UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE E TECNOLOGIE

Corso di Laurea in Informatica per la comunicazione digitale

Streaming decentralizzato di contenuti audiovisivi

Relatore: Trentini Andrea

Tesi di Laurea di:
Mirko Milovanovic
Matricola: 870671

Anno Accademico 2021-2022

Indice

1 Introduzione **1**
- 1.1 TCP/IP, Internet e World Wide Web . . . 2
- 1.2 Il difetto del Web . . . 3
  - 1.2.1 Web 2.0 . . . 3
  - 1.2.2 Controllo delle comunicazioni tra individui . . . 3
  - 1.2.3 Riappropriazione dei dati . . . 3
  - 1.2.4 Forte dipendenza da software proprietario e perdita di privacy 3
  - 1.2.5 Single point of failure . . . 3
- 1.3 Live streaming e interaction . . . 3
  - 1.3.1 Use cases . . . 4

2 Stato attuale del software **9**
- 2.1 Nomenclatura . . . 9
- 2.2 Stato dell'arte . . . 11
  - 2.2.1 Twitch . . . 11
  - 2.2.2 YouTube . . . 13
  - 2.2.3 Tahoe-LAFS . . . 15
  - 2.2.4 IPFS . . . 17
  - 2.2.5 PeerTube . . . 19
  - 2.2.6 Ace Stream . . . 21
- 2.3 . . . 23
- 2.4 Decentralizzazione e federazione . . . 23

3 La piattaforma ideale **25**
- 3.1 Soluzione ideale . . . 25
- 3.2 Realizzazioni e implementazioni possibili ad oggi . . . 26
  - 3.2.1 Possibili problemi nella realizzazione . . . 26
    - 3.2.1.1 Brevetti, copyright, DRM, moderazione dei contenuti e degli utenti . . . 26
    - 3.2.1.2 Necessità di competizione . . . 26
- 3.3 Varie sottosezioni per tutti i modi in cui si potrebbe realizzare ora . 26

3.3.1 Protocolli/Software Stack: P2P, Matrix, Jitsi, Crowd CDN, standalone apps, etc 26
3.4 Uno sguardo al futuro 26

4 Conclusioni **27**

Bibliografia **31**

Capitolo 1

Introduzione

Fin dai suoi albori Internet, nato da ARPAnet, svolge un ruolo importantissimo nella vita di tutti noi, che sia condividere informazioni militari, utilizzare servizi erogati via Web, o più recentemente guardare contenuti audiovisivi "on-the-go", ovvero quello di connettere le persone intorno al globo nei più svariati modi possibili attraverso l'utilizzo di calcolatori come computer e/o dispositivi mobili "connessi". Negli anni e per come è stato concepito Internet stesso architetturalmente, si sono venuti a formare veri metodi di comunicazione tra gli utilizzatori e gli erogatori di servizi Internet, più o meno sicuri, efficaci o resilienti rispetto a tematiche come la privacy dei dati, la centralizzazione e costi di operazione.

atrent: attenzione all'italiano, ho corretto

Quello più diffuso oramai è certamente il modello Client/Server, reso famoso dall'imponente World Wide Web il me più diffuso e oseremo azzardare a dire più intuitivo di utilizzare Internet ma certamente non l'unico.[24]

Questa necessità di avere grandi computer con elevata potenza di calcolo in un unico posto ha creato negli anni una situazione di forte centralizzazione da parte di poche "Big Tech", dovuto principalmente al grande costo necessario per allestire il software e tenere in funzione server indipendenti rispetto soluzioni di tipo *Infrastructure as a Service (IaaS)* o *Platform as a service (PaaS)*.

Questo metodo di *deployment* dei servizi ha creato una situazione in cui pochi gestori hanno un grosso controllo e potere rispetto a un enorme userbase creando una situazione "di troppo potere nelle mani di pochi"[15] che sfocia in un elevato controllo di tutte le comunicazioni effettuate attraverso queste piattaforme con tutti i vari problemi morali e legali annessi come la raccolta eccessiva di metadati sugli utenti, licenze di utilizzo super restrittive, controllo dei contenuti in base a pregiudizi Individuali, crittografia illusoria e rottura della reciproca fiducia, meccanismo alla base di Internet stessa.[21]

Lo scopo di questo elaborato è quello di analizzare lo stato attuale della rete e del software riguardo la fruizione e trasmissione di contenuti audiovisivi online in modo decentralizzato. Nello specifico ci interessa testare sul campo e valutare

mirko: sarebbe da rivedere questa parte per rendere ben chiara la situazione attuale della rete coi suoi problemi di forte centralizzazione
atrent: concordo, prendi pure anche da altri testi, scartabella la biblio del nostro

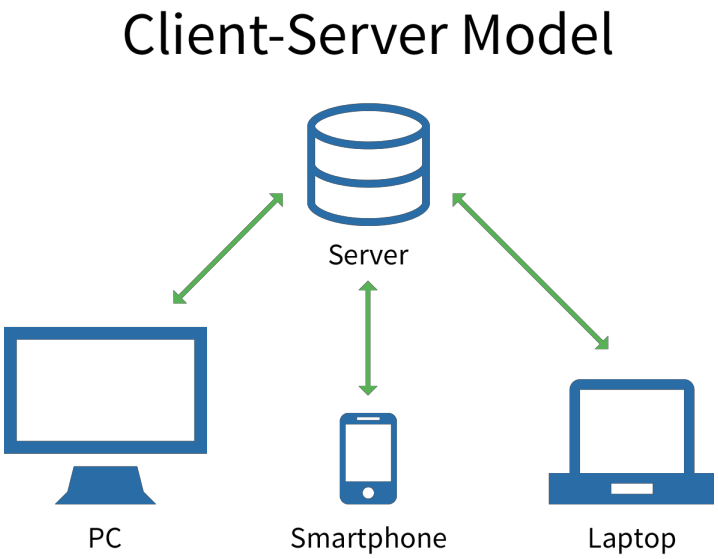
quali siano le soluzioni esistenti disponibili oggi, quali quelle più utilizzate, quale letteratura sia disponibile a riguardo, eventuale definizione e realizzazione di una piattaforma ideale soggettiva se nessuna di quelle trovate ci soddisfi e infine concludere con la presentazione dei del lavoro svolto.

