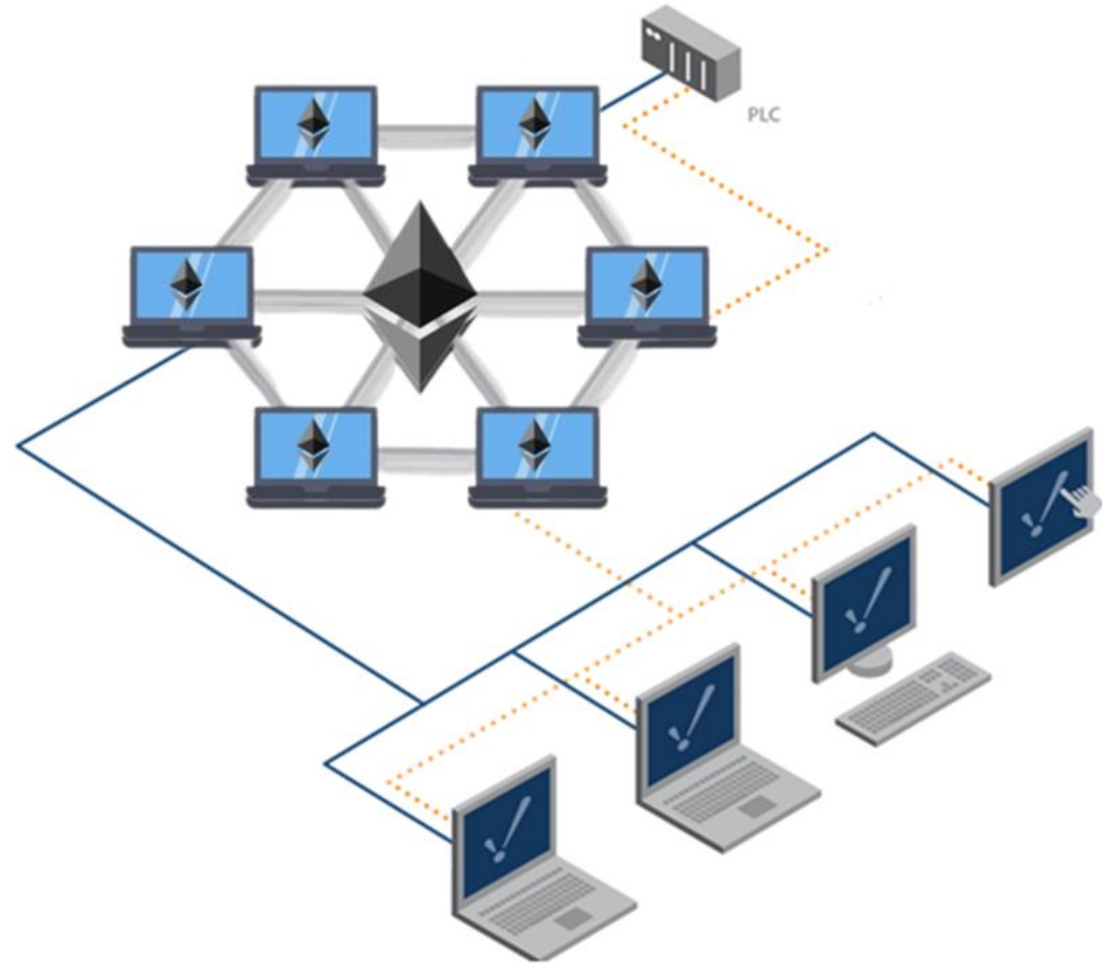


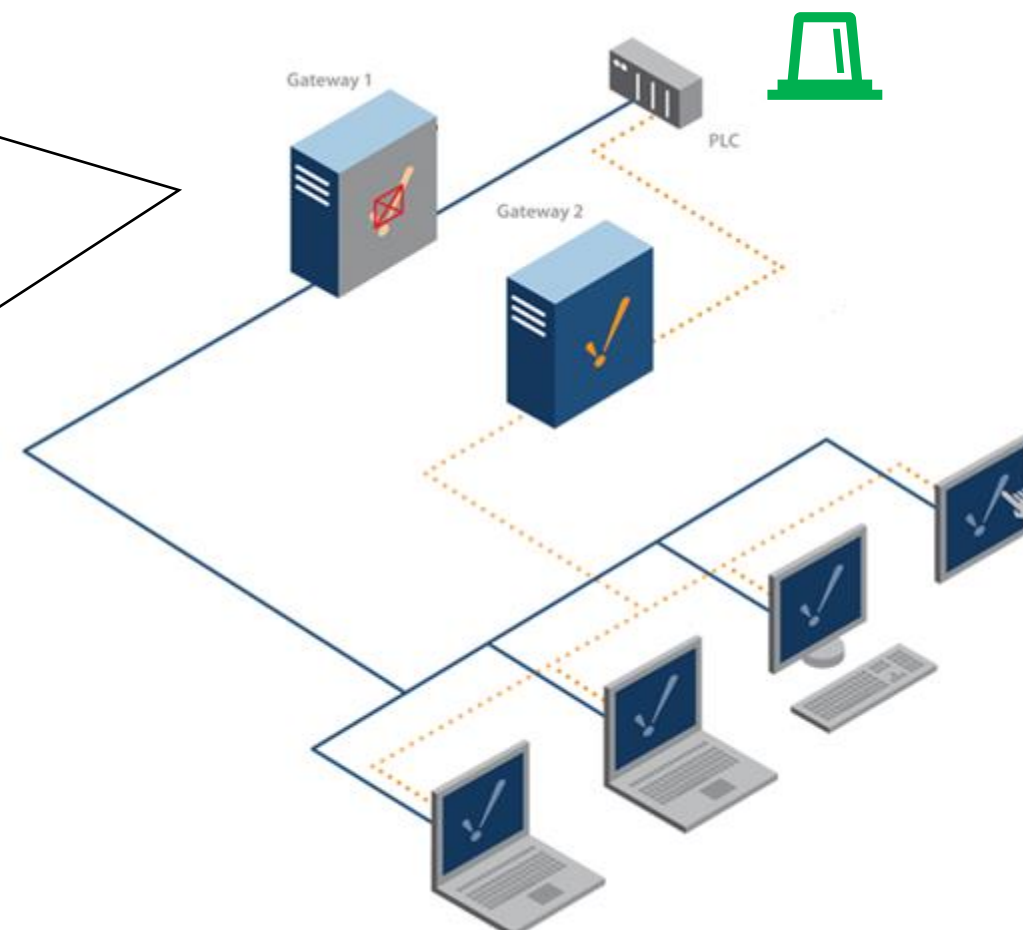
SCADA Descentralizado

V0.2



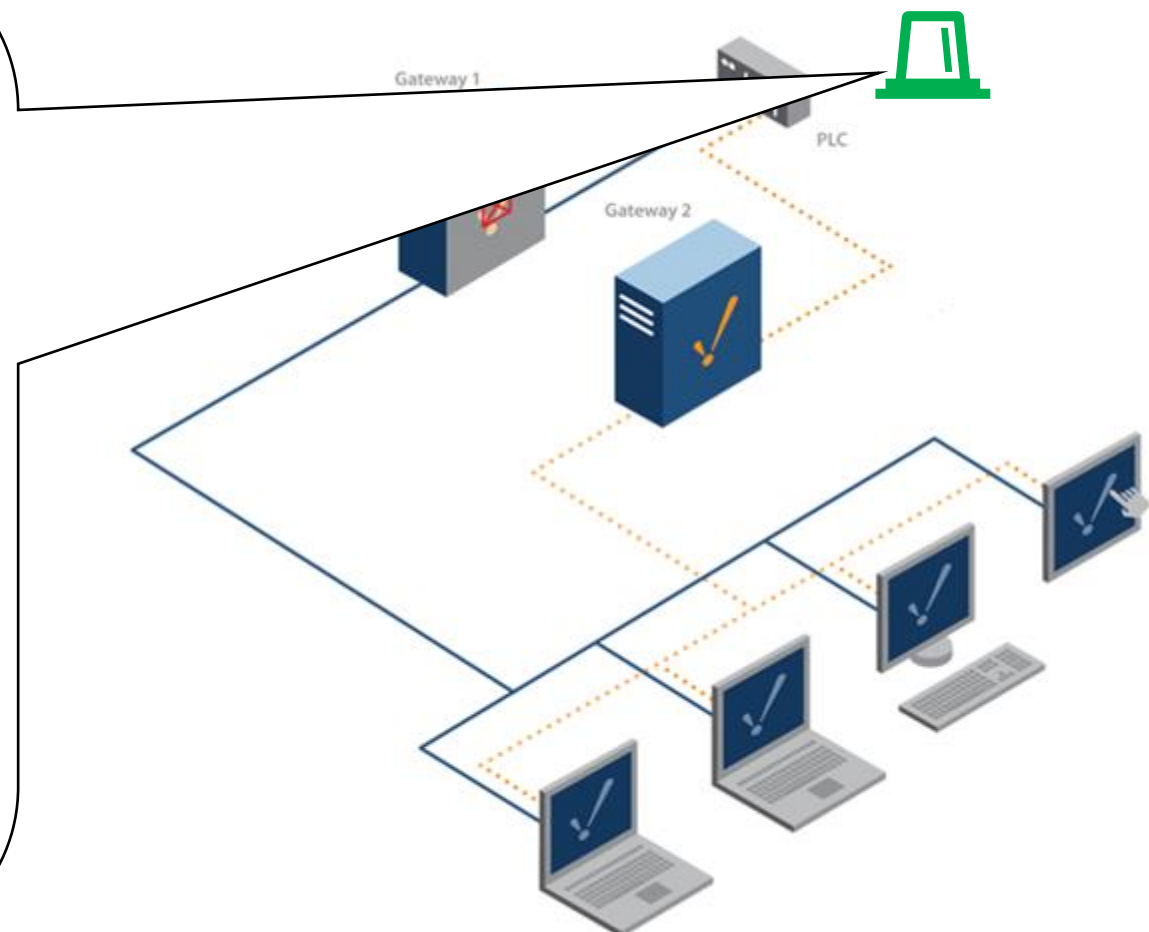
Arquitectura redundante de SCADA convencional

- Convencionalmente se tienen dos servidores.
- Uno solo está activo. Si esta falla, se active el otro.
- Ambos contienen la información actualizada.
- En la imagen hay un PLC que es un equipo que obtiene señales de la realidad y las reporta a los servidores a través de un protocolo de comunicación.
- El servidor activo recibe las señalizaciones, y las almacena en una base de datos. Se asegura de copiarle la información al otro servidor.
- Los HMI presentan la información de una forma visual e intuitiva a las personas para que observen qué está ocurriendo y puedan realizar acciones de operación/mantenimiento.



Arquitectura redundante de SCADA convencional

- Un PLC puede reportar multiples señalizaciones que pueden ser eventos, alarmas, estados de equipos, mediciones analógicas. También puede recibir un commando para operar un equipo.
