SURREAL: **S**ec**UR**e cove**R**t pe**E**r communic**A**tion over BitTorrent protoco**L**

Avinash Srinivasan
Computer and Info. Sciences
Temple University
Philadelphia, PA 19121
Email: avinash@temple.edu
Fellow – National Cybersecurity Institute

Jie Wu
Computer and Info. Sciences
Temple University,
Philadelphia, PA 19121
Email: jiewu@temple.edu

Richard Joy and Hatoon Aldarrhab
Volgenau School of Engineering
George Mason University,
Fairfax, VA 22030
Email: [rjoy, haldharr]@gmu.edu

***Abstract*—Today, peer-to-peer (p2p) networks have risen to the top echelon of information sharing on the Internet. It is a daunting task to prevent sharing of both legitimate and illegitimate information such as – music, movies, software, and child pornography – on p2p *overt* channels. Considering that, preventing *covert* channel information sharing is inconceivable given even its detection is near impossible. In this paper, we describe SURREAL – a technique for covert communication over the very popular p2p BitTorrent protocol. Standard BitTorrent protocol uses a 3-step handshake process and as such does not provide peer authentication service.**

In SURREAL, we have extended the standard handshake to a 6-step authenticated covert handshake to provide peer authentication service and robust peer anonymity with one way functions. After authenticating a potential covert partner, participating peers send data over an encrypted covert channel using one a standard BitTorrent message type. We have also SURREL's security robustness to potential attacks. Finally, we have validated SURREAL's performance and presented results comparing it with [4] and [5].

Keywords*—*Authentication, BitTorrent, covert channel, handshake, information hiding, p2p networks, security, steganography.

I. Introduction

With the rapid evolution and widespread adoption of peer-to-peer (p2p) networks, for information sharing, preventing sharing of both legitimate and illegitimate information – such as music, movies, software, and child pornography – even on p2p *overt* channels is a daunting task. Furthermore, given that, even the detection of *covert* channel ($\mathcal{CC}$) is near impossible, preventing illegitimate $\mathcal{CC}$ information sharing is simply impossible. p2p networks contain ad-hoc, decentralized structures and autonomous peers who can join and leave the network at will. These characteristics of the p2p networks provides a fertile ground for malicious actors. Therefore, security requirements for p2p networks are significantly more challenging.

Steganography, a very old, popular and powerful $\mathcal{CC}$ technique, has been applied in numerous domains with numerous carrier file types – such as audio, video, and graphics. Contemporary steganography methods that implement $\mathcal{CC}$ within network traffic are highly dependent on the particular cover file data or the network protocol. While there are numerous variations of steganography implementations, most of them use the same fundamental information hiding principles.

p2p networks, by default, offers peer anonymity – a critical requirement for the adversary in scenarios such as distributing malware, child pornography, miscellaneous contraband, or when deploying botnets – to prevent attack-source traceback. p2p networks also offer network size anonymity, with which the extent of the network is never truly known, leaving a window of opportunity for the attackers to find alternate routes, attacks, and attack vectors if some parts are disabled. Therefore, p2p networks are very attractive to cybercriminals. However, p2p BitTorrent (BT) protocol's standard 3-step handshake lacks *authentication* services – a key requirement in covert communications to protect the confidentiality of information and anonymity of the peers.

Therefore, in this paper, we propose SURREAL – a mechanism for establishing authenticated $\mathcal{CC}$ over p2p BitTorrent (BT) protocol. SURREAL implements covert peer communication building on steganography-based information hiding. In SURREAL, the covert message is first encrypted with the sender's choice of encryption algorithm. Subsequently, the encrypted secret message is hidden in a $\mathcal{CC}$ within the standard BT overt communication channel – one that is overt and open to all sorts of monitoring and sniffing. Hence, security and robustness against detection and confidentiality breaches are very critical.

We have implemented a proof-of-concept prototype of the proposed SURREAL using SNARK, an open-source java framework of the BT protocol. However, one key requirement for the proposed SURREAL to survive detection is that the $\mathcal{CC}$s must be built on ubiquitous protocols such that the cover communication blends inseparably with legitimate network traffic, yet, the protocol obscures secret messages such that they are not retrievable by a non-colluding peer.

A. Background and Preliminaries

Covert Channel ($\mathcal{CC}$) is a communication channel, that utilizes a portion of the main channel's bandwidth to transport secondary information, with or without malicious intentions. However, $\mathcal{CC}$ communication is a violation of security of the underlying communication network.

