Darklance

A Privacy-Focused Freelancing Marketplace

WWW.DARKLANCE.NET

January 29, 2019

Written by: Shane Radliff

Introduction

With the advent of the Internet, location-independent, freelancing jobs have begun to take hold. Instead of a daily commute to a fixed location and working 40+ hours a week for someone else, many individuals are taking their skills, services, and products "directly" to the consumer via the Internet. This provides many advantages, such as work freedom, a possible increase in take home pay, as well as the ability for individuals to make their own decisions about how they will deal with the taxman. Regardless of the decision, we believe that should be left up to the individual, not some third-party website or corporation (W2 employment).

Due to this shift towards remote work, many websites have sprung up, merely setting up the infrastructure to connect freelancers with clients. Some of these include Upwork, Freelancer, Audible, and a slew of others. While some of us have used these platforms, they are fraught with issues, the most important one being privacy.

To utilize these platforms, personally identifiable information is often necessary, sometimes including an approved piece of government identification, along with a W-9 for tax purposes. In large part, this revealing of personally identifiable information doesn't help improve the reputation systems used on these platforms. Whether someone uses their given name or a pseudonym, the reputation systems still work – if you're a client looking for a freelancer, you will likely not know the individual in question anyway.

Therefore, the invasion of privacy is strictly for regulatory compliance.

One final problem to touch upon is the exorbitant fees one is required to incur when using these sites. Using Upwork as an example, you as a freelancer would end up paying 20% for the first $500 you bill a client and 10% up to $10,000. This forces the freelancer to add in that fee to their bid, passing those costs onto the client. Furthermore, you'll likely have more success with monthly "Pro" plans, which cut into your profits even more.

In this fast-advancing world of decentralized technology, we believe this centralized, regulatory compliant setup to no longer be necessary. In this whitepaper, we will propose a new, decentralized solution to this problem, utilizing similar reputation systems combined with circles of trust and bitcoin for payments. Privacy is sacrosanct, and therefore anonymity and pseudonymity will be encouraged, and in some respects, mandatory.

Enter Darklance.

Terms

- **Bitcoin Core**: Reference implementation for the well-known p2p digital currency
- **Scuttlebutt**: A protocol for building decentralized applications that work well offline and that no one person can control
- **Internet Invisibility Project (I2P)**: An anonymous network layer that allows for censorship-resistant, peer to peer communication
- **Tor**: Free software enabling anonymous communication
- **Circle/Web of Trust**: A model for arranging relationships in order from most trusted (1st layer/hop) to least trusted (outer layer)

Guiding Principles

Before we move onto the inner workings of Darklance, let's begin by covering the guiding principles of this project:

- Working Knowledge of Cryptography
- Security Culture
- Technological Advancement
- Vonu

Working Knowledge of Cryptography

We will make Darklance as user-friendly as possible, but privacy expands to all parts of our daily lives, not just in our efforts to make money: a security flaw in one area could expose the other and vice versa. In addition to building this freelancing platform, we want to educate our users on cryptography, so that they can apply it to other parts of their human experience.

Not only is it just generally a good idea to understand the technology you're using, but more information leads to better decision-making when it comes to action; it can also prevent silly mistakes that could open one up to invasions of privacy, or, worst case scenario, coercion. And, let's face it, all it takes is one silly mistake.

To accomplish this task of education, we will put together tutorials, "best practice" guides, and easy-to-understand explanations of the tools we build into Darklance, in addition to how they work together.

Security Culture

Security culture is defined as the direct application of the right to privacy. As a strategy to maintain liberty, it is focused on the *how* of making privacy happen in the real world, given that the philosophical justification for privacy is self-evident.

Darklance will come chock-full of privacy-enhancing tools so that you can work without worry. All messaging between freelancers and clients will be encrypted, payments will be done in a secure, private manner, and we will encourage all our users to leave their "government names" on Upwork or similar sites.

Some projects will necessitate the sending and receiving of digital files. This will be done peer-to-peer and via encryption within the application. Beyond that, we will advise our users to encrypt and sign their files using Pretty Good Privacy (PGP) as an added layer of security. Of course, not everything is of a sensitive nature but security culture is a lifestyle, a habit, not just something you do every now and again.