- Ejemplo: Puede comunicar que un interruptor está abierto. Al recibir un commando de cierre, opera el interruptor y cuando este cierre, reporta el nuevo estado.
- Los servidores pueden estar comunicados con multiples PLCs en simultáneo.
- Cada PLC está identificado según su protocolo/medio de Comunicaciones por una IP y/o número de esclavo y/o technical key.



Arquitectura redundante de SCADA convencional

Registro de eventos

18/04/2023 15:05:23.123	Equipo ABC	Normal

Pantalla de visualización

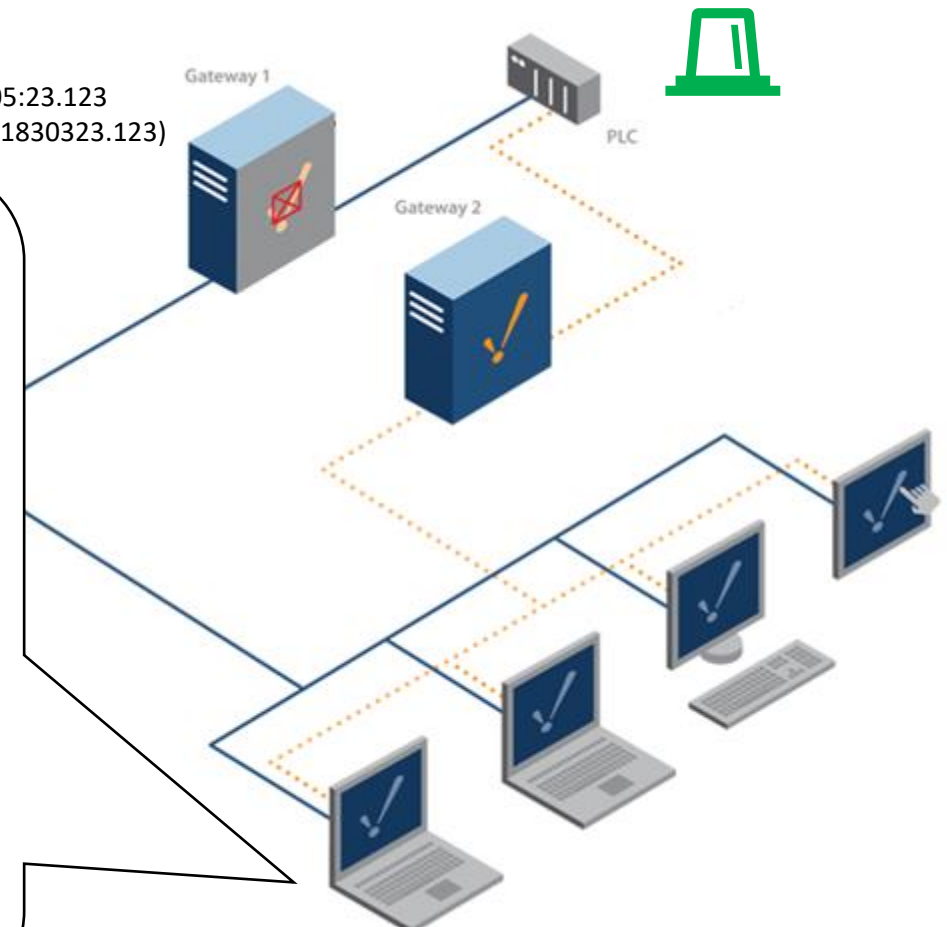


4/25/2023

- Value = 0 (normal)
- Quality = 0 (ok)
- TimeStamp = 18/04/2023 15:05:23.123
(UNIX format = 1681830323.123)

- Las pantallas básicas que ven las personas son las de eventos/alarmas donde observan una especie de log de lo que va ocurriendo.
- También tienen un esquema que les permite de forma intuitiva comprender el estado del Sistema.