The communication through $\mathcal{CC}$ continues to be a significant challenge, especially since even its existence is almost

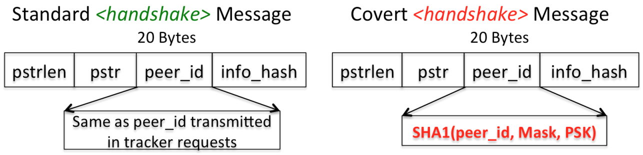

Standard <handshake> Message
20 Bytes
pstrlen
pstr
peer_id
info_hash
Same as peer_id transmitted in tracker requests
Covert <handshake> Message
20 Bytes
pstrlen
pstr
peer_id
info_hash
SHA1(peer_id, Mask, PSK)

Figure 1. $\langle hand\_shake\rangle$ of standard BitTorrent vs. SURREAL.

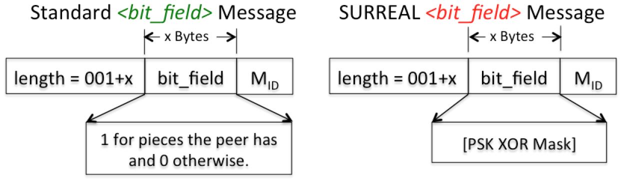

Standard <bit_field> Message
x Bytes
length = 001+x
bit_field
M_ID
1 for pieces the peer has and 0 otherwise.
SURREAL <bit_field> Message
x Bytes
length = 001+x
bit_field
M_ID
[PSK XOR Mask]

Figure 2. $\langle bit\_field\rangle$ of standard BitTorrent vs. SURREAL.

always unknown. $\mathcal{CC}$s can be used for legitimate and malicious purposes – *censorship violations*, *anonymity* and *privacy preservation*, *profiling* users for business interests, *national security* matters, etc.

BitTorrent (BT) **Protocol**, a p2p protocol,in which "tracker" server keeps track of file copies residing on peer machines, availability when clients' request, and also helps to coordinate efficient transmission and reassembly. The peer wire protocol consists of an initial $\langle\ hand\_shake\rangle$. After that, peers communicate via an exchange of integer *length-prefixed* messages. The $\langle\ hand\_shake\rangle$ is a required message, of size $[49 + len(pstr)]$bytes, must be the first message transmitted by the client, and has the format $\langle\ hand\_shake\rangle$.

A peer message exchange starts once a client has selected a torrent file. The peers start with a $\langle\ hand\_shake\rangle$ message followed by a $\langle\ bit\_field\rangle$ message, which has the fixed ID=5. The $\langle\ bit\_field\rangle$ message may only be sent immediately after the handshaking sequence is completed, and before any other messages are sent. It is optional, and need not be sent if a client has no pieces. This is followed by other messages of one of 9 types $\langle\ request\rangle$ with ID=6, $\langle\ piece\rangle$, with ID=7, etc. The other control messages that are intermixed for purposes of communication control include – $\langle\ choke\rangle$, $\langle\ unchoke\rangle$, $\langle\ interested\rangle$ $\langle\ not\_interested\rangle$, and $\langle\ cancel\rangle$ [12].

In the research presented in this paper, we will primarily focus on the following three message types – $\langle\ hand\_shake\rangle$, $\langle\ bit\_field\rangle$, and $\langle\ piece\rangle$. In addition to $\langle\ hand\_shake\rangle$, there are several other message types all of which take the form $\langle\ length\_prefix\rangle$ $\langle\ message\_ID\rangle$ $\langle\ payload\rangle$, where length prefix is a four byte big-endian value, message ID is a single decimal byte, and the payload is message dependent.

Standard BT **peer handshake** is a 3-step handshake and does not provide authentication services – a key requirement for covert communication. Consequently, a peer p_x has no way of knowing the identity of a connecting peer p_y, except through the $\langle hand\_shake\rangle$ message, which can be easily *spoofed*. To counter this weakness, and provision authentication services, we propose design SURREAL – a 6-step authenticated covert handshake protocol for $\langle\ hand\_shake_{cov}\rangle$ message exchange containing hashed p_{id}. Hence, with SURREAL, a peer shall be capable of managing its existing peer connections without solely relying on the p_{id} alone.

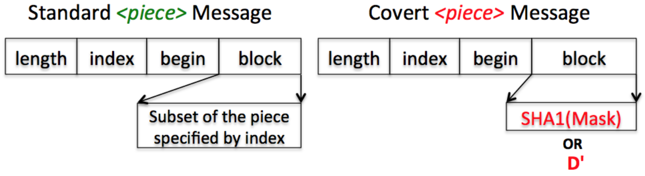


Figure 3. $\langle PIECE\rangle$ of standard BitTorrent vs. SURREAL.