Technological Advancement

The technology world moves fast in general, but in the realm of privacy and open source, change is ever more constant. Bugs (whether malicious or innocent) are constant in the building of new tools, which necessitates vigilance on the part of developers and end-users alike.

We will develop Darklance to be a resilient platform based off of flexible protocols, always ready to implement more secure and/or more efficient products, services, or security features.

Furthermore, Darklance will be designed and constructed utilizing a modular style and thus will follow the principle of "separation of concerns[1]" wherein each module has a specific purpose to fulfill.

Vonu: An Invulnerability to Coercion

Darklance is a tool built in accordance with the philosophy and freedom strategy called vonu. Briefly, vonu is an awkward contraction of VOluntary Not vUlnerable and the goal is to become as invulnerable to coercion as humanly possible.

Financial independence, location-independent employment, and security culture (encryption, Bitcoin, etc.) greatly increase one's invulnerability to coercion. Darklance is an open-source project, but we will request all our contributors to have a passion for expanding vonu, or, in other words, personal freedom.

For more information on vonu, check out *Vonu: A Strategy for Self-Liberation* and *The Vonu Podcast*.

The Network: Scuttlebutt (SSB)

Scuttlebutt Network

The following explanation will be taken in part from the Scuttlebutt Protocol Guide[2], with the rest being summarized. For a full explanation, please check out the guide in its entirety.

Scuttlebutt is a protocol for building decentralized applications that work well offline and that no one person can control. Because there is no central server, Scuttlebutt clients connect to their peers to exchange information. Herein, we will describe the protocols used to communicate within the Scuttlebutt network.

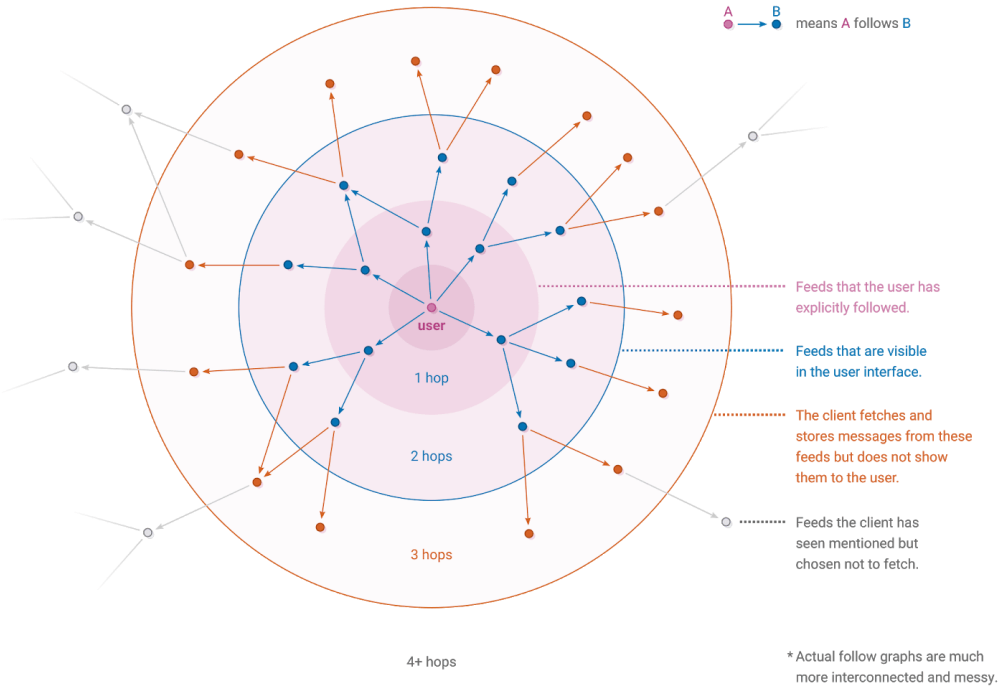
Scuttlebutt is a flexible protocol, capable of supporting many different types of applications. One of its first applications was as a social network, and it has also become one of the most compelling because the people who hang out there are not jerks. This guide has a slight focus on how to use Scuttlebutt for social networking, but many of the explanations will still be useful if the use is for something completely different, or if you are just curious how it works.