Danman Ariel Winnick



Arquitectura redundante de SCADA convencional

Registro de eventos

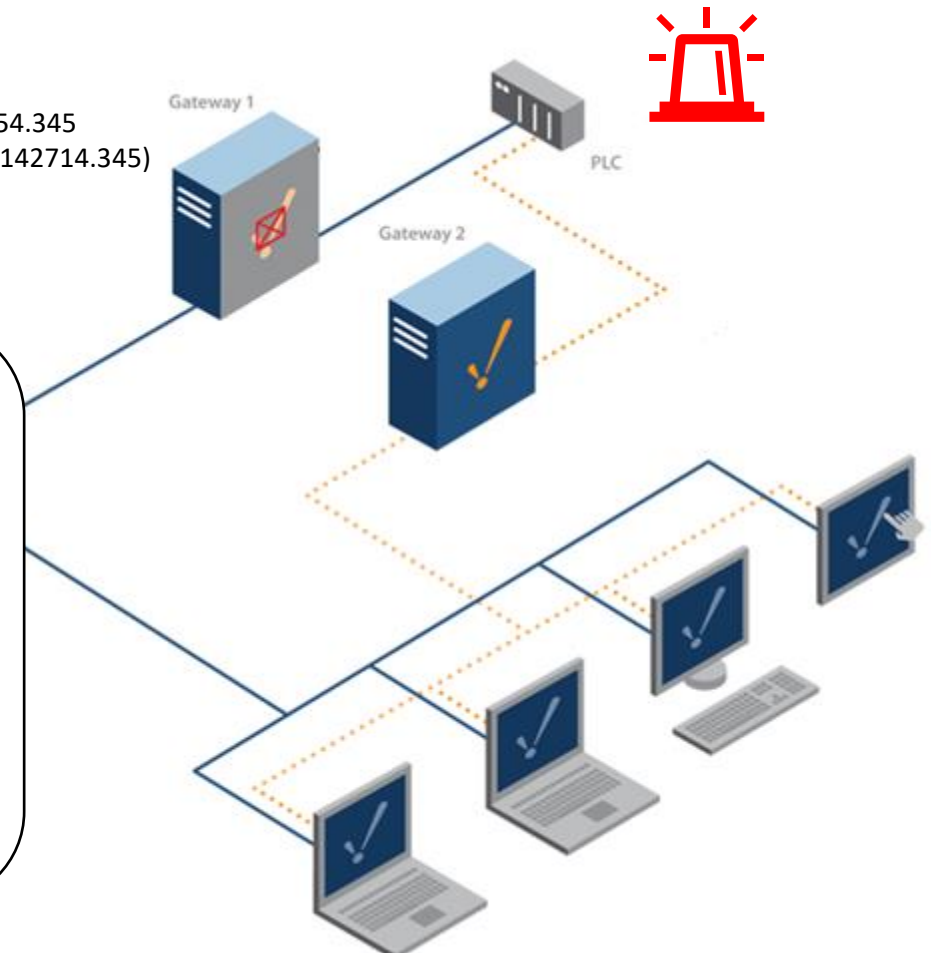
18/04/2023 15:05:23.123	Equipo ABC	Normal
22/04/2023 5:51:54.345	Equipo ABC	Alarma

Pantalla de visualización



- Value = 1 (Alarm)
- Quality = 0 (ok)
- TimeStamp = 22/04/2023 5:51:54.345
(UNIX format = 1682142714.345)

- El PLC se comunica con los servidores cuando hay un cambio de alguna señal y le informa a qué hora ocurrió.
- Esa información el HMI la representa en sus pantallas.



Registro de eventos

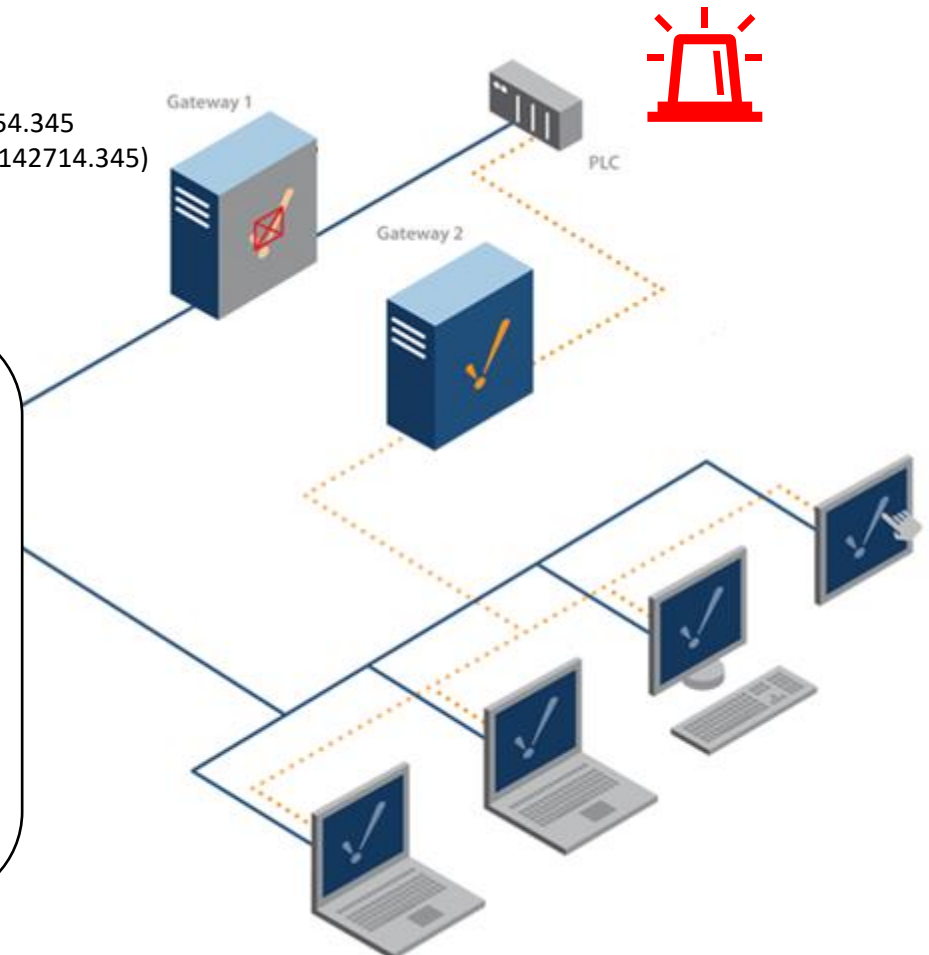
18/04/2023 15:05:23.123	Equipo ABC	Normal
22/04/2023 5:51:54.345	Equipo ABC	Alarma

Pantalla de visualización



- Value = 1 (Alarm)
- Quality = 0 (ok)
- TimeStamp = 22/04/2023 5:51:54.345
(UNIX format = 1682142714.345)

- Esa información es ordenada y presentada al usuario
- El HMI sabe que un valor 0 = ok y un 1 = Alarma



Arquitectura redundante de SCADA convencional

Registro de eventos

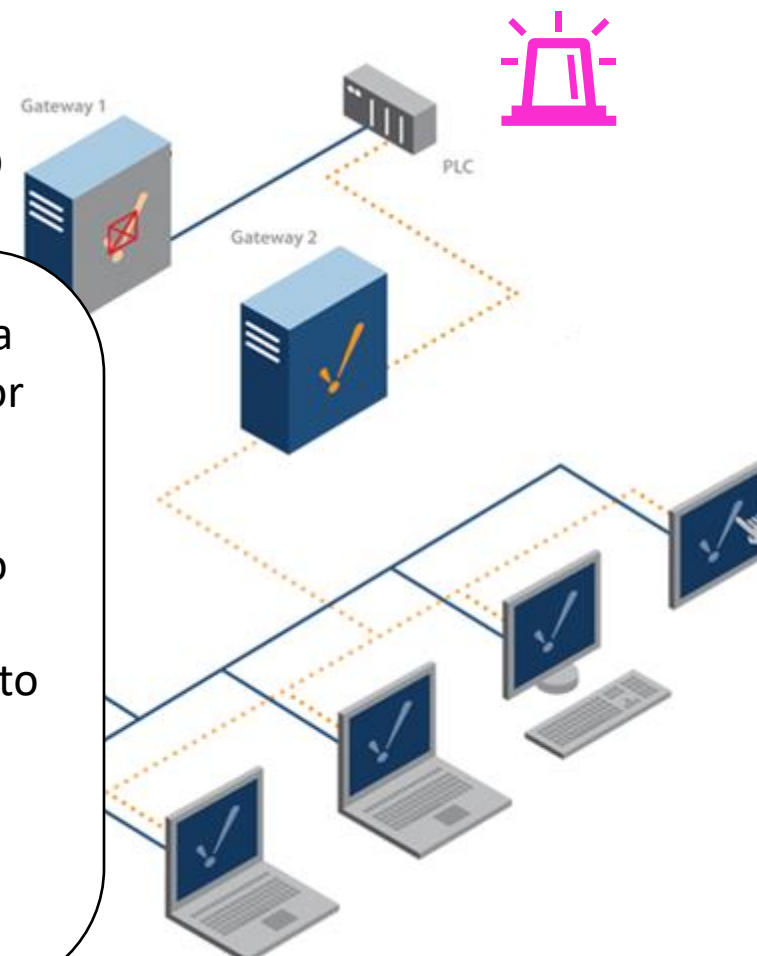
18/04/2023 15:05:23.123	Equipo ABC	Normal
22/04/2023 5:51:54.345	Equipo ABC	Alarma
22/04/2023 5:51:54.345	Equipo ABC	Desactualizado