atrent: così com'è l'intro non spiega bene il problema che ti vuoi porre

atrent: magari prima di continuare a scrivere testo dettaglia meglio l'outline inserendo sezioni e sottosezioni

mirko: il capitolo 2 non so come farlo così come impostato ora dato che questa parte viene già menzionata nella introduzione quando si menziona client/server atrent: potrebbe diventare una sezione del precedente

1.1 TCP/IP, Internet e World Wide Web



Un sistema Client/Server è un tipo di computazione distribuita in cui i clienti effettuano delle richieste verso un server che a sua volta risponde con i dati/servizi richiesti restando in attesa. I client possono essere di vario tipo e trovarsi ovunque nel globo e in generale integrano una parte hardware (smartphone, PCs, ...) ad una software (applicazioni GUI, web browser, ...). I server, invece, sono delle macchine specializzate spesso raggruppate assieme in grandi data center, interconnesse tra di loro per fornire uno o più servizi a molteplici client contemporaneamente.

1.2 Il difetto del Web

1.2.1 Web 2.0

1.2.2 Controllo delle comunicazioni tra individui

1.2.3 Riappropriazione dei dati

1.2.4 Forte dipendenza da software proprietario e perdita di privacy

1.2.5 Single point of failure

1.3 Live streaming e interaction

1.3.1 Use cases

Vediamo degli esempi di come questa interazione potrebbe essere svolta:

Esempio d'interazione "one to many"	
Use case:	Un utente davanti al proprio computer vorrebbe condividere quello che vede sullo schermo con i propri amici o followers
Soggetti:	Utente principale, viewers, computer, connessione internet
Obiettivi:	Condivisione in live streaming di un contenuto a schermo via internet con TCP/IP

Use case 1.1

Esempio d'interazione "many to many"	
Use case:	Più utenti al proprio computer vorrebbero comunicare e interagire tra di loro contemporaneamente simil conference-call
Soggetti:	Utenti multipli, computer, connessione internet, microfono, telecamera
Obiettivi:	Live streaming e interazione real-time tra utenti via internet con TCP/IP

Use case 1.2

Esempio d'interazione "one to many"	
Use case:	Un'azienda per questioni legate alla sicurezza sul lavoro si ritrova con la necessitò di dover fare dei "workshop" in diretta ai propri dipendenti con diverse locazioni sparse per il nel mondo senza utilizzare però grandi servizi cloud dato l'elevato costo di banda e di noleggio del servizio per singolo utente finale, in questo caso il singolo dipendente
Soggetti:	Dipendenti, azienda
Obiettivi:	Video streaming dei "workshop" per i dipendenti sparsi per il mondo

Use case 1.3

Esempio d'interazione "one to many"	
Use case:	Un gruppo di amici deve svolgere un progetto universitario assieme e quindi interagire tra di loro facendo pair programming condividendo lo schermo gli uni con gli altri. Per questioni di privacy e sicurezza non vuole utilizzare un servizio pubblico come Discord in quanto vorrebbero tenere tutto segreto fino al giorno della presentazione
Soggetti:	Studneti
Obiettivi:	Pair programming

Use case 1.4

Esempio d'interazione "one to many"	
Use case:	Una casa produttrice di film emergente vuole condividere i nuovi film in produzione con dei trailer ma a causa di dispute legate ai DRM, copyright e content strike di altre aziende più grandi non vuole utilizzare dei servizi già esistenti con EULA molto restringenti ma vuole avere il controllo dei propri diritti sul contenuto creato da essa stessa
Soggetti:	Filmmakers, appassionati di film
Obiettivi:	Condivisione degli ultimi trailer per i film prodotti dalla casa

Use case 1.5

In tutti i casi elencati sopra (ma anche in molti altri) emerge dalla nostra intuizione la necessità per un attore di eseguire uno streaming uno a molti o molti a molti senza utilizzare un servizio "cloud" già esistente proprietario ma utilizzando un software con licenza libera, senza dover fare il deploy di un intera infrastruttura HW globale, soddisfare le elevate richieste di banda richieste dalla natura dei contenuti video.

Azienda:

- Live stream di un evento/riunione interna
- Assistenza remota
- Conference calls
- come utente o come provider (distributore)
- Riduzione dei costi/banda necessaria

Education:

- Live courses

Common Joe:

- Lan party
- Gaming/Streamer

- Conference calls
- Security cameras
- Content moderation
- Small friends group live sharing

Governo:

- Comunicazioni video di emergenza
- Click day like overload avoidance

Common Lizard:
Metaverso, qualsiasi cosa sarà

Capitolo 2

Stato attuale del software

In questo capitolo andremo ad esplorare ed analizzare tutte le situazioni e le tecnologie attualmente esistenti che possono essere applicate

---

2.1 Nomenclatura

Prima di iniziare con il "dive-in" vorremmo spendere qualche riga per fare un inquadratura generale di tutti i termini, non strettamente di uso comune, che andremo ad utilizzare nei capitoli successivi per cercare di essere il più possibile comprensibili anche ai meno avvezzi del settore

- Livelli di decentralizzazione

  Descrtizione livelli di decentralizzazione

- Licenza

  Per licenza si intende un tipo di contratto con il quale il titolare dei diritti di sfruttamento economico sul software (programma informatico) definisce il regime giuridico di circolazione e le limitazioni nell'utilizzo e nella cessione dell'opera (che sia un'opera creativa, o un software, inteso come programma) [27] che l'utente di un servizio deve accettare per utilizzare il suddetto.