To use the protocol, users must first generate secret and public keys and then assign themselves a nickname or avatar to make themselves more easily recognizable. Connecting to other users is done via a secret, 4-step handshake. This process verifies public keys, creates a shared secret for encrypting future messages, and follows other security protocols to prevent man-in-the-middle attacks, etc.

Once peers (“users”) are connected, Scuttlebutt will then make requests, asking for the latest messages in a particular feed or for a particular blob. For Darklance specifically, these requests will be job feeds that freelancers and clients are subscribed to.

Since blobs generate a sha256 hash, users can be sure that messages, postings, etc. have not been tampered with.

Once peers are connected and immutable blobs exist, feeds can then follow other feeds. This is a way of saying, “I am interested in the messages posted by this feed.” These connections are made public on the network, which arranges them into a graph of who follows who:



Click the image to enlarge.

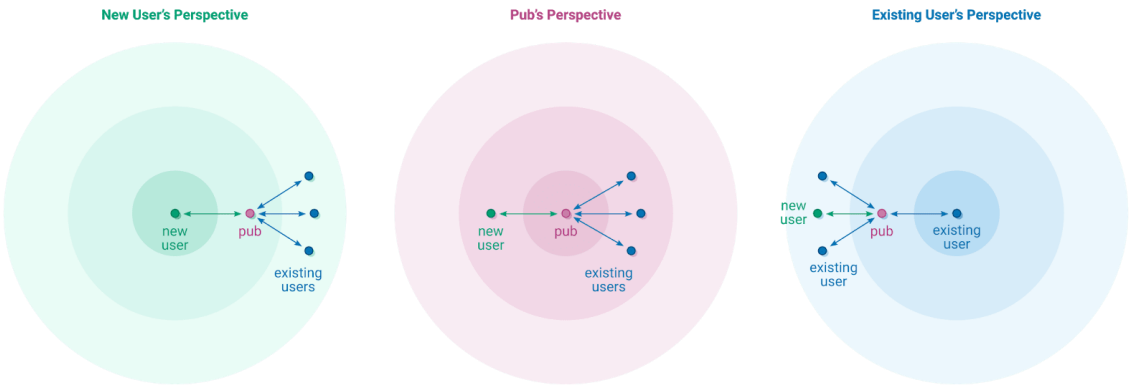
Therefore, by default, Darklance's circle of trust model (see below) is baked in at the protocol level.

Before users can connect with each other and follow feeds, they must first join a "pub", a publicly-accessible Scuttlebutt peer. They serve a social and technical purpose:

- Pubs serve as a gathering point for new users to find other people and for existing users to welcome people who have just joined.
- Pubs have a stable IP address and allow incoming TCP connections, which enables users to connect even if their internet service provider lacks dedicated IP addresses or refuses incoming connections.

Pubs speak the same protocol and behave as regular peers except that they are normally run on servers so that they are always online.

Joining a pub means following it and having it follow you back. After a new user joins a pub they will be able to see posts by the pub's other members, and crucially the other members will be able to see the new member. This works because everyone is now within 2 hops of each other:



Click the image to enlarge.

After joining a pub, a user may decide to follow some of its members directly, in which case the pub is no longer needed for feed visibility but is still useful for accepting TCP connections and replicating messages.

That said, we may decide to modify the way users connect to pubs, for at least the Darklance portion, if not for the social media portion as well. The way it's currently configured isn't the most user-friendly implementation for a freelancing marketplace, wherein individuals may not know each other personally or will otherwise fall outside the 2-hop default configuration.

Pub Moderators

Darklance will clearly be decentralized, but as the originators of this project, there are precautions we need to take in order to ensure we remain invulnerable to the coercion of governments.

We are specifically building a freelancing marketplace, but, due to the open-source nature of the project, it could be forked or built into many other things we never intended. The "it's decentralized" fact is no defense when there are violations of United States federal law.

Therefore, each pub will have moderators who will be paid to curate posts, verify ratings/reviews for jobs within their purview, and keep the server open so interested freelancers/clients are able to connect. This also allows a moderator to comply with law enforcement in the event that illicit goods/services are broadcasted in the pub.