B. Summary of Contributions and Roadmap

To the best of our knowledge, SURREAL is the first attempt to present a $\mathcal{CC}$ by substantially enhancing the standard BT protocol. SURREAL is the first attempt at a synchronous $\mathcal{CC}$ that modifies the $\langle\ hand\_shake\rangle$, $\langle\ bit\_field\rangle$, and the $\langle piece\rangle$ messages of a standard BT. SURREAL enhances the standard 3-step BT handshake into a 6-step authenticated covert handshake $\langle\ hand\_shake_{cov}\rangle$ enabling not only *peer authentication* service, but also providing robust *peer anonymity* service, in addition to *confidentiality* and *data integrity* services.

SURREAL is secure and robust to popular p2p attacks and colluding peers can use any arbitrary torrent for covert message exchange. We have discussed its security robustness to the three core security requirements – *Confidentiality*, *Integrity (Authentication)*, and *Availability*. Finally, we have compared the performance of SURREAL with two other state-of-the-art techniques [4] and [5] and confirm that it achieves significantly higher throughput per round of protocol execution.

The rest of the paper is organized as follows. In section II, we discuss SURREAL system model followed by the details of SURREAL design in section III. Then, in section IV-A, we discuss the implementation details of SURREAL followed by its security analysis in section-V. Finally, in section VI, we present relevant related works followed by conclusion with directions for future research in section VII.

II. System Model

We assume two types of peer nodes – regular (non-colluding) peers p^{reg}_{id} and covert peers p^{cov}_{id}. A given p_{id} satisfies $\in p^{reg}_{id}$, $\in p^{cov}_{id}$, or $\in p^{reg}_{id} \cap p^{cov}_{id}$. Each peer maintains two lists – ${}^{pot}L^{cov}_{id}$ is the list of potential covert partners to be authenticated, and ${}^{conf}L^{cov}_{id}$ is the list of authenticated covert partners.

Additionally, all covert peers have access to a *pre-shared secret key*, PSK, which is used for message encryption/decryption, and for cryptographic operations, any symmetric key algorithm such as 3-DES or AES can be used. Also, during encryption, if PSK is shorter than D, then PSK is repeated to make the two equal.. To being, all potential p_{id}'s are kept in the ${}^{pot}L^{cov}_{id}$, and the authenticated ones are moved to ${}^{conf}L^{cov}_{id}$. During the $\langle\ hand\_shake_{cov}\rangle$, peers also exchange a unique mask value that is randomly generated for each communication session, to be used along with PSK for encrypting their communications.

A key aspect of the standard BT is that only two message types carry a payload data of any significant size – $\langle bit\_field\rangle$

and $\langle piece\rangle$ messages [3]. However, since $\langle bit\_field\rangle$ message is sent only once, it provides limited utility by constraining the $\mathcal{CC}$ capacity impeding the long term covert message throughput. For this reason, we choose the $\langle piece\rangle$ message as our covert data delivery vehicle in designing SURREAL.

Finally, in designing SURREAL, we assume that the sender over the $\mathcal{CC}$ is also the sender over the overt channel. Hence, sender can potentially manipulate the overt channel properties to influence the characteristics of $\mathcal{CC}$ to maximize his payoff.

III. SURREAL Design

We begin with a discussion on the proposed 6-step $\langle$ $hand\_shake_{cov}\rangle$ protocol, which we build on top of the existing BT's 3-step handshake process. Then we discuss the covert communication using SURREAL, with detailed examples and the associated algorithms. In our proposed SURREAL, we consider two peers, denoted as p_x and p_y, who transfer information overtly on the network, but employ data hiding within BT protocol to communicate covert information.

A. The $\langle$ $hand\_shake_{cov}\rangle$ Message

The p_{id} in a standard BT $\langle hand\_shake\rangle$ message is usually the same as the p_{id} transmitted in the tracker requests. In designing SURREAL, we have modified this $\langle$ $hand\_shake\rangle$ message to be the $sha1$ hash of the triple $\langle p_{id}, PSK, p_{id}^{mask}\rangle$, as shown in equation 1.

$$\langle hand\_shake_{cov}\rangle^{x\to y} = sha1(p_x, PSK, p_x^{mask}) \tag{1}$$

This modification provides strong *anonymity* since sha1 is a one-way function. In this triple, the peer mask, p_{id}^{mask}, is randomly generated by each peer, which is exchanges with all the other peers on itsL_{id}^{cov}, and is used in establishing a secure covert channel. Figure 1 compares $\langle$ $hand\_shake\rangle$ and $\langle$ $hand\_shake_{cov}\rangle$.

