localización web aplicada a las redes sociales. La geolocalización permite localizar un dispositivo fijo o móvil en una ubicación geográfica.

FT: Fungible Token. Es un token que puede fraccionarse. Sirven para intercambiarlo por otras cosas, incluso del mundo real. Además, permiten la transferencia a otros usuarios. Las criptomonedas como el Bitcoin, o Ether son fungibles, pueden intercambiarse.

Gnutella: es un proyecto de software distribuido para crear un protocolo de red de distribución de archivos entre pares, sin un servidor central.

Google Cloud: es un conjunto de servicios en la nube que brinda la empresa Google. Permite realizar una serie de tareas que requieren hardware y software por medio de la nube de Google como única forma de acceso, almacenamiento y gestión de datos.

Gossip Protocol: es un protocolo que permite diseñar sistemas de comunicaciones distribuidos (P2P) altamente eficientes, seguros y de baja latencia.

Hash: es el resultado de una función hash, la cual es una operación criptográfica que genera identificadores únicos e irrepetibles a partir de una información dada.

HootSuite: es una plataforma web y móvil para gestionar redes sociales por parte de personas u organizaciones, creada por Ryan Holmes en 2008

IBM Cloud: es una nube empresarial en la que se puede diseñar y desarrollar soluciones tecnológicas como aplicaciones web o móviles en cualquier tipo de negocio de cualquier tamaño.

IRC: es un protocolo de comunicación en tiempo real basado en texto, que permite la comunicación entre dos o más personas.

ISP: Internet Service Provider. El proveedor de servicios de internet es la empresa que brinda conexión a Internet a sus clientes.

Kademlia Protocol: es un protocolo para la implementación de una tabla de hash distribuida, desarrollado en la universidad de Nueva York por David Mazières y Petar Maymounkov en el año 2002.

Lenguaje Ensamblador (assembly): es un lenguaje de programación de bajo nivel. Consiste en un conjunto de mnemónicos que representan instrucciones básicas para los procesadores.

LISP: es un lenguaje de programación de computadoras de tipo multiparadigma con larga historia y una inconfundible y útil sintaxis homoicónica basada en la notación polaca.

Mastodon: es una red social libre y descentralizada de microblogging, similar a Twitter, lanzada en octubre de 2016 bajo el dominio principal mastodon.social.

MFA: Multi-factor Authentication. La autenticación de múltiples factores es un método de control de acceso en el que a un usuario se le concede acceso a una aplicación solo después de que presente dos o más pruebas diferentes de que es quien dice ser. Estas pruebas

pueden ser diversas, como una contraseña, que posea una clave secundaria rotativa, o un certificado digital instalado en el equipo.

Microsoft Azure: es un servicio de computación en la nube creado por Microsoft para construir, probar, desplegar y administrar aplicaciones y servicios.

MITM: Man In The Middle. Es un ataque informático que consiste en un usuario que se sitúa en el medio de una comunicación entre otros usuarios y roba la información que se envían.

Módem: es un dispositivo que convierte las señales digitales en analógicas y viceversa, y permite así la comunicación entre computadoras a través de la línea telefónica o del cablemódem.

Napster: es un servicio de distribución de archivos de música. Fue la primera gran red P2P de intercambio creado por Sean Parker y Shawn Fanning.

NFT: Non-fungible token. Un token no fungible, es un tipo especial de token criptográfico que representa algo único. Los tokens no fungibles no son, por tanto, mutuamente intercambiables.

Nodo: es todo aquel dispositivo conectado a una red Blockchain, que ejecuta el software que se encarga de todo su funcionamiento.

Nodo Minero: es un nodo cuya tarea es sellar los bloques que se van armando con un conjunto de transacciones. Estos nodos sellan en un esquema de PoW.

Nodo Sellador: es un nodo cuya tarea es sellar los bloques que se van armando con un conjunto de transacciones. Estos nodos sellan en un esquema de PoA.

Nodo Transaccional: es un nodo cuya tarea es enviar transacciones a la Blockchain. Estas transacciones luego son sumadas a un bloque y selladas por los nodos selladores o mineros dependiendo del esquema utilizado.

Open Source: es el software cuyo código fuente y otros derechos que normalmente son exclusivos para quienes poseen los derechos de autor, forman parte del dominio público a través de una licencia a tal efecto.

OSI: Open Systems Interconnection. El modelo de interconexión de sistemas abiertos es un modelo de referencia para los protocolos de la red, creado en el año 1980 por la Organización Internacional de Normalización.

P2P: o Peer-to-Peer. Es un tipo de red de computadoras descentralizada, en la que las conexiones se realizan sin servidores, o sea está conformada por una serie de nodos que se comportan como iguales entre sí.

PoA: Proof of Authority. Es un protocolo de consenso que consiste en que los nodos selladores ponen su identidad real y reputación como garantía de transparencia

conformando un conjunto conocido y autorizado de nodos selladores que son seleccionados para sellar sin competir entre ellos.

PoW: Proof of Work. Es el más conocido y antiguo protocolo de consenso que consiste en que los nodos mineros de una red realicen con éxito un trabajo computacionalmente costoso para acceder a sellar un bloque y obtener una recompensa por ello.

Smart Contract: es un programa almacenado en la Blockchain. Está almacenado de forma inmutable y transparente.

Solidity: es un lenguaje de programación orientado a objetos para escribir contratos inteligentes en varias plataformas Blockchain, la más destacada es Ethereum.

TCP/IP: La familia de protocolos de internet es un conjunto de protocolos de red en los que se basa internet y que permiten la transmisión de datos entre computadoras.

Transacción: es una operación que se realiza para agregar información a una Blockchain. Puede ser simplemente una línea de texto, o incluso un hash de un documento almacenado fuera de la cadena de bloques.

Turing completo: es una característica de los lenguajes de programación que indica que éste tiene un poder computacional equivalente a una Máquina de Turing Universal. Dicho de forma más fácil, puede realizar cualquier tipo de cálculo.

UNLaM: Universidad Nacional de La Matanza. Es una universidad pública de Argentina. Fue fundada el 29 de septiembre de 1989 en San Justo, ciudad cabecera del partido de La Matanza.

USENET: Users Network, consistente en un sistema global de discusión en Internet, que evoluciona de las redes UUCP. Fue creado por Tom Truscott y Jim Ellis, estudiantes de la Universidad de Duke, en 1979.

UUCP: Unix to Unix Copy Protocol. Es una utilidad UNIX estándar que administra la transmisión de información entre sistemas UNIX, utilizando conexiones en serie y líneas telefónicas regulares. Fue desarrollado en Bell Laboratory por Mile Lesk a mediados de la década de 1970.

WAF: Well Architectured Framework. Es el Marco de Buena Arquitectura de AWS, esta es un conjunto de recomendaciones que sirven como guía para el diseño de infraestructuras.

WWW: (World Wide Web) es un sistema para la distribución de información donde los documentos y otros recursos web son identificados por una URL e interconectados a través de hiperenlaces y son accesibles a través de internet.