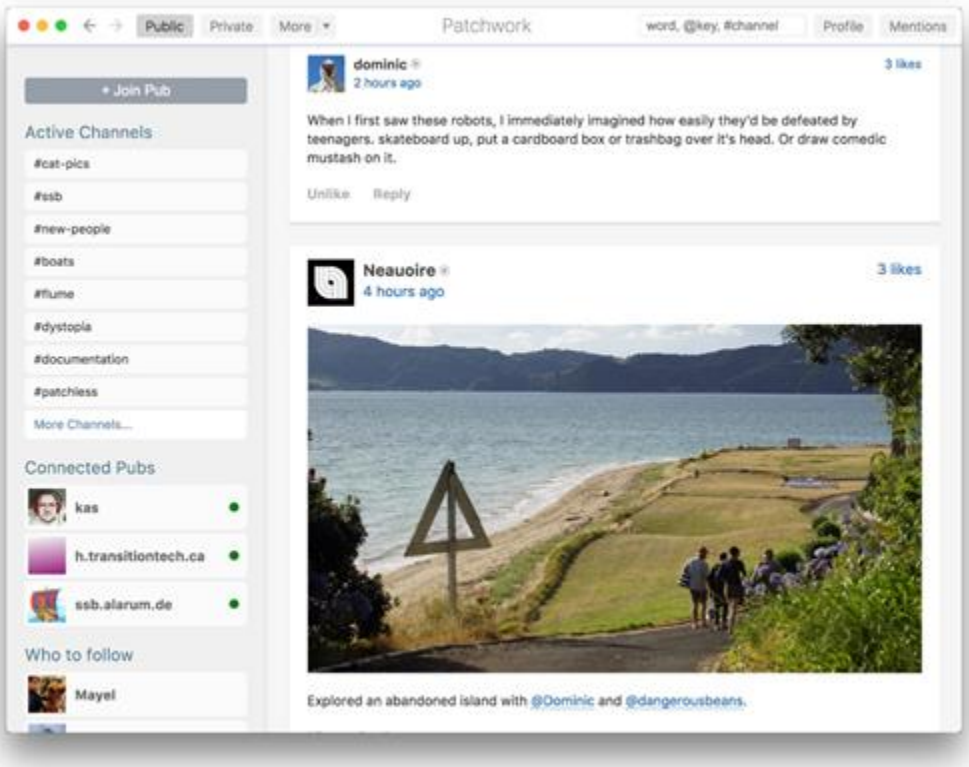
Undoubtedly, we aren't fans of compliance, but, as was mentioned with taxes in the introduction, individuals running the pubs (moderators) should be able to make their own decisions.

Internet Invisibility Project (I2P)

I2P is an anonymous network layer that allows for censorship-resistant, peer-to-peer communication. All traffic on the Darklance platform will be routed through I2P to increase user privacy.

Social Media Platform

As was mentioned above, Scuttlebutt is merely the network protocol and various applications can be built on top of it. One of those possibilities, a social media platform, already exists in the form of Patchwork. We will add that into Darklance as an additional feature and in-road into our ecosystem.



Click the image to enlarge.

The user interface will be re-designed, re-named, and more intuitive, but the above image will at least give you an idea. Much like Slack/Discord, channels ("pubs") you're connected to appear on your toolbar; the only difference for Darklance is that some of those channels/pubs will be feeds you subscribe to for work as a freelancer or to post available work as a client.

Having a social media platform alongside Darklance will be advantageous for numerous reasons:

1. With the "censorship"/purging that is taking place on outlets like Fascistbook, Twitter, etc., individuals are looking for decentralized alternatives. We can use social media to draw in more potential freelancers/clients.

2. Everyone who uses channels (“pubs”) routed through Tor and/or I2P automatically acts as nodes for these networks, making it and Darklance more secure.
3. It’s a great tool for networking – if John is looking for writing work, he can put up a post on social media and get help from his network! They may post some links to job postings, tag individuals who may be seeking a freelancer, etc. And it’s all done within the same platform.
4. If we achieve our goal of user-friendliness without sacrificing security, this social media platform could help draw new folks into the open-source, decentralized, bitcoin ecosystem.