- Network Address Translation (NAT)

Il 'Network Address Translation' o anche detto NAT è un meccanismo usato tipicamente nelle reti locali che permette di a molteplici host/client di condividere uno solo indirizzo IP pubblico accessibile dall'esterno. Un dispositivo sulla rete si occupa di modificare e instradare i pacchetti IP da una rete all'altra facendo un 'mapping' delle porte pubbliche a quelle private e di instrdare i pacchetti IP sia in entrata che in uscita attraerso queste porte temporanee. [14]

- NAT Traversal

  Il problema del NAT Traversal sorge quando vari peer dietro NAT diversi cercano di comunicare. Siccome gli indirizzi IP reali vengono mascherati quando una applicazione, che non è stata pensata per il NAT traversal, prova a chiamare dei servizi esterni potrebbe inviare erroneamente il proprio indirizzo IP privato come indirizzo di risposta, cosa che impedirebbe il server esterno di rispondere e inviare dati verso il client.
  Un modo per risolvere questo problema è utilizzare il port forwarding. La tecnica più utilizzata per risolvere il problema del NAT traversal è il TCP hole punching. [14] [17] [13]

- Content delivery Network (CDN)

  Una Content Delivery Network (CDN) è un sistema distribuito di server composto da un grande numero di nodi che consente agli utenti di richiedere oggetti da nodi vicini. Le CDN vengono usate di solito per ridurre la latenza e i tempi di caricamento end-to-end dal lato utente e sono tipicamente forniti da servizi esterni di terze parti. Vengono usate tipicamente per alleggerire il carico dai server dei provider di contenuti e fornire resistenza contro gli attacchi di 'denial' of service distribuiti (DDoS). Tuttavia, però, memorizzando i dati dei provider e elaborando richieste degli utenti finali, i fornitori di CDN possono applicare tecniche di deuzione e tracking delle preferenze degli utenti che possono essere usate per scopi malevoli. Tale diulgazione di informazioni potrebbe comportare una perdita di privacy degli utenti e rivelare informazioni private specifiche del business o degli utenti a fornitori di CDN non affidabili o compromessi. [10]

- Distributed Hash Table (DHT)

  Una DHT [16]

2.2 Stato dell'arte

Per identificare i punti cardine di quello che vorremmo realizzare, abbiamo bisogno di studiare alcune soluzioni già esistenti sul mercato, con lo scopo di trarne vantaggi e difetti da cui partire per poter proporre una possibile soluzione alternativa a quelle già esistenti.

Per semplicità, abbiamo deciso di prendere in considerazione al massimo tre soluzioni software, già esistenti per ogni macro-categoria.

Questa lista include sia protocolli essenziali che software completi 'full-stack' e servizi web, per comparare la fattibilità di realizzazione di un software partendo dalle basi o utilizzando qualcosa che c'è già e, sono stati generalmente scelti, in base alla loro popolarità loro diffusione, licenza di utlizzo e utilità per il nostro caso d'uso.

2.2.1 Twitch

Twitch è una piattaforma di live streaming di proprietà di Amazon lanciata nel 2011, originariamente specializzata nella trasmissione in diretta di videogiochi, eSports ed eventi riguardanti il mondo videoludico.

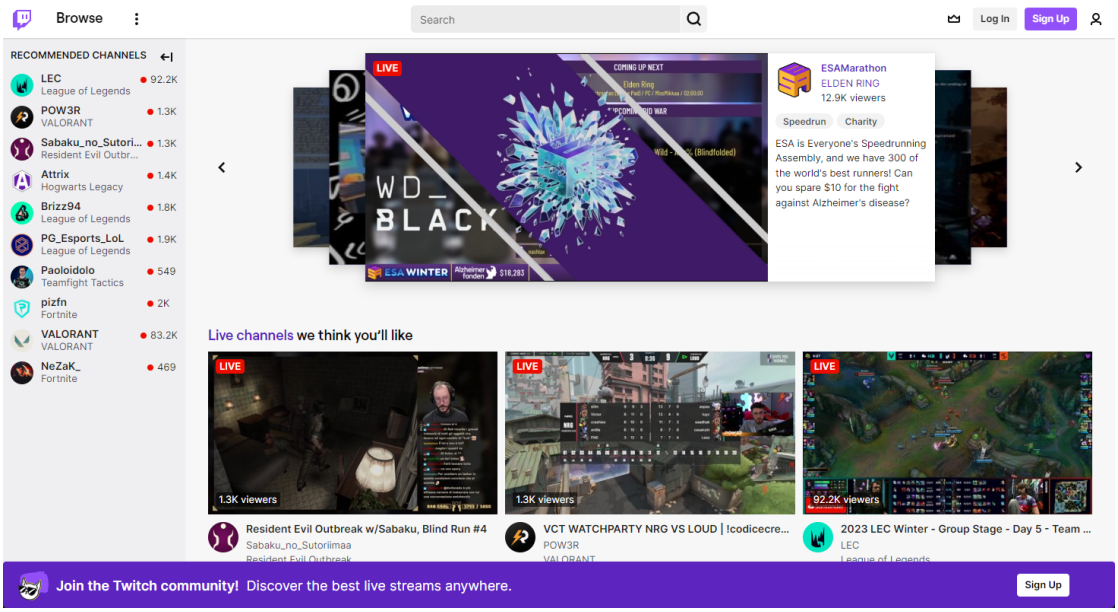


Figure 2.1. Home page di Twitch.

Una delle più famose e grandi, la piattaforma è stata originariamente sviluppata come una controparte di streaming per Justin.tv, una piattaforma di streaming

generale. Tuttavia, Twitch si è concentrata esclusivamente sui contenuti di videogiochi e di eSports, anche se oggi rimane tutt'ora la scelta "de-facto" per molti utenti non solo per lo streaming di contenuti videoludici ma anche per lo "streaming general purpose" come ad esempio lo streaming di musica, arte, cucina e di varie attività creative.

Negli anni successivi, ha visto una rapida crescita e ha attirato una vasta gamma di creatori di contenuti, dalle grandi organizzazioni di esports ai singoli streamer indipendenti, fino ad oggi con una user base di circa 3 milioni di viewers e 1.5 milioni di broadcaster giornalieri rispettivamente.