Pantalla de visualización



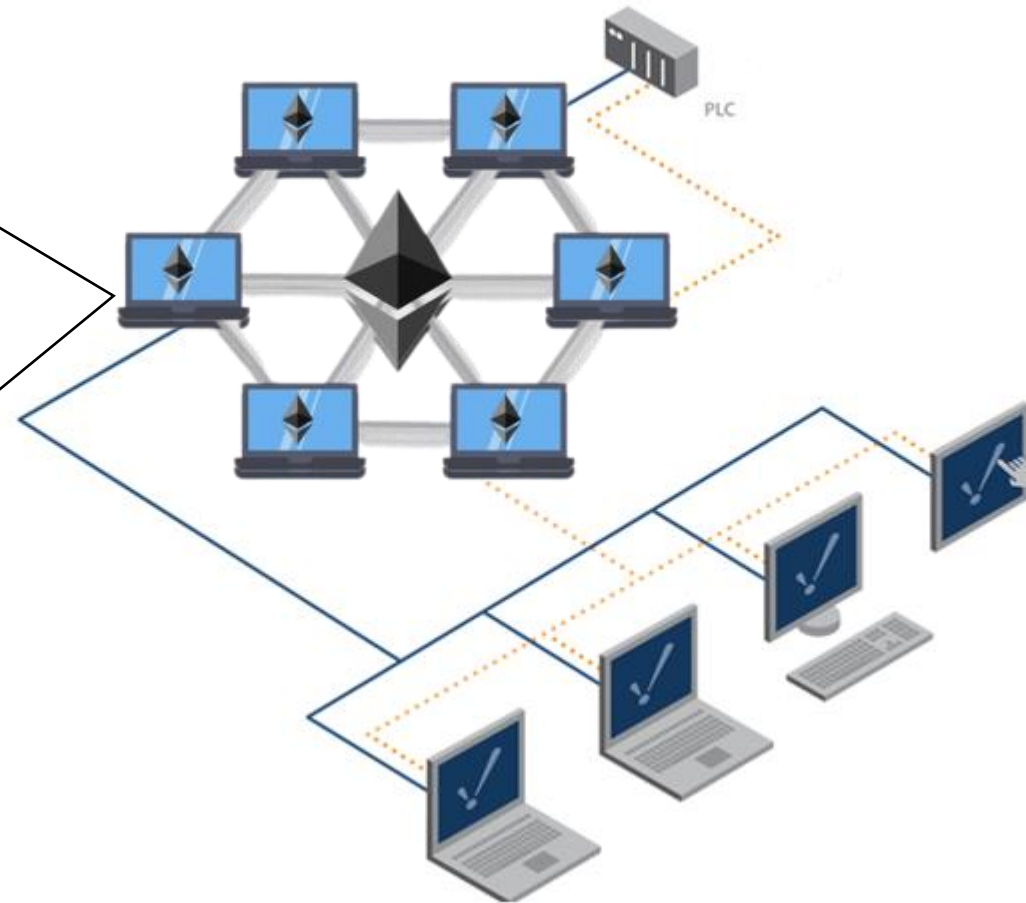
- Value = 1 (Alarm)
- Quality = 2 (Unknown – Showing last value)
- TimeStamp = 22/04/2023 5:51:54.345
(UNIX format = 1682142714.345)

- La calidad del dato se asocia a cuando el PLC informa un valor pero no está “Seguro” de que esté bien.
- Puede ser por ejemplo que no esté seguro de cómo se encuentra el equipo, por lo tanto lo reporta con calidad = 2.
- O puede ser que no sepa la hora, entonces lo indica con una calidad = 3.



Arquitectura SCADA Descentralizado

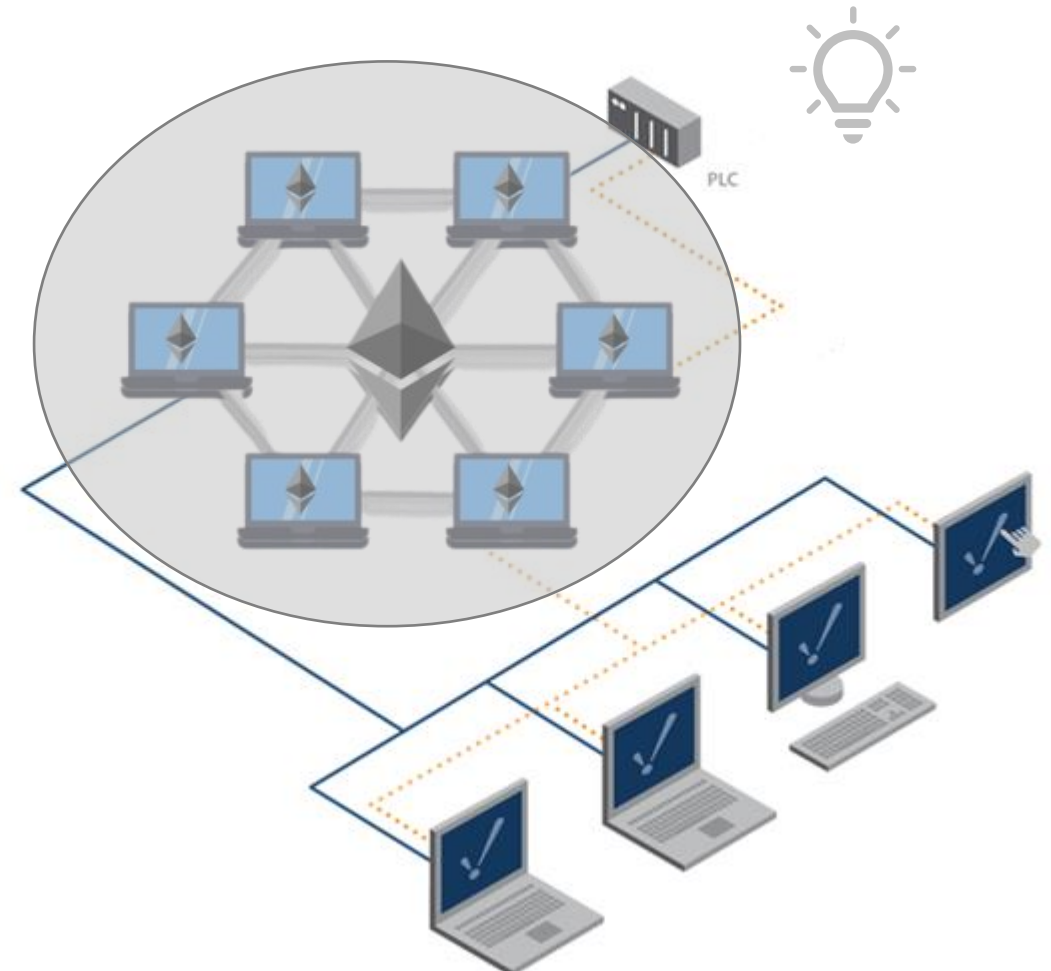
- La propuesta es descentralizar los servidores. De esa forma:
 - La base de datos y sus eventos del pasado almacenados en la blockchain serán inmutables.
 - Mejora la seguridad del al disminuir la cantidad de puntos de falla