Reputation Systems

Bitcoin will not only be used for payments, but will also be the basis for reputation within Darklance. A bitcoin public key will be tied to a Scuttlebutt key, and therefore, previous activity will be trackable and verifiable on the immutable bitcoin blockchain. That said, we will implement two other systems to reduce the risk of fraud.

Ratings and Reviews
While this won't be our core reputation system, it is the tried and true one. Etsy, Ebay, and other sites have been utilizing the rating/review model for a long time and it works: if someone has 1,000 ratings averaging 4.9/5 stars, you can likely feel quite confident that both parties will leave the exchange satisfied.

When users on the Darklance platform get beyond the second layer of their circle/web of trust (below), this will be the most secure way to approach potential transactions.

Circle/Web of Trust
As was defined in the terms section above, a circle/web of trust is a model to arrange relationships from most trusted to least trusted. Your inner circle (1st layer) would include your close friends and family members, whereas your outer circle (3rd layer) would contain complete strangers. In between is the second layer, which would be your "friend-of-a-friend" or maybe a more distant relative.

Regardless, the idea is that the closer towards the inner circle you are, the less susceptible you are to fraud. Scuttlebutt's network protocol "enforces" this by default and the only "modification" we may make for Darklance is the user-friendliness in making connections.

Our best practice guides will always recommend going with someone from your first or second layer, but obviously, that may not always be possible.

Privacy and Security

Scuttlebutt Security/Privacy
Scuttlebutt utilizes public-private key encryption by default (Ed25519) and a secret handshake to authenticate and setup encrypted channels between peers. In other words, public keys are exchanged and users can now interact.

During the transmission of messages, Scuttlebutt employs Box stream, which protects against man-in-the-middle interference; all messages and transmissions are signed/authenticated by the sending party.

Instant Messaging
Encrypted messaging is a default process in Scuttlebutt. The sender generates a Curve25519 key pair to encrypt the message header and a random 32-byte secret key to encrypt the message body. Then, the sender uses scalar multiplication to derive a shared secret for each recipient.

Voice/Audio Chat
If we are able to, we would like to add-in encrypted voice/audio chat functionality on Darklance. Thankfully, Jitsi[3] has been around and building this encrypted, open-source software for over 15 years.

Their library of API documentation is extensive and all calls and messages are end-to-end encrypted using Zimmerman Real-Time Protocol and Off-The-Record, respectively.

File Transfers
File transfers are processed via the RPC protocol described above for fetching feeds, messages, etc.

For example, let's say John requests a blob from Jane. Jane receives said request and begins streaming the contents of the blob in return. If it's a larger file of, say, 161,699 bytes, Jane's system may decide to send it in two full pieces of 65, 536 bytes and a final piece with the remaining 30,627 bytes.

Once John begins to receive the transmission, the requester will check to ensure the details (hash, size, maximum size of the blob) match the sha256 checksum they asked for. If not, the request will be rejected and an error message will be sent back to Jane.

Depending upon the construction and ease of this process, we may or may not decide to utilize Interplanetary File Storage or BitTorrent for larger files.

Payment

Bitcoin

When Darklance is launched, only one crypto-currency will be available for use: bitcoin. It is the most secure, most immutable, and most censorship-resistant chain in addition to having the highest trading volume of any digital currency out there. Of course, Darklance is open-source/decentralized, so users could add in whatever coin they wish. It's just not something we'll put development time into.

We plan on using Electrum servers over Tor for privacy reasons.

Trustless Multisig Escrow

(Excerpted from *The Future of Bitcoin Escrow*[4] by Jeremy Spilman)

Typically, an escrow involves three parties, where a neutral third party holds the funds while goods are delivered. The escrow agent can either return payment to the buyer, release payment to the seller, or provide for some split in case of disputes. The escrow agent model doesn't require any fancy crypto; full trust is placed in the escrow agent by both the buyer and seller, and funds are sent from buyer → escrow → seller as usual. However, multisig offers two different ways to run an escrow which changes things up a bit.