For illustration, consider two colluding peers $Alice(p_A)$ and $Bob(p_B)$. To begin, p_A computes $\langle hand\_shake\rangle^{A\to B}$ (equation 2), and exchanges it with p_B. Upon receiving $\langle hand\_shake_{cov}\rangle^{A\to B}$, p_B is not certain if p_A is a covert partner, because $sha1(p_A||PSK||p_A^{mask}) \notin L_B^{cov}$. Therefore, p_B saves p_A from the $\langle$ $hand\_shake_{cov}\rangle^{A\to B}$ on$^{pot}L_B^{cov}$ for verification. p_B treats p_A as a potential covert partner and responds with $\langle$ $hand\_shake_{cov}\rangle^{B\to A}$ (equation 3), from which p_A extracts and saves p_B on$^{pot}L_A^{cov}$ for verification.

It is important to note that since peers have no way to know the p_{id} of an incoming connection, except the handshake, they must treat all incoming connections as potential partners and need to authenticate them prior to engaging in covert communication. It then computes $sha1(p_A||PSK||p_A^{mask})$ for each p_{id} on L_B^{cov}. Only if the p_id sent in the $\langle$ $hand\_shake_{cov}\rangle$ matches the $sha1$ value for a p_id (p_A in this case) on ${}^{pot}L_{id}^{cov}$, p_A will be moved to ${}^{conf}L_{id}^{cov}$ and receives $\langle$ $bit\_field\rangle$ (equation 5) from p_B. Figure I-A compares the format of standard BT $\langle bit\_field\rangle$ with SURREAL.

Algorithm 1 Computing Ones

```
procedure ONES()
    $CountOne \leftarrow 0$
    $len \leftarrow \mathcal{M}.length()$
    while $i \le len$ do
        if $\mathcal{M}[i] == 1$ then
            $CountOne \leftarrow CountOne + 1$
    return $CountOne$
```

$$\langle hand\_shake_{cov}\rangle^{A\to B} = sha1(p_A||PSK||p_A^{mask}) \tag{2}$$

$$\langle hand\_shake_{cov}\rangle^{B\to A} = sha1(p_B||PSK||p_B^{mask}) \tag{3}$$

B. The $\langle bit\_field\rangle$ Message

In the standard BT protocol execution, the peers exchange a $\langle$ $bit\_field\rangle$ message immediately after the $\langle$ $hand\_shake\rangle$. The $\langle$ $bit\_field\rangle$ message has the following format: $\langle$ len=0001 + $X\rangle$ $\langle$ id = 5$\rangle$ $\langle$ $bit\_field\rangle$. The $\langle$ $bit\_field\rangle$ message contains $1's$ for pieces that the peer has, and $0's$ otherwise. In SURREAL, p_A will compute a modified $\langle$ $bit\_field\rangle$ message (equation 4) and exchange it with p_B. p_B then computes $(PSK \oplus \langle bit\_field\rangle)$ to extract p_A^{mask}.

$$\langle bit\_field\rangle^{A\to B} = \left[PSK \oplus p_A^{mask}\right] \tag{4}$$

$$\langle bit\_field\rangle^{B\to A} = \left[PSK \oplus p_B^{mask}\right] \tag{5}$$

C. The $\langle$ $piece\rangle$ Message

The standard BT $\langle piece\rangle$ message has the format:

$$\langle piece\rangle = \left[\langle len = 0009{+}X\rangle\langle id = 7\rangle\langle index\rangle\langle begin\rangle\langle block\rangle\right] \tag{6}$$

Algorithm 2 Message Encryption in SURREAL

```
procedure ENCRYPTION
    $p_x, p_y$ : $colluding\_peers$
    $p_x \leftarrow sending\_peer$; $p_y \leftarrow receiving\_peer$
    if $p_x$ has $\Big((S)$ && $\langle piece\rangle$ in $\langle block_D\rangle\Big)$ then
        $D_x \leftarrow D \oplus PSK$
        if $\mathcal{M}[n \mod length(\mathcal{M})] == 1$ then
            $S_p \leftarrow pad(S)$
        for all $[n]$ do
            if $\mathcal{M}[n \mod length(\mathcal{M})] == 1$ then
                $D'[n] \leftarrow D_x[n] \oplus S_p[n+1]$
            else $D'[n] \leftarrow rand[0, 1]$
        Return $D'$
```

Algorithm 3 Message Decryption in SURREAL

```
procedure DECRYPTION
    $input \leftarrow \langle D', D, PSK, M\rangle$
    $D'_x \leftarrow D' \oplus D$
    $D_x \leftarrow D'_x \oplus PSK$
    $count \leftarrow 0$
    for all $[n]$ do
        if $\mathcal{M}[n \mod length(\mathcal{M})] == 1$ then
            $S_p[count] \leftarrow D_x[n]]$
            $count \leftarrow [count + 1]$
    return $S$
```

where $\langle\ index\rangle$ specifies the zero-based piece index, $\langle\ begin\rangle$ specifies the zero-based byte offset within the piece, $\langle\ block\rangle$ is the actual data, which is a subset of the piece specified by index. In SURREAL, p_A sends a $\langle\ piece_{cov}\rangle$ message, with the $sha1$ hash of p_B^{mask} as the payload.