SCADA Descentralizado – Etapa 1

Smart Contract

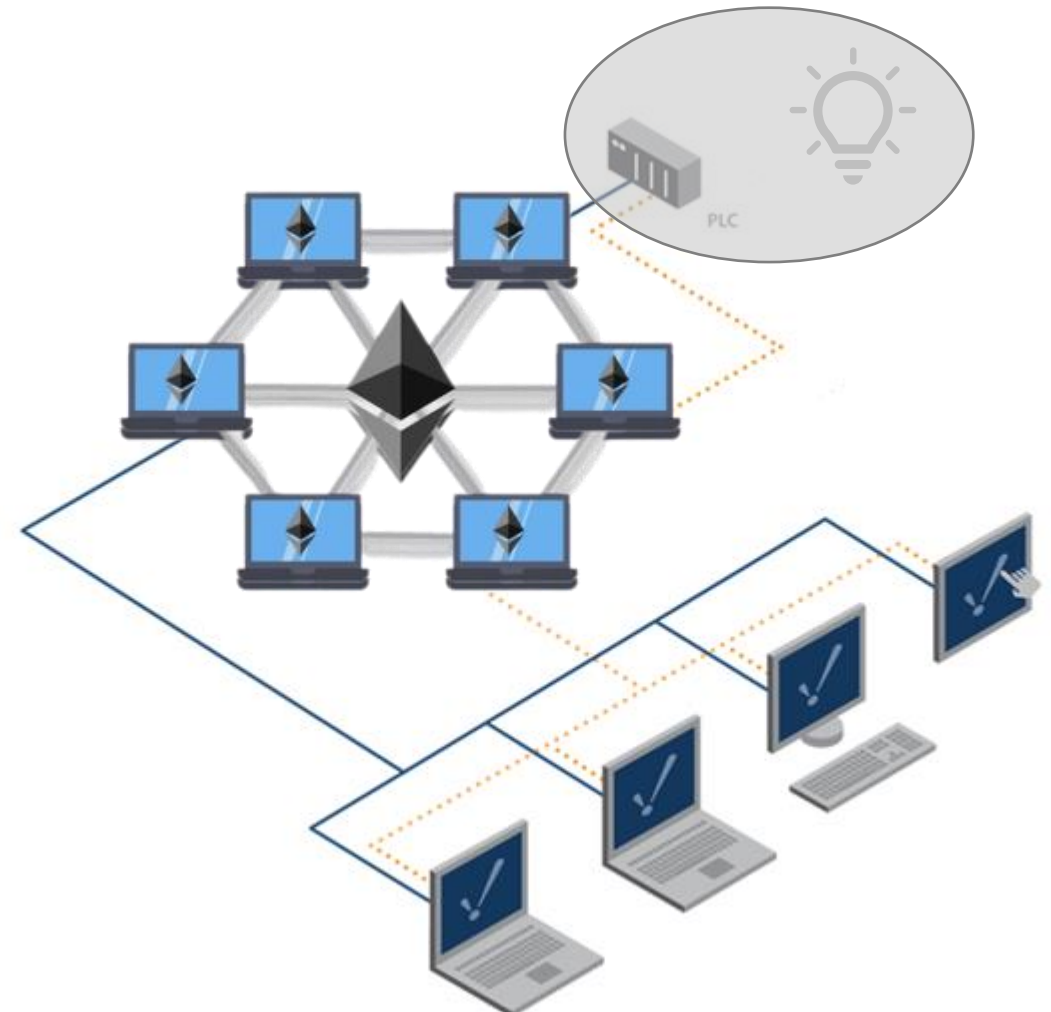
- Contiene Base de Datos. Posibles objetos:
 - Objeto *Entrada Binaria*: Valor $\in(0,1)$; Quality $\in(0...10)$; timestamp $\in(\text{int})$
 - Objeto *Comando binario*: Valor $\in(0,1)$; Quality $\in(0...10)$; timestamp $\in(\text{int})$
 - Objeto entrada *Analógica*: Valor $\in(\text{float32})$; Quality $\in(0...10)$; timestamp $\in(\text{int})$
- Genera eventos (Librería PUSH?)



SCADA Descentralizado – Etapa 1

Simulador de equipo servidor de datos (esclavo)

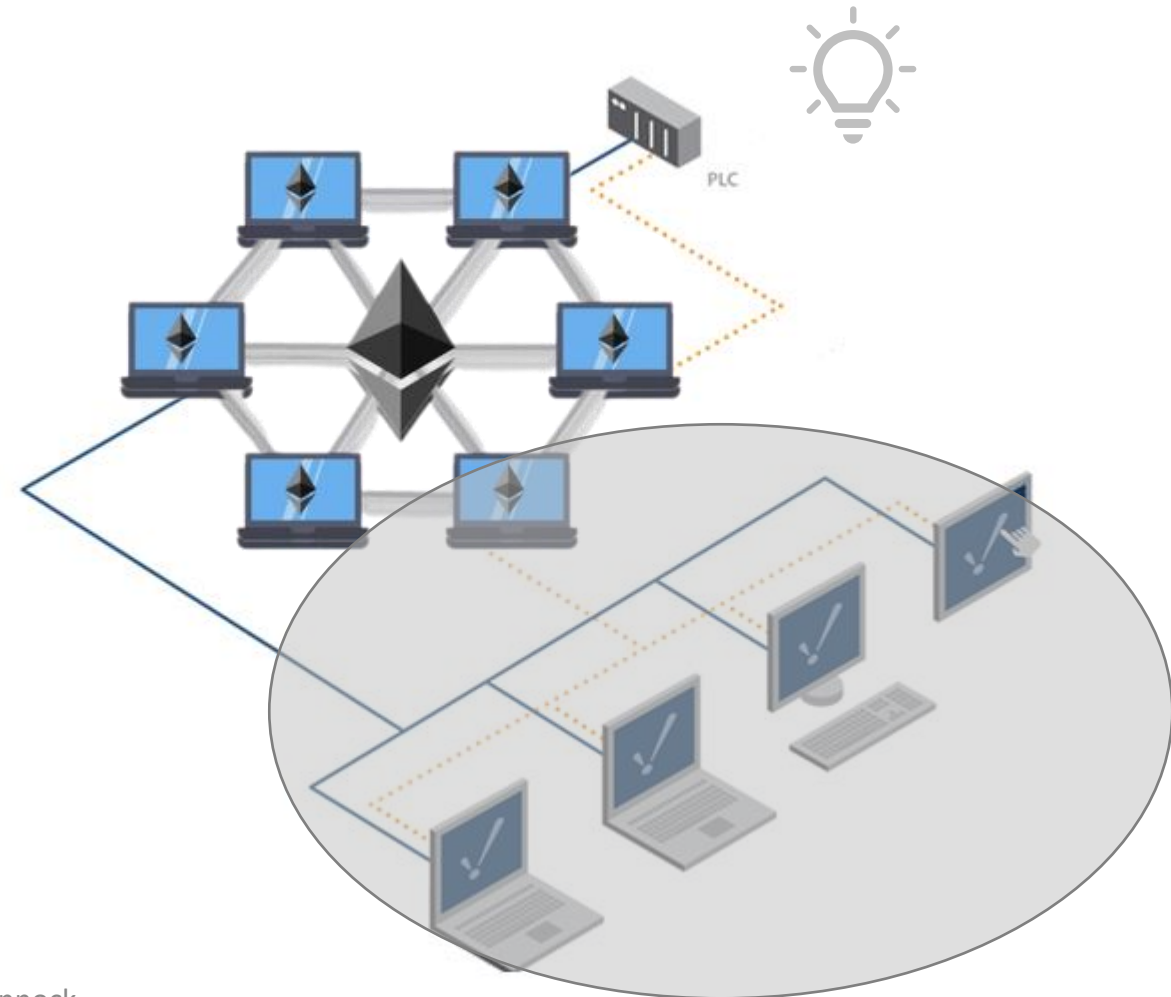
- Es una página web que nos permite simular el estado de un objeto.
- Permite forzar un valor que se escribe en la blockchain en storage o como un evento
- Permite recibir un comando y en consecuencia simular que cambia su estado
- El identificador de un esclavo sería su dirección pública



SCADA Descentralizado – Etapa 1

HMI – Interfaz gráfica




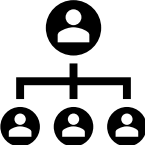
- Es una página web que nos permite ver el estado del objeto remoto leyendo variables en storage de la blockchain (o leyendo el ultimo evento?)
- Permite enviar un commando al equipo remote para cambiar su estado
- Permite ver valores analógicos
- Lleva un registro de eventos que hayan ocurrido (Lo hace a partir de los eventos de Ethereum? O lo hace reconstruyendo el historial al ir buscando las cadenas de bloques pasadas?)

