A peer-to-peer solution to standardise business-to-business data transactions

Description

Thomas Barratt

Friday 11th October, 2019

1 Background

Peer-to-peer networks allow for data transmission directly to another peer (node), without being transferred through a central server. This allows the network to be resilient against any node failures, and also prevents central servers from caching any transactions. Research suggests that peer-to-peer networks are responsible for 40 - 70% of all internet traffic (Schulze, 2011). Peer-to-peer networks have been used in many bandwidth intensive applications, such as Skype and Spotify (Pejchinovski, 2014), (Baset & Schulzrinne, 2004).

As the internet enabled world continues to grow, there are increasing concerns regarding privacy and how sensitive data is handled. A recent survey of 1000 people, completed by an SMS marketing company who were looking into the damage of the Facebook-Cambridge Analytica scandal, suggested *"94.1% of consumers surveyed are unlikely to do business with a company if they have concerns about their privacy practices"* (*One Year After Cambridge Analytica, Survey Reveals Strong Consumer Privacy Fears Remain*, 2019). This, combined with the necessary responsibility (and concerns) businesses have about personal data [(Lwin, Wirtz, & Williams, 2007) (Kurucz, Colbert, & Wheeler, 2008) (Cavoukian et al., 2009)] suggest that there is a place for a more robust and privacy oriented alternative to current practices.

Peer-to-peer networks allow access permissions to be controlled entirely by the original vendor. By constructing constraints, it is possible to limit the exposure of sensitive data to only those peers that need to view it (Åslund, 2002). Through cryptographic methods and segmentation (piecing), it is possible to share frequently accessed files across the networks while still only giving access to those peers with appropriate permissions. The act of piecing, encryption techniques, and piece selection are well researched topics (Liao, Papadopoulos, & Psounis, 2006), (Johnsen, 2005), (Åslund, 2002).

Although notable implementations often rely on some central server to manage sensitive issues like authentication (Johnsen, 2005), methods do exist which allow for total reliance on peers. Distributed Hash tables are an implemented technique to manage node lookups, ensuring nodes can find each other as necessary (Stoica, Morris, Karger, Kaashoek, & Balakrishnan, 2001).

Over the last ten years, there has been a notable correlation between interest in the term 'business-to-business', and 'peer-to-peer' (Ruiz Forns, 2019). This interest combined with the privacy capabilities of a peer-to-peer network show real potential for a business-oriented data transmission software, allowing businesses to only share sensitive data with those who need it.

This project seeks to provide a software experience which can manage a business' presence

on a custom peer-to-peer network. This software would allow businesses to share a custom set of files and data to each of the peers using the software. By alleviating intermediary servers, businesses will be able to maintain better control of their data, and one central repository that they can share subsets of as needed. Increasing the control and spread of data would allow for businesses' to have a better understanding of their data needs, and better management solutions in regards to access of sensitive materials.

2 Analysis

It is beneficial to divide the project into four sections: gathering data to transmit, the data transmission, peer lookup, and user interface.

2.1 Gathering data to transmit

The data gathering element of the project would poll LAN accessible ports, collecting the data together each time a authorised and authenticated request is given. Until permissions are revoked, the peer who requested data will have access. The software could allow custom routing, so that specific ports could be accessible at specific URLs. Each peer could be able to choose poll frequency, and could allow connected peers to access specific snapshots of the namespace.

A LAN accessible web server with an API to retrieve invoices runs at 192.168.0.7:8000. The business (Acme Inc.) wishes to share the default set of all invoices (no API parameters) with another peer on the network. Using the software, the business shares this API endpoint with the namespace 'invoices', and allows every peer they have connected with to access. This request could be formalised as: [p2p-protocol]://acme-inc/invoices

2.2 Data transmission

The data transmission element is responsible for sending the correct data to the correct peers. At its rawest form, this component should send the material directly to another peer without having to be transferred through an intermediary. Using NAT-PMP (Krochmal & Cheshire, 2013), the software will be able to connect to a port forwarded machine in another network, and transfer the data accordingly.

Because the software should be able to transfer files of unknown size, the transport mechanism has to have the capability to send large files. A TCP/IP port could transfer data in pieces. Pieces could contain some kind of status type so receiver knows what to do with information. Transmission could be done in an existing peer-to-peer protocol, such as BitTorrent (Johnsen, 2005).

2.3 Peer lookup

If a peer is not within the current peers' address book, there has to be a mechanism to add a peer by some unique identifier, or search the network for the relevant peer.

Peers should have unique metadata which is not bound to an IP address, to both allow further privacy and allow for peers to maintain connection despite dynamic IP address (Prime, 2019) allocation. This metadata should be the basis of peer lookup.

2.3.1 Peer has to be found on network

Each peer could have a unique name (stored in a DHT, perhaps using Chord, a popular DHT implementation) which could be used to locate the desired peer.

The software will have to request the address books from its peers, to see if they know the peer in question.