Twitch offre molti servizi sotto un unico sito web accessibile con un semplice browser:

Caratteristiche	
Partecipanti:	Chiunque con un account può condividere e guardare
Architettura software:	Centralizzata
Architettura hardware:	Servizio accessibile via browser o applicazione mobile
Tipo di dato condiviso:	Stream video e testo per funzionalità di live chat
Licenza:	Proprietaria con contratto di utilizzo 'EULA'

2.2.2 YouTube

Youtube è una piattaforma di sharing online di video statici, di proprietà di Google, specializzata nella condivisione globale di video generici con funzionalità aggiuntive di social media, monetizzazione e live streaming.

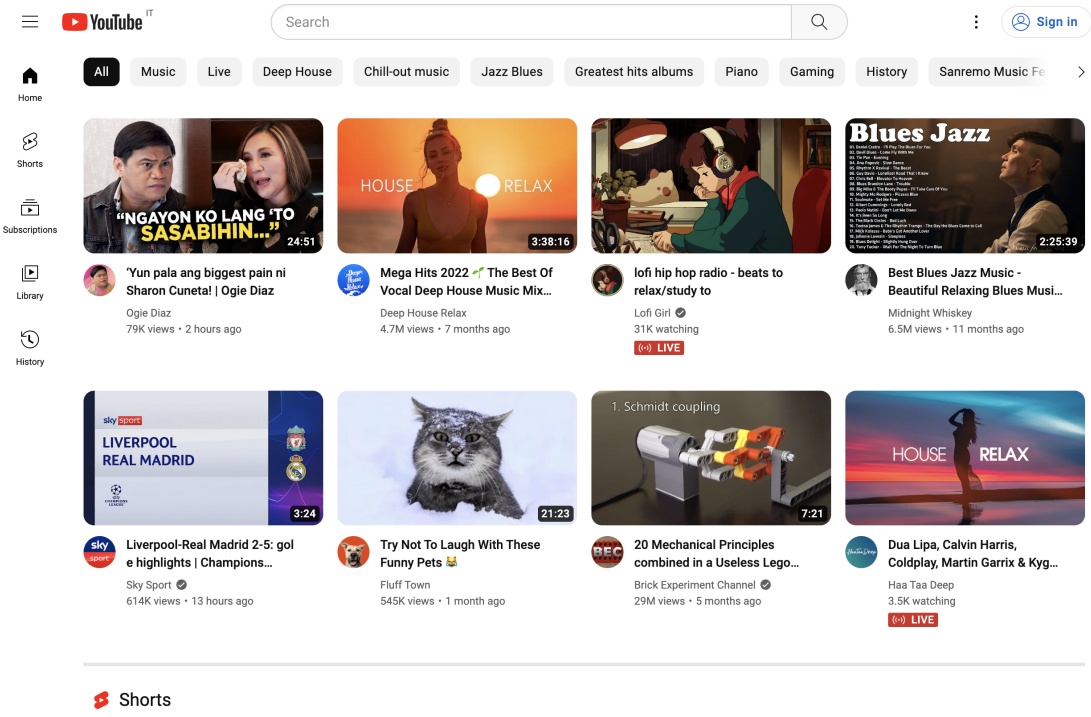


Figure 2.2. Home page di Youtube.

La compagnia è stata lanciata nel febbraio 2005 come indipendente ed è poi stata successivamente acquisita da Google. Al momento viene considerata la piattaforma più grande e longeva di questo segmento, con più di 2.5 miliardi di utenti mensili e milioni di ore di video condivisi ogni giorno. Dall'acquisizione, YouTube ha espanso la propria offerta al di fuori della sola condivisione di video "amatoriali", includendo contenuti come film di produzione professionale, video musicali ufficiali, documentari, news, ecc. Ha anche integrato la piattaforma pubblicitaria AdSense, sempre di proprietà di Google, permettendo a tutti gli utenti amatoriali o professionali approvati di poter ricevere un ricavo economico dalle pubblicità e

dal marketing presenti sul sito.

Caratteristiche	
Partecipanti:	Chiunque sul web
Architettura software:	Servizio accessibile via browser o applicazione mobile
Architettura hardware:	Centralizzata con utilizzo di CDN
Tipo di dato condiviso:	Video statici, live stream
Licenza:	Proprietaria con contratto di utilizzo 'EULA'

2.2.3 Tahoe-LAFS

Tahoe-LAFS è un sistema di file distribuito, open source, decentralizzato e sicuro che permette agli utenti di memorizzare e condividere file in un ambiente di rete distribuita chiamata 'Grid'. Sviluppato da un gruppo di ricercatori dell'Università di Maryland, dal 2007 ed è stato rilasciato come software open source sotto licenza GPL. Tahoe-LAFS è stato progettato per essere un sistema di file distribuito sicuro, affidabile e scalabile, in modo da poter essere utilizzato in ambienti di rete distribuita, dato che promette di essere un sistema 'provider-independent', ovvero che i fornitori dei server intermediari non hanno mai la possibilità di accedere o modificare i dati memorizzati dagli utenti finali perché che non sono loro a garantire la confidenzialità, l'integrità o l'assoluta disponibilità dei dati, ma sono i client finali a farlo.

Alla base, Tahoe-LAFS è essenzialmente un sistema di archiviazione chiave-valore. L'archivio utilizza delle corte stringhe, circa 100 byte, chiamate 'capabilities' come chiavi e dati arbitrari come valori.

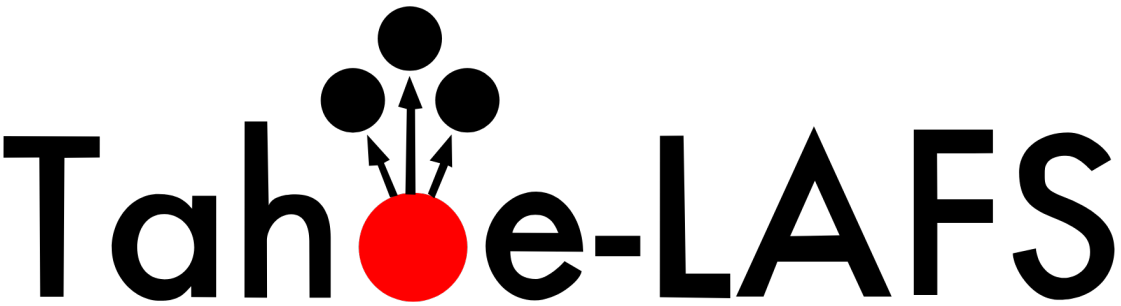


Figure 2.3. Tahoe-LAFS logo.