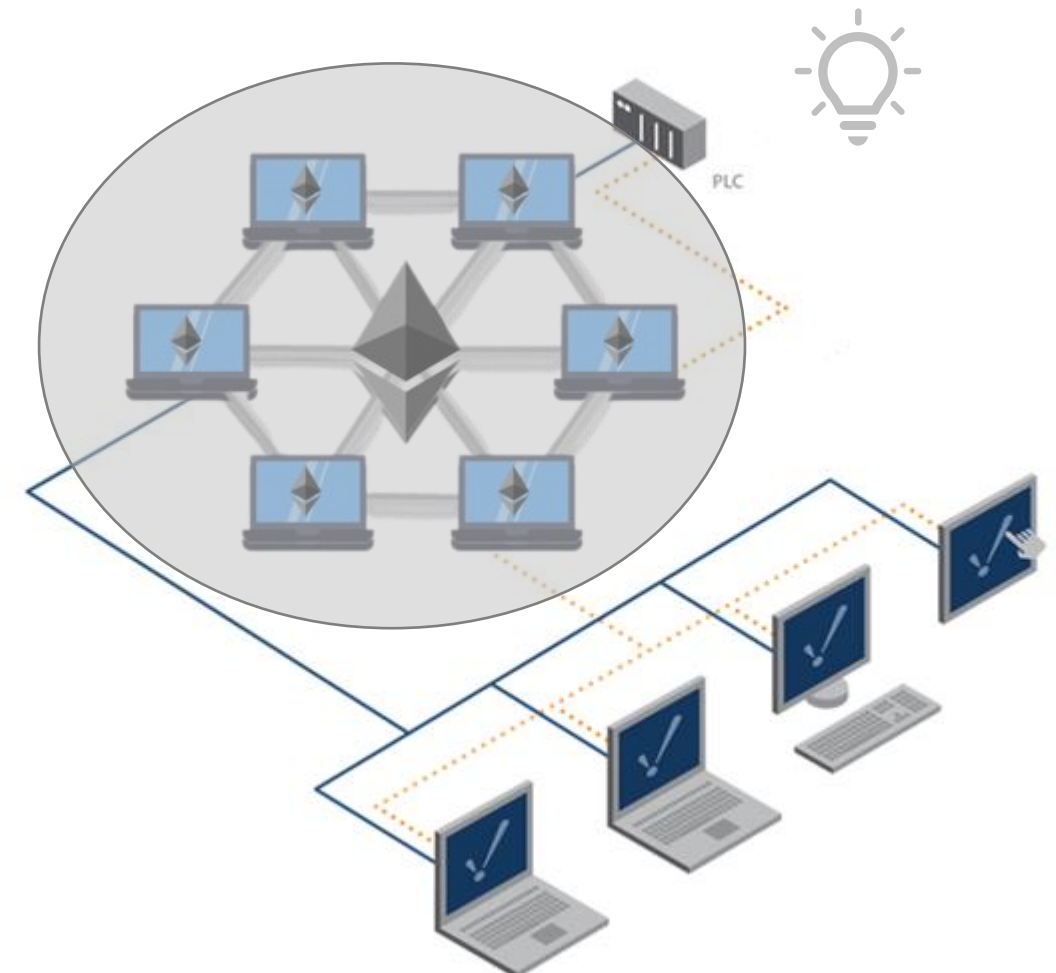


SCADA Descentralizado – Etapa 2

Smart Contract – Permisos y usuarios

- Contiene usuarios y maneja permisos (Librería openzeppelin?)

	Visualización: Solo puede ver. Todos pueden ver.
	Operación: Puede enviar comandos. Su dir. pública estará guardada en el SC.
	Ingeniería: Puede agregar-eliminar objetos-esclavos en el SC. Su dir. pública estará guardada en el SC.
	Root: Puede agregar-eliminar-modificar usuarios y permisos. Su dir. pública estará guardada en el SC.



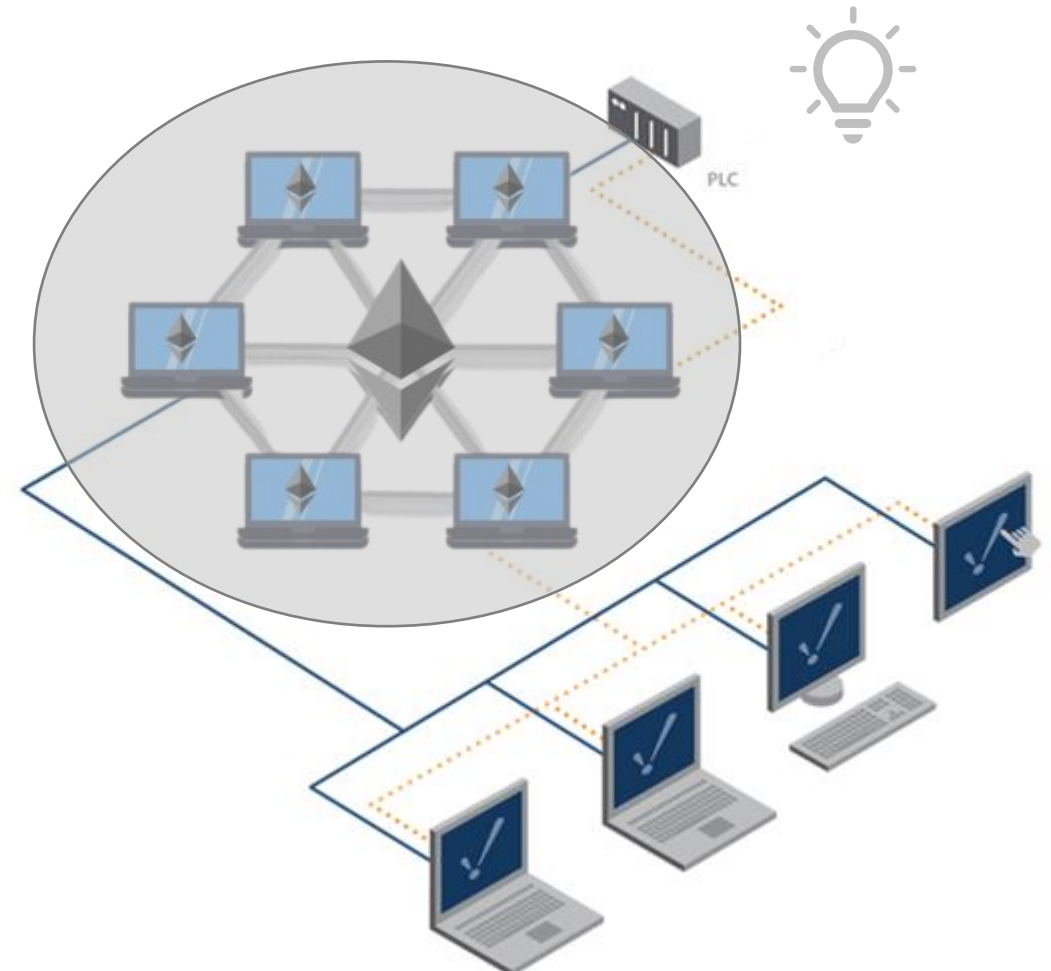
SCADA Descentralizado – Etapa 2

Smart Contract y Esclavos

- En el smart contract también están los esclavos, que son los equipos que reportan las señales a la blockchain. En el caso de la imagen sería el PLC.