One option is for the buyer to send funds into a 2-of-3 multisig transaction instead of directly to the escrow agent. The buyer collects one of their own public keys, plus a public key each from the escrow agent and the seller, and then sends the desired number of their own coins into a new account ('output') with the requirement that those coins can only be spent if 2 of 3 parties sign off.

Even more interesting, is the two party escrow using a 2-of-2 multisig transaction. When Alice sends funds into a 2-of-2 signature multisig address with Alice and Bob's public keys, what she's doing is giving Bob a say in how to spend those coins. Alice and Bob must agree on how to spend them, or else the coins cannot be spent. Removing the third party presents an interesting case where simpler structure leads to more complex behavior. What if Alice and Bob simply can't agree on how to release the coins? How might Alice or Bob try to exploit this?

When Alice sends funds into the 2-of-2 multisig, she's put those coins at risk. If Bob does nothing, and signs nothing, Bob never lifted a finger, but for Alice those coins are effectively lost. This basic asymmetry undermines the core value of a two party escrow, so we are encouraged to find some ways to remedy the situation. We either need Bob to have some skin in the game, so Alice feels more comfortable putting her coins at risk, or we need some fail-safe mechanism to give Alice some downside protection. Bitcoin lets us do either, or both.

The first option is to lock up some of Bob's coins along with Alice's. That way, Bob can't simply troll Alice into losing money without losing some of his own. Technically, what we need to do is collect properly sized inputs from both Alice and Bob, and then collect all the coins together into a single multisig output. Doing everything within a single transaction is important, because that way the operation is atomic — either both Alice and Bob's coins end up in escrow, or nothing ends up in escrow. The coordination level required in order to construct such a transaction is trivial for a centralized service holding both Alice and Bob's wallets, but highly unusable assuming client-hosted wallets where Alice and Bob only communicate over PGP and Tor.

The second option is to setup an automatic release from the escrow where coins are either sent back to Alice, or split some way between Alice and Bob, at some predefined points in the future. This can be done if Alice creates the initial transaction sending her coins into escrow, and she signs the transaction in order to determine its transaction ID (TxID), but then instead of broadcasting the transaction, she keeps the transaction itself secret.

Alice sends just the TxID of the not-yet-escrowed coins to Bob, and tells Bob he must sign a new transaction which sends the coins from the escrow back to Alice (or split somehow between Alice and Bob). But this refund transaction will be locked, so that it can only be redeemed at some specified time in the future, using a feature called nLockTime. Alice only broadcasts the transaction which releases her coins into the escrow after she gets back the signed, post-dated refund transaction(s) from Bob. In other words, Bitcoin lets Bob commit to the fail-safe before Alice commits her coins to escrow, which is frankly pretty awesome.

Of course, the problem with Bob providing Alice a fail-safe is that now Bob might be worried about Alice simply waiting him out. But the fail-safe can be as 'strong' or 'weak' as the two parties want to make it, so it's likely they can agree on something. For example, 25% refund back to Alice after 6 months, or 50/50 split after 2 years, anything is possible.

So, what does the concept of "trustless multsig escrow" mean for Darklance? Quite frankly, it means a third-party is optional and it is purely peer-to-peer. Some may feel more comfortable with a moderator, and that's totally fine. We just want you to have options.

Transactions/Payments

Keeping the above excerpt in mind, let's examine how a transaction would be completed in Darklance.

Let's say Bob (a freelancer) accepts a job from Alice (a client). Bob finishes the job to Alice's satisfaction and now it's time to complete the transaction. If they decide to use an escrow service, Alice will send the payment to a third party's multisig Bitcoin address.

After she confirms receipt of the files (if applicable) and completion of the job in question, funds will be released from escrow to Bob.

If, on the other hand, they decide to use trustless multisig escrow, the process is as follows.

If the job in question is being paid on a project basis rather than hourly, Bob and Alice will likely choose to setup the escrow ahead of time, in order to reduce the risk of fraud. If it is an hourly job without a known payment amount, the trustless multisig will take place at the completion of the job but before final payment.

Regardless, here's how it would look protocol-wise.

After Bob's Darklance client verifies that the refund transaction is correct and signed by Alice, they are prompted to fund the escrow. Bob will then send Alice the bill/receipt using the 'payment' message.