Si possono condividere queste 'capabilities' per dare agli altri accesso a determinati valori sulla 'Grid'. Ad esempio, si può dare la possibilità di lettura ad un amico e conservare il permesso di scrittura per te stessi. Per eliminare un valore, basta dimenticare (cioè eliminare) la chiave della 'capability', e, una volta fatto sarà impossibile recuperare i dati. I server di archiviazione però hanno un modo per fare 'garbage-collection' da condivisioni non referenziate.

In aggiunta al sistema chiave-valore, viene affiancato un livello i file-storage classico, che consente di condividere, con altri utenti, sotto-directory senza, ad esempio, rivelare l'esistenza o il contenuto delle directory principali.

Come detto prima, sono i clienti a garantire l'integrità e la confidenzialità dei dati, e questo viene realizzato grazie alla crittografia che viene eseguita su ogni 'capability' prima di essere caricata sul 'Grid'. Ogni valore viene prima crittografato con una chiave asimmetrica e poi spezzettato in parti più piccole, e più maneggiabili. Questi segmenti diventano poi effettivamente gli 'share' che verran-

no memorizzati nei noti della rete, che ricordiamo, svolgono solo la funzione di memorizzazione dei dati, gli utenti non si affidano a loro per altro. [2] [20]

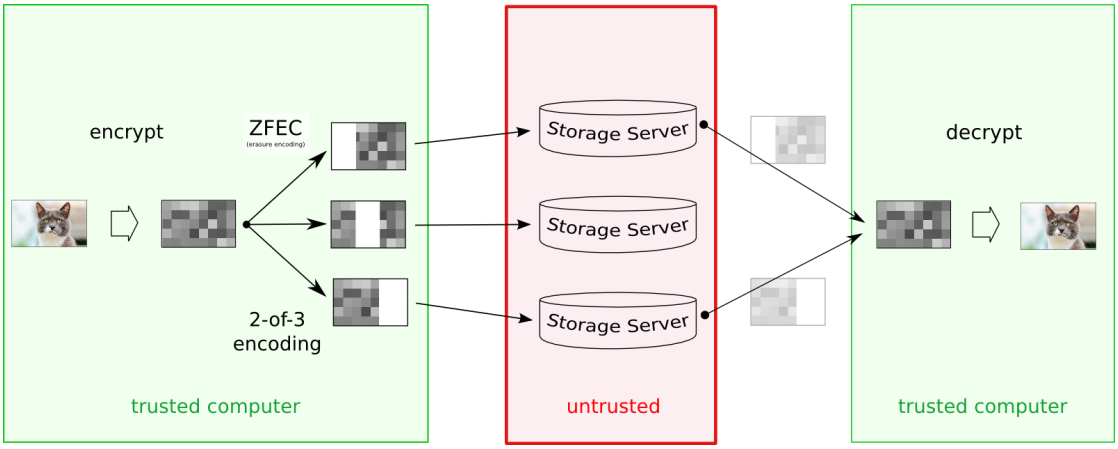


Figure 2.4. Tahoe-LAFS simple data flow.

Caratteristiche	
Partecipanti:	Utenti e server di terze parti
Architettura software:	Software client lato utente
Architettura hardware:	Decentralizzata e 'provider-indipendent'
Tipo di dato condiviso:	Qualsiasi tipo di file binario
Licenza:	GPL

2.2.4 IPFS

IPFS o 'InterPlanetary File System' è una suite di protocolli e librerie open source per la condivisione di file in un ambiente di rete distribuita utilizzando meccanismi di 'content-addressing'. IPFS è stato sviluppato da Protocol Labs, un'azienda di ricerca e sviluppo nel 2014.

IPFS punta a creare un file system condiviso da una rete di nodi decentralizzata che comunica attraverso P2P, dove i singoli file sono organizzati come blocchi indipendentemente individuabili ed immutabili (ovvero che non possono essere modificati ma solo aggiunti o eliminati) con degli identificatori chiamati 'CID (Content identifiers)' memorizzati in un database distribuito chiamato 'DHT (Distributed Hash Table)' che viene condiviso con ogni singolo nodo della rete al fine di facilitare il routing dei dati attraverso essa stessa.

Uno dei problemi di IPFS è che per design stesso, ogni singolo file è visibile a tutti i nodi della rete e quindi non è possibile creare un sistema di condivisione di file privati, ma solo di file pubblici, che però volendo possono essere crittografati, impedendo la lettura dei dati da parte di attori terzi. [25] [8]

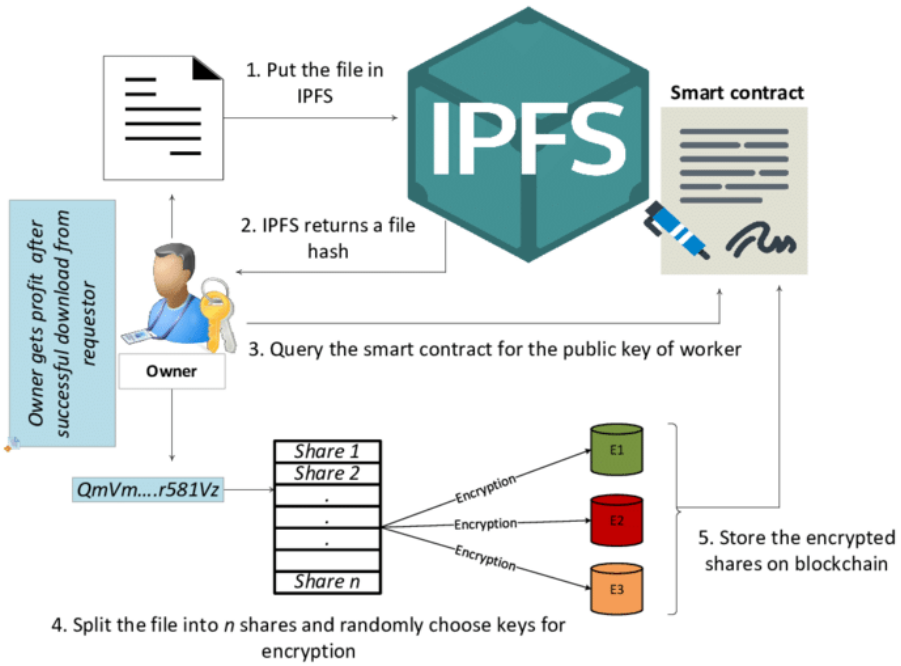


Figure 2.5. IPFS data flow. [18]