Un usuario con permiso de **Ingeniería** puede agregar-eliminar esclavos y objetos en la blockchain.

La BD está en memoria. Cada objeto debería estar asociado a un esclavo, y el SC debería asegurarse que solo ese esclavo modifique a ese objeto.



SCADA Descentralizado – Etapa 2

Ejemplo simple de BD en memoria

dir esclavo	id señal	valor	calidad	estampa de tiempo
0x....AA	0	1	0	1682142714
0x....AA	1	123.45	1	1682184950
0x....BB		1	0	1682142923
0X00...00		0	0	1682147483

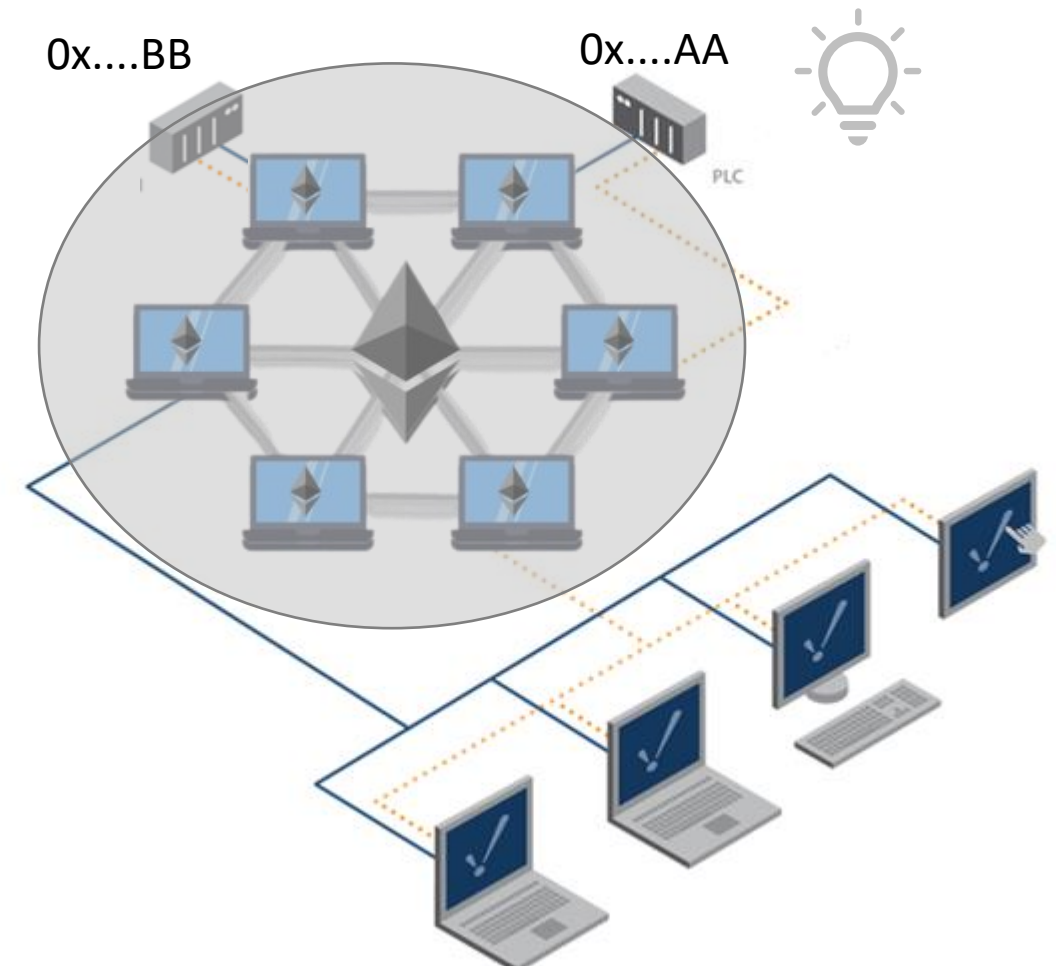
Ejemplo: El equipo con dirección 0x...AA reporta una señal binaria y una señal analógica. La señal binaria tiene un ID=0 y la analógica un id=1.

El SC se debe asegurar que solo esa dirección puede escribir valores ahí.

El esclavo indica con su dirección quién es, y qué valores quiere escribir para cada id de señal.

4/25/2023

Damián Ariel Minnock

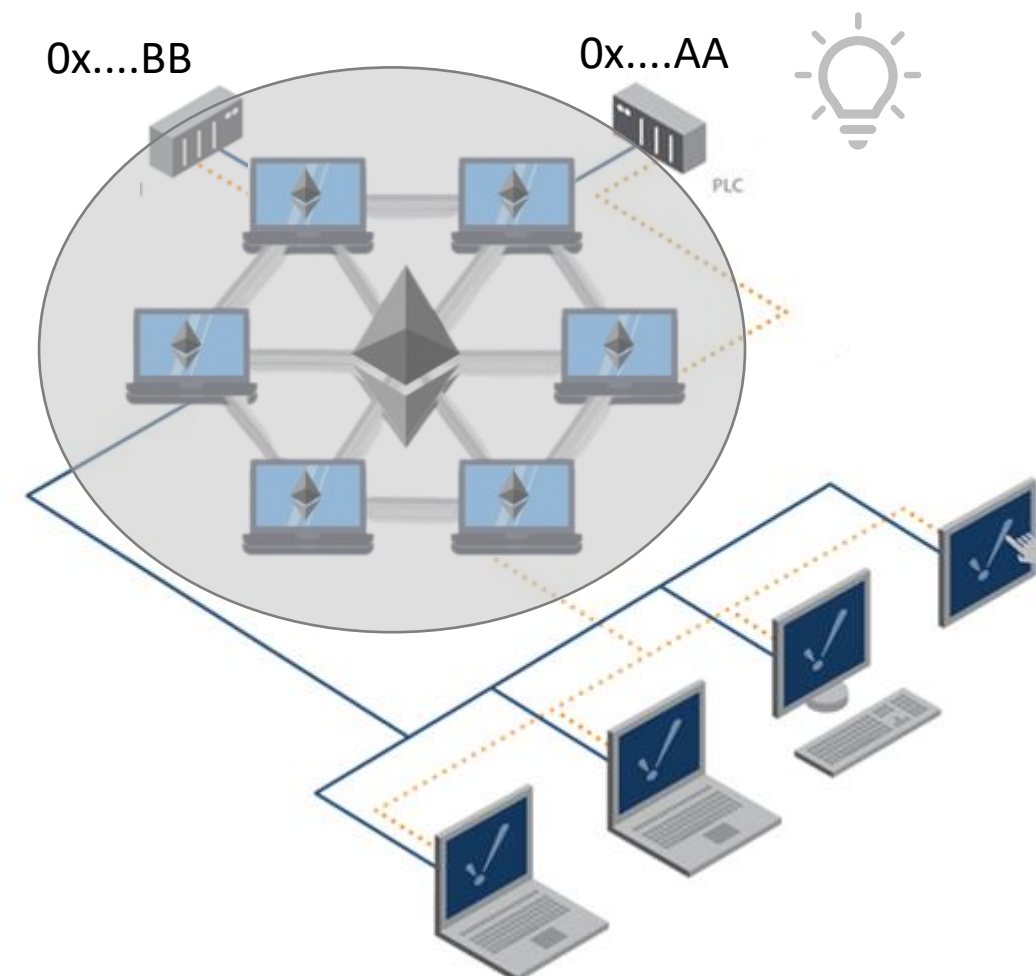


SCADA Descentralizado – Etapa 2

Ejemplo simple de BD en memoria

dir esclavo	id señal	valor	calidad	estampa de tiempo
0x....AA	0	1	0	1682142714
0x....AA	1	123.45	1	1682184950
0x....BB	0	1	0	1682142923
0X00...00	0	0	0	1682147483

Ejemplo: El equipo con dirección 0x...BB solo reporta una señal binaria. La señal binaria tiene un ID=0. El SC se debe asegurar que solo esa dirección puede escribir valores ahí.