$$\langle piece\rangle^{A\to B} = \big[index = 0; begin = 0; block = sha1(p_B^{mask})\big] \quad (7)$$

$$\langle piece\rangle^{B\to A} = \big[index = 0; begin = 0; block = sha1(p_A^{mask})\big] \quad (8)$$

The message should indicate that for this $\langle\ piece_{cov}\rangle$, $index = 0$, $begin = 0$, and $block = p_A^{mask}$ (equation 7). p_B compares $sha1$ hash from the $\langle\ piece_{cov}\rangle$ with the true $sha1$ hash of p_B^{mask}. A successful match authenticates the identity of p_A to p_B. Similarly, when p_B sends $\langle\ piece_{cov}\rangle$ with the $sha1$ hash of p_A^{mask} as the payload (equation 8), p_A authenticates p_B, as above, prior to covert communication. Once p_A and p_B have successfully authenticated each other, it completes a two-way authenticated $\mathcal{CC}$ between p_A and p_B. Figure 3 compares the $\langle\ piece\rangle$ with $\langle\ piece_{cov}\rangle$.

D. SURREAL Operations – Discussions

During the initial $\langle hand\_shake\rangle$ and $\langle bit\_field\rangle$ exchange, three specific situations will result in peer severing the connection – 1) connecting peers do not receive a matching info hash, 2) requesting peer does not receive the expected p_{id}, and 3) $\langle bit\_field\rangle$ message is longer than expected or has reserved bits overwritten [12]. Once peers get past this stage, then communications are controlled through peer messages such as choke or interested messages.

Message En/Decryption with the random mask. For discussions in this section, let us assume p_x, p_y are colluding peers. In figure 4, let $\mathcal{S}$ be the secret message, $\mathcal{M}$ be the mask of p_x, and D be the next block to be transmitted. Algorithm 2 and figure 4(a) collectively illustrate SURREAL's encryption of $\langle block\rangle$ portion of the $\langle piece_{cov}\rangle$ message prior to transmission.

Algorithm-1 is the routine that is used during encryption to check for significant bits to hold the covert message. Note that the $\mathcal{M}$ is used to determines which bits of the data payload contain hidden data and $\mathcal{M}$'s length is equal to the number of pieces in the torrent being exchanged by the swarm. First, p_x will compute $D_x = [D \oplus PSK]$. Then, using $\mathcal{M}$, p_x will compute $\mathcal{M}(D_x)$ - the significant bits of D. Now, if needed, p_x will pad $\mathcal{S}$ starting with '1' followed by required number of '0's to make $\mathcal{S}.lenght$= $\mathcal{M}(D_x).length$, and denote the padded version as $\mathcal{S}_p$. Finally, p_x will compute D', the encrypted secret that it transmits to p_y. The encryption process is presented with an example, capturing the idea intuitively, in Algorithm 2 and figure 4(a).

p_y, upon receiving D', will retain all $\langle\ piece_{cov}\rangle$ messages and decrypts once all relevant $\langle\ piece_{cov}\rangle$ have been received. In order to successfully decrypt D' to obtain $\mathcal{S}$, p_y must know the PSK and $\mathcal{M}$ used for encryption. Additionally, the peer must also have a valid copy of D, which was used to generate D'. A valid copy of D may be obtained from either p_{id}^{reg} or p_{id}^{cov} in the BT swarm during the course of normal BT activities. Algorithm 3 and figure 4(b) graphically depict the decryption process.

An important contributor to the strength of this method is the randomly generated one-time mask used to determine the *significant bits* – data payload bits that will actually contain the hidden data. Because it is transmitted using the $\langle bit\_field\rangle$ message during the $\langle hand\_shake\rangle$ protocol, its length in bits will be equal to the number of pieces in the torrent being exchanged by the swarm. Therefore, it is better to use a meta-info file that describes a large set of data when creating the $\mathcal{CC}$, because such a meta-info file will generate a longer mask than one describing a small set of data.