Caratteristiche	
Partecipanti:	Nodi della rete
Architettura software:	Protocolli e software lato client
Architettura hardware:	Decentralizzata simil Blockchain
Tipo di dato condiviso:	Qualsiasi tipo di file binario
Licenza:	MIT

2.2.5 PeerTube

PeerTube è un servizio di video sharing open source, federalizzato e decentralizzato (lato client), basato su protocolli peer-to-peer come 'WebTorrent', 'WebRTC' e altre tecnologie web standard. Fa parte del cosiddetto 'Fediverse', un insieme di server interconnessi che comunicano tramite ActivityPub, un protocollo di comunicazione aperto per creare reti federate. Inizialmente creato come una piattaforma di video sharing per i creatori di contenuti indipendenti, è stato progettato per essere estensibile e adattabile a qualsiasi tipo di contenuto video che rispetti i termini delle singole istanze che vengono messe a disposizione pubblicamente.

Il funzionamento è del tutto simile ad altre piattaforme video tipo (YouTube, Vimeo, Dailymotion, ecc.) con supporto per video statici e livestream, ma con la differenza che i video possono essere memorizzati e condivisi non solo dalle singole istanze della federazione attraverso il normale HTTP ma anche tra i client finali usando P2P per alleggerire il carico di banda. PeerTube è iniziato come progetto indipendente da un singolo sviluppatore che poi è stato affiancato dalla no-profit Framasoft. [12] [26] [4]

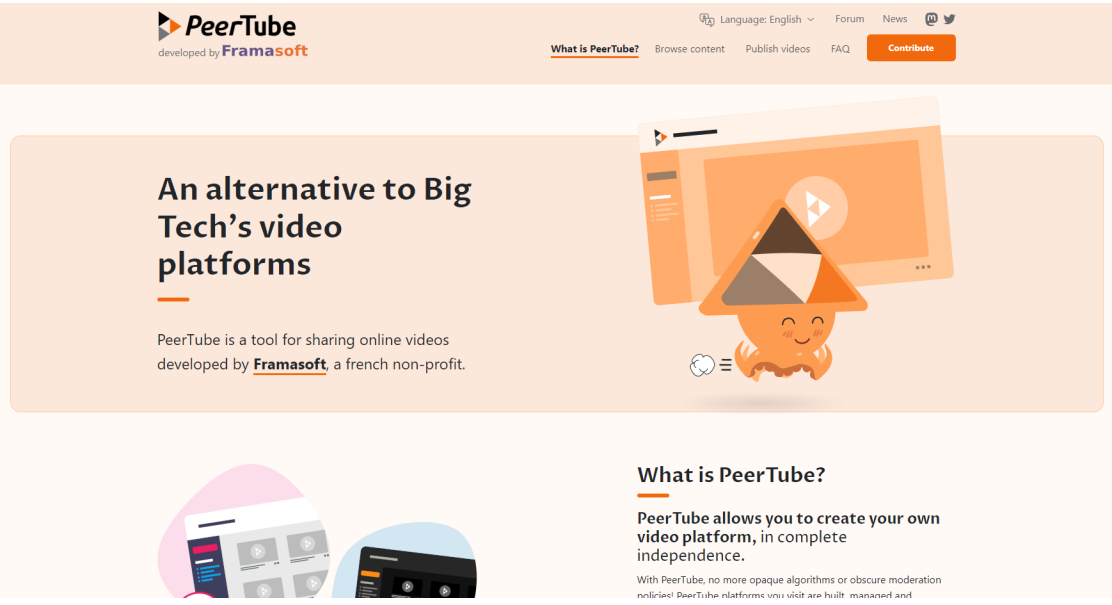


Figure 2.6. PeerTube.

Caratteristiche	
Partecipanti:	Utenti registrati e non delle singole istanze
Architettura software:	Browser web e server indipendenti
Architettura hardware:	Federalizzata e decentralizzata P2P
Tipo di dato condiviso:	Video statici e livestream
Licenza:	AGPLv3+

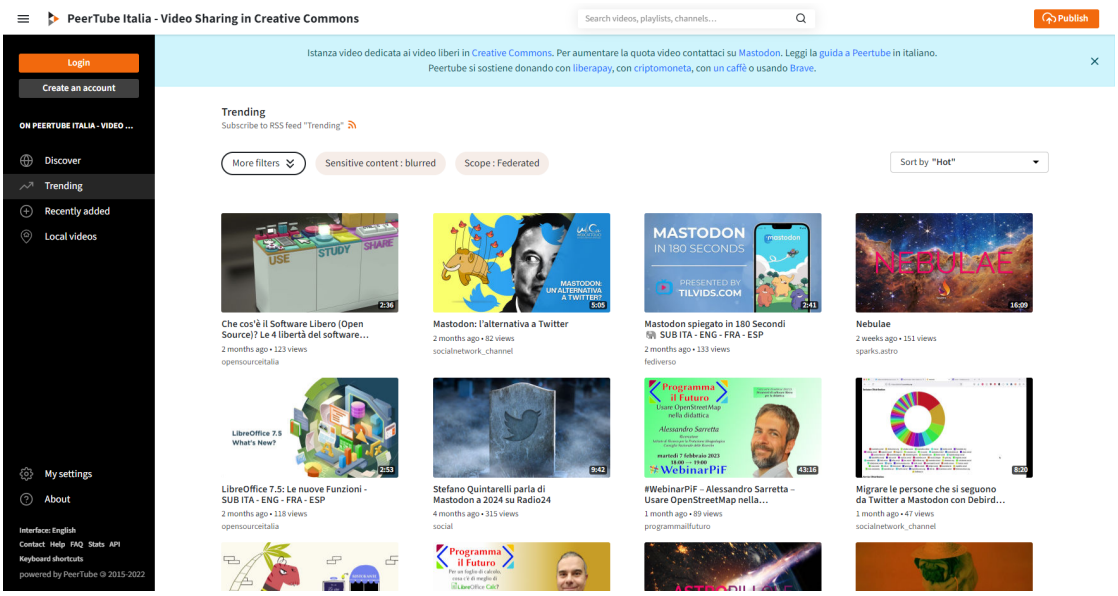


Figure 2.7. Una istanza di PeerTube Italiana.