SCADA Descentralizado – Etapa 2

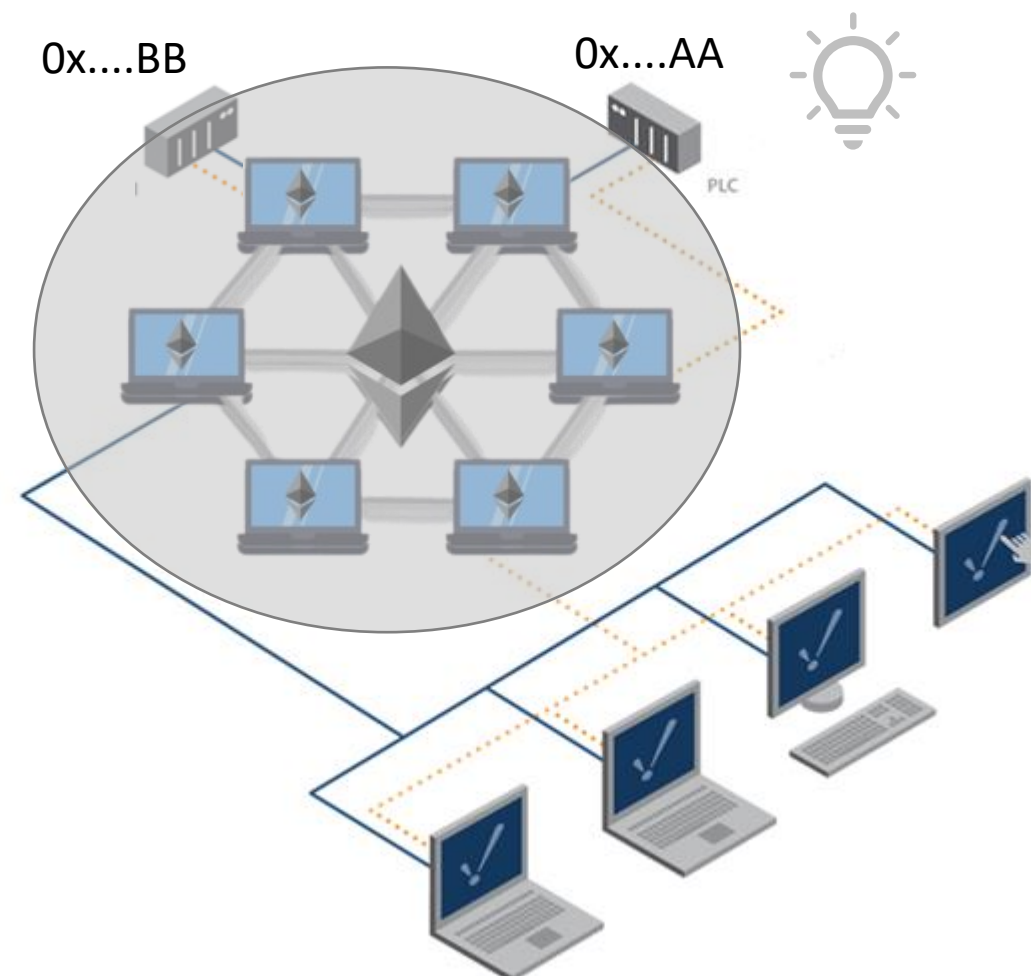
Ejemplo simple de BD en memoria

dir esclavo	id señal	valor	calidad	estampa de tiempo
0x....AA	0	1	0	1682142714
0x....AA	1	123.45	1	1682184950
0x....BB	0	1	0	1682142923
0X00...00	0	0	0	1682147483

Se deja una dirección = 0 reservada que significa que no pertenece a ninguna dirección en particular. Eso se puede utilizar por ejemplo para el caso de un commando que puede ser enviado desde multiples direcciones. El SC se asegurará que la dirección tenga permisos de operación.

4/25/2023

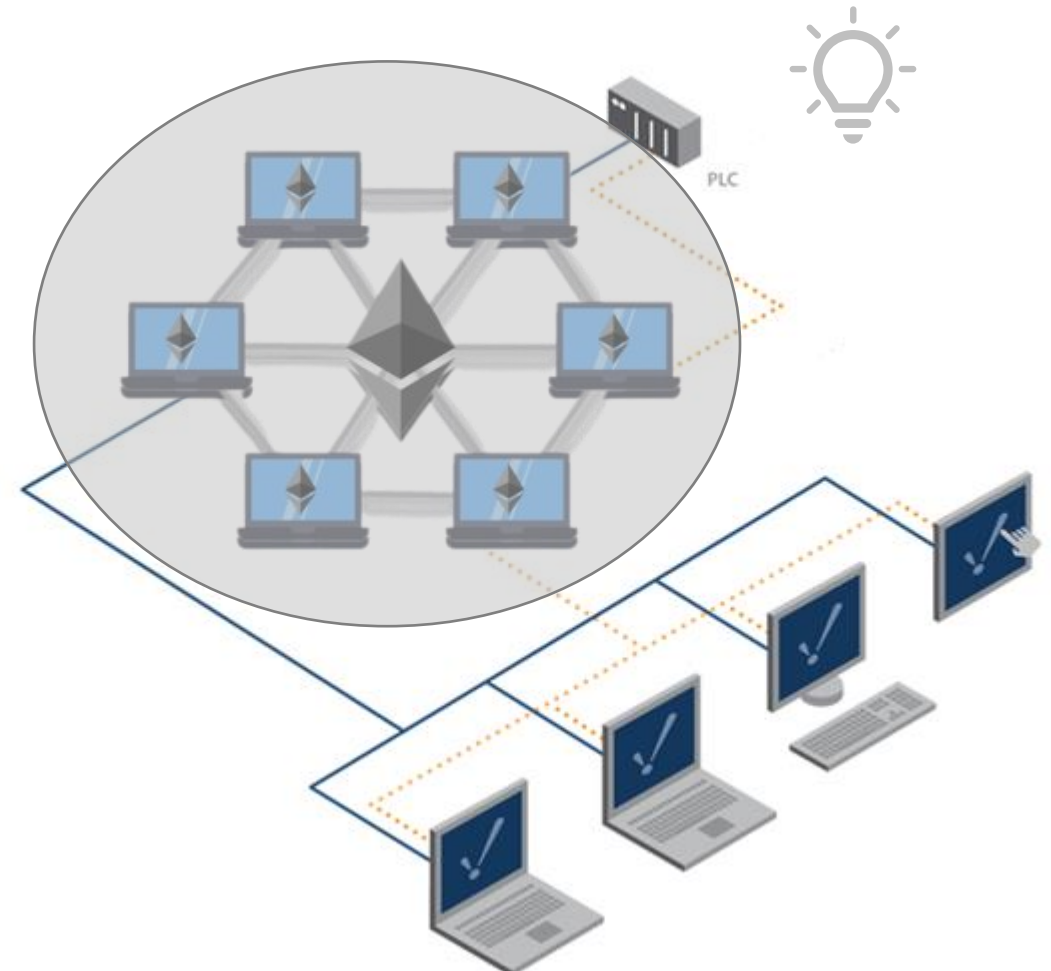
Damián Ariel Minnock



SCADA Descentralizado – Etapa 3

Smart Contract – Supervisión de equipos de comunicación

- El Smart Contract automáticamente podría “supervisar” al PLC. Eso significa que periódicamente verifica una conexión con el PLC. En caso de no tener respuesta, puede colocar en alarma en su base de datos.
- Los HMI recibirán ese estado y sabrán que perdieron conexión con ese equipo.
- La periodicidad de esta verificación puede ser modificada por un usuario de ingeniería



SCADA Descentralizado – Etapa 4

HMI – Interfaz gráfica y usuario de ingeniería

- Permite fácilmente realizar tareas de ingeniería para insertar objetos en la pantalla de operación/visualización y crear también los objetos en el SC
- Permite fácilmente que el root visualice y modifique permisos de usuarios