In equation 6, $\langle block\rangle$ is a portion of the data being shared via the torrent, $\langle index\rangle$ specifies the source $\langle piece\rangle$ for the data in the $\langle block\rangle$, and $\langle begin\rangle$ specifies the corresponding $\langle block\rangle$'s offset from the beginning of the $\langle piece\rangle$. In equation 9, $ones(\mathcal{M})$ represents the number of bits in $\mathcal{M}$ set to 1; and $\mathcal{M}[n] = \mathcal{M}[D.size \mod \mathcal{M}.size]$.

$$\mathcal{E} = \frac{ones(\mathcal{M}) \times \left\lfloor \frac{len(D)}{len(\mathcal{M})} \right\rfloor + ones(\mathcal{M}[0], \mathcal{M}[1], \cdots, \mathcal{M}[n])}{2(len(D))} \quad (9)$$

The efficiency $\mathcal{E}$ of SURREAL's $\mathcal{CC}$, as shown in equation 9,is the ratio of the number of significant bits for a given $\langle\mathcal{M}, D\rangle$ to $2 \times D.length$, since the original D must also be transmitted in order for the the receiving peer to be able to decrypt the received message. However, obtaining the original message is simple during regular BT operations as previously noted. With a randomly generated mask, the average case efficiency $\mathcal{E}$ of the $\mathcal{CC}$ will be approximately 0.25, i.e., 2 bits of covert data per byte of D.

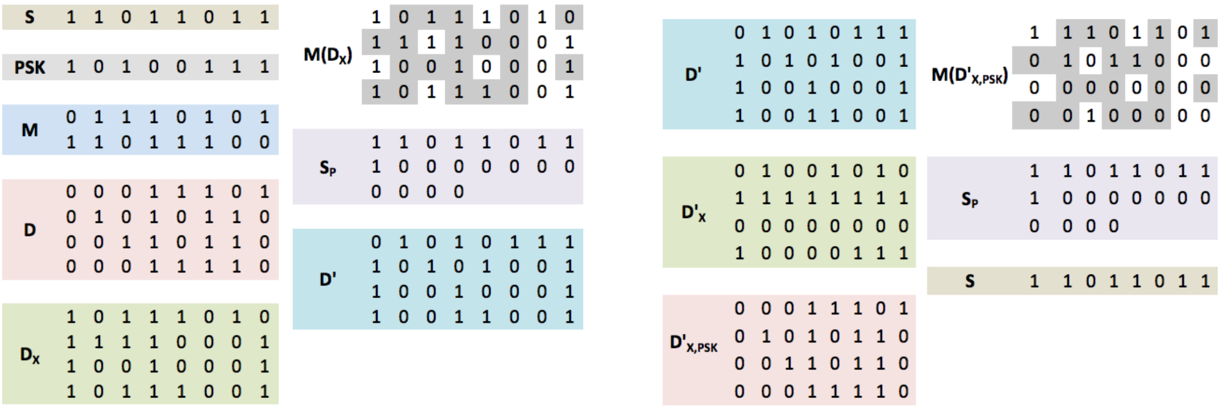
IV. Experiments and Performance Validation

A. SURREAL Implementation

The original BT protocol is simple and flexible enough to incorporate the changes required for this proof of concept. The key requirement for extending the standard handshake into an extended covert handshake is for the client to appear to be operating normally to an outside observer. We chose the open-source *Snark Project* as our foundation for implementing the BT $\mathcal{CC}$. Snark is an open-source client for downloading and sharing BT files.

Although Snark can be used as a regular BT client, it was mainly developed to explore the BT protocol, and therefore it is a simple implementation that follows the published BT specification. The Snark Project itself is developed in java with GNU Compiler for Java (gcj) [12]. A key reason for choosing *Snark* is that, because it is an older implementation of a BT client, it follows the published BT standard, without any modifications or extensions.

The original BT protocol is simple and flexible enough to incorporate the changes required for our proof of concept. While the Snark program is designed as both a standard BT client and server for sharing local files, we once again focus on the *handshake*, *bitfield*, and the *piece* messages for our proof-of-concept implementation of SURREAL.



(a) Encryption

(b) Decryption

Figure 4. Example illustrating En/Decryption of $\mathcal{S}$ with random mask $\mathcal{M}$.

B. SURREAL Comparison with other State-of-the-art

Desimone et al. [5] have the client send covert messages by hiding them in the p_{id} field in an $\langle$ *announce*$\rangle$ request. The sender connects to the tracker's web interface and sends an announce request with a p_{id} containing a covert message. This message is then stored in the tracker's database.

To retrieve this message, the receiver performs an announce request to the tracker as a legitimate p2p client. The server replies to the receiver with a list of clients active in the specified torrent. This reply will contain the covert message. In this CC, $20bytes$ of information can be sent at a time. However, for a more legitimate-looking CC, they recommend reducing it to $12bytes$. Unlike this, SURREAL uses the variable length $\langle$ *piece*$\rangle$ message, with which clients can send significantly larger amounts of encrypted covert data.