2.2.6 Ace Stream

Ace Stream è un software proprietario di video streaming basato su P2P, sviluppato da un gruppo di sviluppatori russi, che permette agli utenti di trasmettere in diretta video e audio, riprodurre file multimediali, e ricevere e condividere contenuti in tempo reale. Il software è stato rilasciato per la prima volta nel 2013 e da allora è stato scaricato più di 100 milioni di volte. Ace Stream offre una vasta gamma di funzionalità che lo rendono popolare tra gli utenti che desiderano guardare contenuti in streaming. Utilizzando la tecnologia P2P, il software consente agli utenti di trasmettere video e audio in tempo reale, senza dover attendere il completamento del download. Questo significa che gli utenti possono godere di una riproduzione fluida e senza interruzioni, anche con connessioni internet più lente.

Figure 2.8. PeerTube.

Inoltre, Ace Stream supporta la riproduzione di file multimediali locali, consentendo agli utenti di guardare i propri film, serie TV e altri contenuti salvati sul proprio dispositivo. Il software supporta una vasta gamma di formati video e audio, garantendo un'alta compatibilità.

Un'altra caratteristica interessante di Ace Stream è la possibilità di ricevere e condividere contenuti in tempo reale. Gli utenti possono creare i propri canali di streaming e condividere i propri contenuti con altri utenti. Questo rende il software ideale per gli appassionati di sport, che possono trasmettere eventi sportivi in diretta e condividerli con altri appassionati.

Ace Stream è disponibile per diverse piattaforme, tra cui Windows, Mac, Linux e Android, il che lo rende accessibile a un vasto pubblico. Inoltre, il software è gratuito da scaricare e utilizzare, rendendolo ancora più attraente per gli utenti.

In conclusione, Ace Stream è un software potente e versatile per lo streaming di video e audio. Grazie alla sua tecnologia P2P, offre una riproduzione fluida e senza interruzioni, consentendo agli utenti di godersi i propri contenuti preferiti in tempo

reale. Con la possibilità di condividere contenuti e creare canali di streaming, Ace Stream offre un'esperienza di streaming interattiva e coinvolgente. [6] [3]

Caratteristiche	
Partecipanti:	Utenti registrati e non delle singole istanze
Architettura software:	Browser web e server indipendenti
Architettura hardware:	Federalizzata e decentralizzata P2P
Tipo di dato condiviso:	Video statici e livestream
Licenza:	AGPLv3+

2.3

2.4 Decentralizzazione e federazione

Capitolo 3

La piattaforma ideale

3.1 Soluzione ideale

Iniziare a spiegare come creare questa piattaforma ideale vs sistemi centralizzati esistenti

In base a quello che abbiamo esposto fin'ora sorge spontanea la domanda e di conseguenza la ricerca di una piattaforma decentralizzata che soddisfi alcuni requisiti essenziali o meno.
Possiamo azzardare a stilare l'elenco di requisiti e features che vorremmo fossero presenti in questa piattaforma decentralizzata:

1. Decentralizzazione totale della infrastruttura
2. Facilità nell'utilizzo
3. Basso costo di entrata
4. Interazione tra utenti in real time
5. Moderazione
6. Bassa latenza
7. Possibilità di monetizzazione del contenuto da parte dei singoli utenti
8. Altro? Si accettano proposte

Andiamo ad analizzare meglio la nostra proposta:

mirko: Sarebbe magari utile usare uno spettro come viene fatto nel libro di Cittadinanza digitale? atrent: per ora non ne vedo utilità, vediamo come si sviluppa prima l'outline

Punto 1: Quindi eliminazione del rischio di controllo da parte di un singolo, Resistenza agli attacchi DDOS, no single point of failure

Punto 2: Se la piattaforma è accessibile via Web o standalone app, o mobile app, se ci sono situazioni di relatività della rete con blocco di porte non standard e/o protocolli

Punto 3: È relazionato al punto 2 con la questione di facilità di accesso e utilizzo rispetto a piattaforme già esistenti PERO' nello specifico sarebbe più la parte di facilità di passaggio da una piattaforma all'altra da parte degli streamer e nella compatibilità dei software/strumenti

Punto 4: Banalmente come fa twitch e youtube live una chat pubblica tra utenti (quindi con autenticazione), in caso di conference audio/video condiviso tra più utenti in real time

Punto 5: moderazione degli utenti/streamer se ha senso farla

Punto 6: relazionato al punto 1, abbastanza ovvia come cosa

Punto 7: io la penso come incentivo da parte di una rete/piattaforma di attrarre nuove utenze

3.2 Realizzazioni e implementazioni possibili ad oggi

3.2.1 Possibili problemi nella realizzazione

3.2.1.1 Brevetti, copyright, DRM, moderazione dei contenuti e degli utenti

3.2.1.2 Necessità di competizione

3.3 Varie sottosezioni per tutti i modi in cui si potrebbe realizzare ora

3.3.1 Protocolli/Software Stack: P2P, Matrix, Jitsi, Crowd CDN, standalone apps, etc ...

3.4 Uno sguardo al futuro

Capitolo 4

Conclusioni

Bibliografia

[1] *"Cittadinanza digitale e tecnocivismo. In un mondo digitale la cittadinanza inizia dai bit di Andrea Trentini", Giovanni Biscuolo e Andrea Rossi*. it-IT. Dic. 2020. URL: https://www.letture.org/cittadinanza-digitale-e-tecnocivismo-andrea-trentini-giovanni-biscuolo-andrea-rossi.