2.3.2 Peer exists with in address book

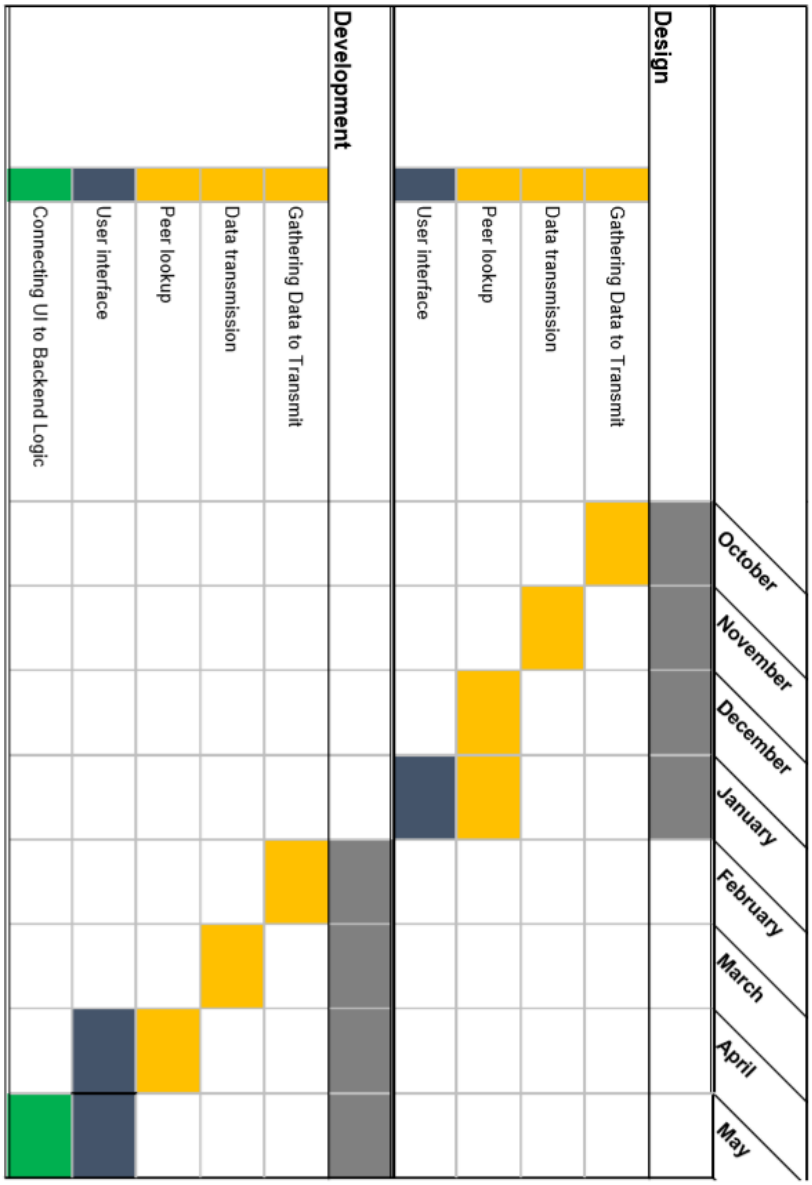
The software will take the peer's ID (generating an ID is something Chord may be able to solve) and attempt to connect at that address. If approval is given (perhaps via a specific handshake), then the peers have connected.

3 User interface and software development

The software's UI will be developed using the web stack, the React library, and Electron. Electron is a wrapper which allows this web stack to be used to build desktop applications (*Build cross platform desktop apps with JavaScript, HTML, and CSS.*, n.d.).

Because Electron uses Node.js, I will use either Node.js as the server language, or use node to start a Python or C++ server, which will be responsible for the peer-to-peer logic of the software.

4 Schedule



Orange: Backend task
Purple: Frontend task
Green: Backend+Frontend

References

Åslund, J. (2002). *Authentication in peer-to-peer systems.* Institutionen för systemteknik.

Baset, S. A., & Schulzrinne, H. (2004). An analysis of the skype peer-to-peer internet telephony protocol. *arXiv preprint cs/0412017*.

Build cross platform desktop apps with javascript, html, and css. (n.d.). Retrieved from https://electronjs.org/

Cavoukian, A., et al. (2009). Privacy by design: The 7 foundational principles. *Information and Privacy Commissioner of Ontario, Canada*, *5*.

Johnsen, B., Karlsen. (2005, 12). *Peer-to-peer networking with bittorrent.*

Krochmal, M., & Cheshire, S. (2013). Nat port mapping protocol (nat-pmp).

Kurucz, E. C., Colbert, B. A., & Wheeler, D. (2008). The business case for corporate social responsibility. In *The oxford handbook of corporate social responsibility.*

Liao, W.-C., Papadopoulos, F., & Psounis, K. (2006). An efficient algorithm for resource sharing in peer-to-peer networks. In *International conference on research in networking* (pp. 592–605).

Lwin, M., Wirtz, J., & Williams, J. D. (2007). Consumer online privacy concerns and responses: a power–responsibility equilibrium perspective. *Journal of the Academy of Marketing Science*, *35*(4), 572–585.

One year after cambridge analytica, survey reveals strong consumer privacy fears remain. (2019, Aug). Retrieved from https://www.slicktext.com/blog/2019/02/survey-consumer-privacy-fears-after-cambridge-analytica/

Pejchinovski, G. (2014, 02). Spotify - combining cache, peer-to-peer and server-client architectures for user satisfaction.

Prime, J. (2019, 09). *What is a dynamic ip address?* Retrieved from https://support.opendns.com/hc/en-us/articles/227987827-What-is-a-Dynamic-IP-Address-

Ruiz Forns, M. (2019). *Pageviews analysis | peer-to-peer and business-to-business.* Retrieved from https://tools.wmflabs.org/pageviews/?project=en.wikipedia.org&platform=all-access&agent=user&start=2015-07&end=2019-08&pages=Business-to-business|Peer-to-peer

Schulze, H. (2011, Jun). *Internet study 2008/2009.* https://www.ipoque.com/. Retrieved from https://web.archive.org/web/20110626192933/http://www.ipoque.com/sites/default/files/mediafiles/documents/internet-study-2008-2009.pdf

Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., & Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, *31*(4), 149–160.