Cunche et al. [4] propose an asynchronous CC mechanism based on BT trackers that uses HTTP commands. There is no direct message exchange between the covert peers. Instead, a centralized BT tracker is utilized for storing covert messages. They have analyzed the detectability of CCs by an adversary and show that its detection is unlikely. However, this mechanism appears to have some key limitations. Despite being asynchronous, it has limited recovery window by the receiving peer. Also, CC capacity is very limited – 122 bits per announce request. Most importantly, since none of the colluding peers have supervisory/administrative control of the the tracker server, their mechanism completely hinges on the fact that the tracker's behavior and configuration will remain consistent. In a real world scenario, this is rather a very strong assumption. Covert communication among peers should never be coupled with a system that is not in direct control of the colluding peers. In SURREAL, this is overcome with client side implementation.

C. Experiments and Results

We have compared SURREAL with [4] and [5], as discussed in the previous section. In figure 5, we have presented results comparing the average number of messages required to transmit covert messages of varying sizes between 5KB and 100MB. The results compare number of $\langle$ $piece_{cov}\rangle$ messages that surreal needs, with an eficiency $\mathcal{E} = 25\%$, to completely transfer covert messages of the above sizes, with that of [4] with 12bytes of covert data per announce message and [5] with 122bits of covert data per request.

V. SURREAL – Security Analysis

We now analyze SURREAL's security robustness to common and powerful attacks targeted at p2p networks.

A. Integrity Attacks and Defense

Replay attacks. SURREAL mandates that each peer reply with a $sha1$ of its partner's mask to complete the authenticated $\langle$ $hand\_shake_{cov}\rangle$. The hash value of the mask is intended to provide *confidentiality*, making the *mask* robust to packet sniffing and subsequently brute forcing. Additionally, the freshness of the randomly generated session mask can readily thwart packet replay attacks.

Impersonation attacks. Peer p_x can impersonate another peer p_y initially by simply replaying a previous $\langle hand\_shake\rangle$ and $\langle bit\_field\rangle$ message sent by p_y to a third peer p_z. However, when p_z challenges p_x to decrypt p_z^{mask}, p_x will be unsuccessful since the mask is different from what px replayed. Consequently, the attacker – p_x in this case – will not be able to reply with a valid $\langle piece\rangle$ message to finish the authenticated $\langle hand\_shake\rangle$ with peer p_z.

Modification attacks. Messages subsequent to the initial $\langle hand\_shake\rangle$ are encrypted. A MITM attacker will not be able to decrypt the messages without the decryption key. Even if an attacker knows the PSK and the p_{id} it intends to attack, the p_{id}^{mask} value is unique and randomly generated for each session. Hence, the adversary cannot successfully modify the contents of messages exchanged.

VI. Related Work

With the evolution of of p2p networks, there has been an increasing interest, especially in the BT protocol. Numerous works have explored the use of covert channels in botnets in numerous contexts including – malware distribution [9]; DNS-based command and control [1]; and torrent files-based storage CC [11]. Numerous CC techniques have also been proposed for different network and application protocols.

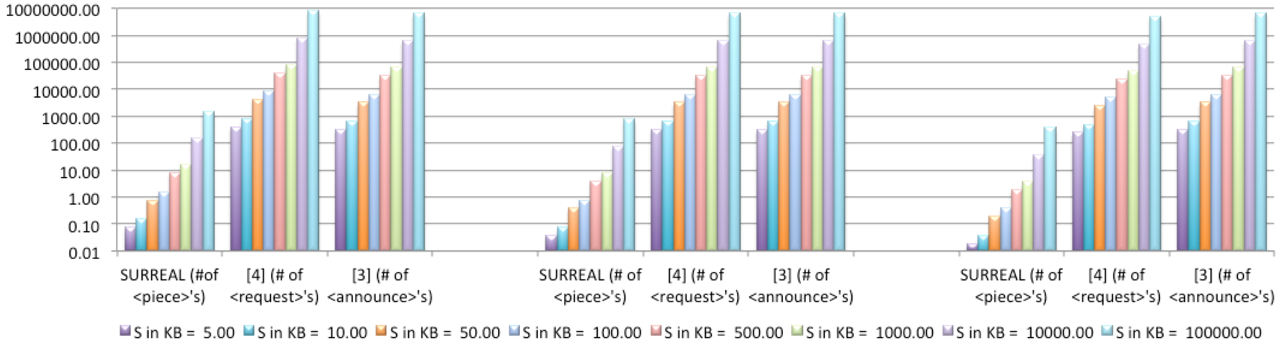


Figure 5. Comparison of SURREAL with [4] and [5] for efficient message transfer.

In [6], Ahsan and Kundur's present a $\mathcal{CC}$ technique that manipulates IPv4 header and packet ordering in an IPSec environment to pass supplementary information. In [7], Eidenbenz et al. demonstrate how BT-like protocols can be exploited to accomplish steganography handshake and broadcast in point-to-point networks. Their methods do not provide security guarantees of authentication, confidentiality, and anonymity.