[2] *About-tahoe; Tahoe-LAFS — tahoe-lafs.org*. https://www.tahoe-lafs.org/trac/tahoe-lafs/browser/trunk/docs/about-tahoe.rst. [Accessed 02-Apr-2023].

[3] *Ace Stream — web.archive.org*. [Accessed 28-09-2023]. URL: https://web.archive.org/web/20180618052904/http://info.acestream.org/#/about/acestream.

[4] *ActivityPub Rocks! — activitypub.rocks*. https://activitypub.rocks/. [Accessed 04-Apr-2023].

[5] Sawsan Ali et al. «What is Client-Server System: Architecture, Issues and Challenge of Client -Server System (Review)». In: (feb. 2020). DOI: 10.5281/zenodo.3673071.

[6] *Announcement! ACE Stream; New era of TV and Internet broadcasting — oldforum.acestream.media*. [Accessed 28-09-2023]. URL: http://oldforum.acestream.media/index.php?topic=1479.0.

[7] *ARPANET — Wikipedia, The Free Encyclopedia*. en. Page Version ID: 1105956998. Ago. 2022. URL: https://en.wikipedia.org/w/index.php?title=ARPANET&oldid=1105956998.

[8] Juan Benet. *IPFS - Content Addressed, Versioned, P2P File System*. 2014. DOI: 10.48550/ARXIV.1407.3561. URL: https://arxiv.org/abs/1407.3561.

[9] Shilpa Budhkar e Venkatesh Tamarapalli. «An overlay management strategy to improve QoS in CDN-P2P live streaming systems». In: *Peer-to-Peer Networking and Applications* 13.1 (apr. 2019), pp. 190–206. DOI: 10.1007/s12083-019-00755-x. URL: https://doi.org/10.1007/s12083-019-00755-x.

[10] Shujie Cui, Muhammad Rizwan Asghar e Giovanni Russello. «Multi-CDN: Towards Privacy in Content Delivery Networks». In: *IEEE Transactions on Dependable and Secure Computing* 17.5 (set. 2020), pp. 984–999. DOI: 10.1109/tdsc.2018.2833110. URL: https://doi.org/10.1109/tdsc.2018.2833110.

[11] Jie Deng et al. «Internet Scale User-Generated Live Video Streaming: The Twitch Case». In: feb. 2017, pp. 60–71. ISBN: 978-3-319-54327-7. DOI: 10.1007/978-3-319-54328-4_5.

[12] *FAQ — JoinPeerTube — joinpeertube.org*. [Accessed 04-Apr-2023]. URL: https://joinpeertube.org/faq#what-are-the-main-advantages-of-peertube.

[13] Bryan Ford, Dan Kegel e Pyda Srisuresh. *State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)*. RFC 5128. Mar. 2008. DOI: 10.17487/RFC5128. URL: https://www.rfc-editor.org/info/rfc5128.

[14] *Introduction to Network Address Translation (NAT) and NAT Traversal (2.12) — pjsip.org*. https://www.pjsip.org/pjnath/docs/html/group__nat__intro.htm. [Accessed 07-Mar-2023].

[15] Federica Maria Rita Livelli. *AI e big data, troppo potere nelle mani di pochi: il dibattito - AI4Business — ai4business.it*. https://www.ai4business.it/intelligenza-artificiale/ai-e-big-data-favoriscono-la-concentrazione-del-potere/. [Accessed 01-Sep-2022].

[16] Eng Keong Lua et al. «A survey and comparison of peer-to-peer overlay network schemes». In: *IEEE Communications Surveys & Tutorials* 7.2 (2005), pp. 72–93. DOI: 10.1109/comst.2005.1610546. URL: https://doi.org/10.1109/comst.2005.1610546.

[17] *NAT traversal - Wikipedia — en.wikipedia.org*. https://en.wikipedia.org/wiki/NAT_traversal. [Accessed 08-Mar-2023].

[18] Muqaddas Naz, Nadeem Javaid e Sohail Iqbal. «Research Based Data Rights Management Using Blockchain Over Ethereum Network». Tesi di dott. Set. 2019.

[19] M. Parameswaran, A. Susarla e A.B. Whinston. «P2P networking: an information sharing alternative». In: *Computer* 34.7 (lug. 2001), pp. 31–38. DOI: 10.1109/2.933501. URL: https://doi.org/10.1109/2.933501.

[20] *Tor at the Heart: Tahoe-LAFS — Tor Project — blog.torproject.org*. https://blog.torproject.org/tor-heart-tahoe-lafs/. [Accessed 02-Apr-2023].

[21] A. Trentini, G. Biscuolo e A. Rossi. *Cittadinanza digitale e tecnocivismo. In un mondo digitale la cittadinanza inizia dai bit*. Copy-left Italia. Ledizioni, 2020. ISBN: 9788855261609. URL: https://books.google.it/books?id=hHgczgEACAAJ.

[22] *Twitch (service) - Wikipedia — en.wikipedia.org*. https://en.wikipedia.org/wiki/Twitch_(service). [Accessed 09-Feb-2023].

[23] *TwitchTracker — twitchtracker.com*. https://twitchtracker.com/statistics/active-streamers. [Accessed 11-Feb-2023].

[24] W3C. *What is the difference between the Web and the Internet?* https://www.w3.org/Help/. [Accessed 31-Aug-2022]. 2021.

[25] *What is IPFS? — IPFS Docs — docs.ipfs.tech*. https://docs.ipfs.tech/concepts/what-is-ipfs/. [Accessed 04-Apr-2023].

[26] Wikipedia. *ActivityPub — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=ActivityPub&oldid=1134432238. [Online; accessed 05-April-2023]. 2023.

[27] Wikipedia. *Licenza (informatica) - Wikipedia — it.wikipedia.org*. https://it.wikipedia.org/wiki/Licenza_(informatica). [Accessed 25-Jan-2023]. 2022.