Receipt
Once the freelancer's client receives the payment message, they will wait for the funding transaction (which locks funds in escrow) to confirm. The client will then ask the freelancer to deliver the project files to the client. Then the freelancer's client will construct a final bitcoin transaction which sends all escrowed funds to the freelancer. The client sends a signed copy of this transaction to the freelancer in a 'receipt' message.

Finalization
After the freelancer provides the project files to the client, the client may sign and broadcast the final escrow release transaction which sends the payment to the freelancer. The client will then send back a 'finalize' message indicating the escrow has been released.

Darklance Fee
It is at this point the Darklance system will take a 2% fee for the completion of the contract (see below).

Refund
If the project files are not provided, the client may alternatively sign and broadcast the refund transaction created earlier which disperses the escrowed funds between both parties based on the terms agreed to by the client.

Feedback
Regardless of the outcome of the trade, both parties are likely to want to give feedback in order to participate in the reputation system. Each party will send a 'feedback' message directly to the pub moderator which contains the IDs of the involved parties, the TxID of the transaction which released escrow (be it a normal finalization, or a refund) and a boolean value indicating an upvote or downvote.

Upon receipt of this message, the moderator will verify the registration of the two parties, verify that the referenced transaction has been released from an escrow between these specific parties, and record the feedback as valid.

Bitcoin Fees

To use the bitcoin network, fees must be paid to the miner. Depending upon the amount of transactions within the mempool, this fee and the confirmation time may vary. This will be paid by the client when they send the transaction to the freelancer in question.

Fees

The Darklance platform will have fees-for-use, but nothing even close to what is required by Upwork, Freelancer, and similar sites. There are two methods in which these fees will be used.

Development Funding
A great way to ensure Darklance is continually developed upon is to have a self-sufficient way to fund improvements and protocol changes. For every completed contract, the Darklance platform will take 2% of the total cost of the job for fees.

For example, if John did a $1,000 contract for Jane, Darklance would take $20. That same contract on Upwork could cost as much as a couple hundred dollars in fees.

Darklance Community Funds
50% of the above fees will be allocated towards community development. If an individual/team is working on a great addition to Darklance, the community may decide to allocate some funds to them for the building/completion of the project.

It's likely these funds would be stored in a multisig Bitcoin wallet.

The method for disbursing these funds hasn't been explored fully as of the time of publication. We are open to suggestions.

Roadmap

We are currently in the early stages of development, but being able to build off of Scuttlebutt and Patchwork will help speed up the progress significantly, rather than having to start from scratch. Although, it's worth noting that while we are all deeply committed to Darklance, it is not something we can devote all of our time to, at least right now. With the support of a community to assist in marketing, development, testing, etc., we are hoping we can have it in alpha/beta by the end of 2019.

If this describes you, we would love for you to join our Discord channel (https://discord.gg/PaXYH6j) and get connected with the dozens of individuals already contributing to the project in some way.

Here are some short and long term goals for Darklance:

- Lightning Network: When the lightning network is further developed upon, we would love to add it into Darklance. This would negate the high fee/slow confirmation time risk associated with full blocks.
- Mesh Networking: This is a long-term goal, but if/when Darklance has a large, worldwide user base, we would love to do some testing with mesh networks. That is really the next major step in the building of free societies -- having our own Internet outside the control of Internet service providers.
- Smartphone Applications: For a product/service to really scale in user base, mobile applications are absolutely necessary. This will be more of a long-term goal, but is something we need to begin thinking about.

This is not a comprehensive list, but should give you a gauge as to our goals for the project, beyond just Darkbook and the freelancing platform.

"An ethical enclave is defined here as voluntary transactions between individuals who are living under a collectivist government, when such transactions are conducted independent of that government. 'Ethical' denotes the distinguishing characteristic of the participating individuals: an adherence to the ethical principle of voluntarism…And 'enclave' denotes physical emersion within a philosophically alien society."
-Rayo, November 1965

References

1. Separation of Concerns
2. Scuttlebutt Protocol Guide
3. Jitsi
4. The Future of Bitcoin Escrow by Jeremy Spilman