In [2] authors present a technique to exploit Vuze, a popular file-sharing client. One of the advantages of Vuze is its improved efficiency implemented through a network coordinate system. However, network coordinate systems are inherently ussecure and a malicious peer can lie about its coordinate to appear closest to every peer in the network consequently hijacking every search query.

Goudar et al., in [8] propose a system that uses features of both cryptography and steganography, with a TCP/IP header serving as a steganographic carrier to hide encrypted data. Yaroshkin et al. [13] have presented a detailed discussion on covert communication in p2p architecture. While there solutions have good methods for establishing covert communication, they do not provide the most important requirement - peer authentication.

In [11], Li et al. propose "Stego-Torrent" for torrent files which uses two approaches to hide data – 1) letters in the URL are changed according to the secret message, and 2) some optional fields in the torrent file are reused. On the other hand, "StegTorrent" [10] is another steganography scheme developed over BT. In this approach, the order of data packets in the exchange protocol is changed to create the $\mathcal{CC}$. The basis for development of this scheme is one-to-many packet transmissions in the Torrent protocol, with the header support for packet numbering and retrieval of packet sequence. Neither of the above two methods provide peer authentication, a key requirement for covert communication.

VII. Conclusion and Future Work

In this paper, we have presented SURREAL, which is an extension of the standard BT p2p protocol. SURREAL extends BT's standard 3-step $\langle\ hand\_shake\rangle$ to a 6-step $\langle\ hand\_shake_{cov}\rangle$ secure peer authentication service and robust peer anonymity with a one-way function.

In designing SURREAL, we primarily focus on three standard BT message types extending them to support $\mathcal{CC}$ – $\langle\ hand\_shake_{cov}\rangle$, $\langle\ bit\_field_{cov}\rangle$, and $\langle\ piece_{cov}\rangle$. We have implemented SURREAL's proof-of-concept using the open-source *Snark*, and validated its performance by comparing it with [4] and [5]. The results confirm that SURREAL outperforms [4] and [5] by a significant margin.

References

[1] Patrick Butler, Kui Xu, and Danfeng Daphne Yao. Quantitatively analyzing stealthy communication channels. In *Applied Cryptography and Network Security*, pages 238–254. Springer, 2011.

[2] Eric Chan-Tin, Victor Heorhiadi, Nicholas Hopper, and Yongdae Kim. Hijacking the vuze bittorrent network: all your hop are belong to us. *Information Security, IET*, 9(4):203–208.

[3] Arun Chokkalingam and Firasath Riyaz. Bittorrent protocol specification v 1.0.

[4] Mathieu Cunche, Mohamed Ali Kaafar, and Roksana Boreli. Asynchronous covert communication using bittorrent trackers. In *High Performance Computing and Communications, 2014 IEEE 6th Intl Symp on Cyberspace Safety and Security, 2014 IEEE 11th Intl Conf on Embedded Software and Syst (HPCC, CSS, ICESS), 2014 IEEE Intl Conf on*, pages 827–830. IEEE, 2014.

[5] Joseph Desimone, Daryl Johnson, Bo Yuan, and Peter Lutz. Covert channel in the bittorrent tracker protocol, 2012 - (Accessed: 21 April 2014).

[6] Bret Dunbar. A detailed look at steganographic techniques and their use in an open-systems environment, 2002 - (Accessed: 21 April 2014).

[7] Raphael Eidenbenz, Thomas Locher, and Roger Wattenhofer. Hidden communication in p2p networks steganographic handshake and broadcast. In *INFOCOM, 2011 Proceedings IEEE*, pages 954–962. IEEE, 2011.

[8] RM Goudar, SJ Wagh, and MD Goudar. Secure data transmission using steganography based data hiding in tcp/ip. In *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, pages 974–979. ACM, 2011.

[9] Daryl Johnson, Peter Lutz, and Bo Yuan. Behavior-based covert channel in cyberspace.

[10] Pawel Kopiczko, Wojciech Mazurczyk, and Krzysztof Szczypiorski. Stegtorrent: a steganographic method for the p2p file sharing service. In *Security and Privacy Workshops (SPW), 2013 IEEE*, pages 151–157. IEEE, 2013.

[11] Zishuai Li, Xingming Sun, Baowei Wang, and Xiaoliang Wang. A steganography scheme in p2p network. In *Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP'08 International Conference on*, pages 20–24. IEEE, 2008.

[12] Mark Wielaard. The hunting of the snark project - bittorrent application suite, 2003 - (Accessed: 21 April 2013).

[13] Fedor V Yarochkin, Shih-Yao Dai, Chih-Hung Lin, Yennun Huang, and Sy-Yen Kuo. Introducing p2p architecture in adaptive covert communication system. In *Internet, 2009. AH-ICI 2009. First Asian Himalayas International Conference on*, pages 1–7. IEEE, 2009.