Secure Instant Messaging Using OTR (Off the Record) Method

[1]Ashutosh Jainvi, [2]Shivam Gupta, [3] Sanchit Goel
Department of Information Technology, EN & ME,
Ajay Kumar Garg Engineering College(A.K.G.E.C). Ghaziabad, India
[1]ashujainvi@gmail.com, [2]sgshivamgupta2419@gmail.com ,[3]sgoelmzn@gmail.com

***Abstract:*— In this paper we have discussed about the encryption of the instant messaging service using the "Off The Record" method(OTR) method technique. The primary motivation behind the protocol was providing deniable authentication for the conversation participants while keeping conversations confidential, like a private conversation in real life. An urge to provide OTR as secure messaging tools in several messaging application has been shown in the research paper.**

***Keywords:* protocol, cipher text, authentication, encypryt, decrypt**

I. INTRODUCTION

It is a cryptographic protocol designed to provide strong encryption of your communications. OTR is an encryption protocol for real time chat. OTR (Off-the-record) is a protocol that allows users of instant messaging or chat tools to have conversations that are secure. Off-the-Record Messaging (OTR) is a cryptographic protocol that provides encryption for instant messaging conversations. OTR uses a combination of AES symmetric-key algorithm with 128 bits key length, the Diffie–Hellman key exchange with 1536 bits group size, and the SHA-1 hash function. In addition to authentication and encryption, OTR provides forward secrecy and malleable encryption.

- Forward secrecy: Messages are only encrypted with temporary per-message AES keys, negotiated using the Diffie-Hellman key exchange protocol. The compromise of any long-lived cryptographic keys does not compromise any previous conversations, even if an attacker is in possession of cipher texts.

Deniable authentication: Messages in a conversation do not have digital signatures, and after a conversation is complete, anyone is able to forge a message to appear to have come from one of the participants in the conversation, assuring that it is impossible to prove that a specific message came from a specific person. Within the conversation the recipient can be sure that a message is coming from the person they have identified.

II. DESCRIPTION

Ask if you can talk 'off the record' then tell them something that appears to be new information for them. You can also ask them about something that they may be holding back.

Another method is simply to add 'off the record' in the middle of a conversation and then continue with what you want to say, assuming that the other person will not repeat what you tell them.

You can even ask for an entire 'off the record' meeting where you might for example exchange information or give a presentation about the overall business context. The negotiation meetings are then held separately to these.

Variations of 'off the record' include:

- 'As an aside...'
- 'Confidentially...'
- 'Don't tell anyone, but...'
- 'Truthfully,...'

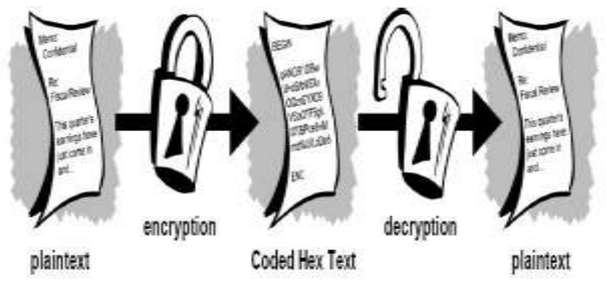
Do not tell them anything that would cause you a problem if they brought it up later.

III. EXAMPLE

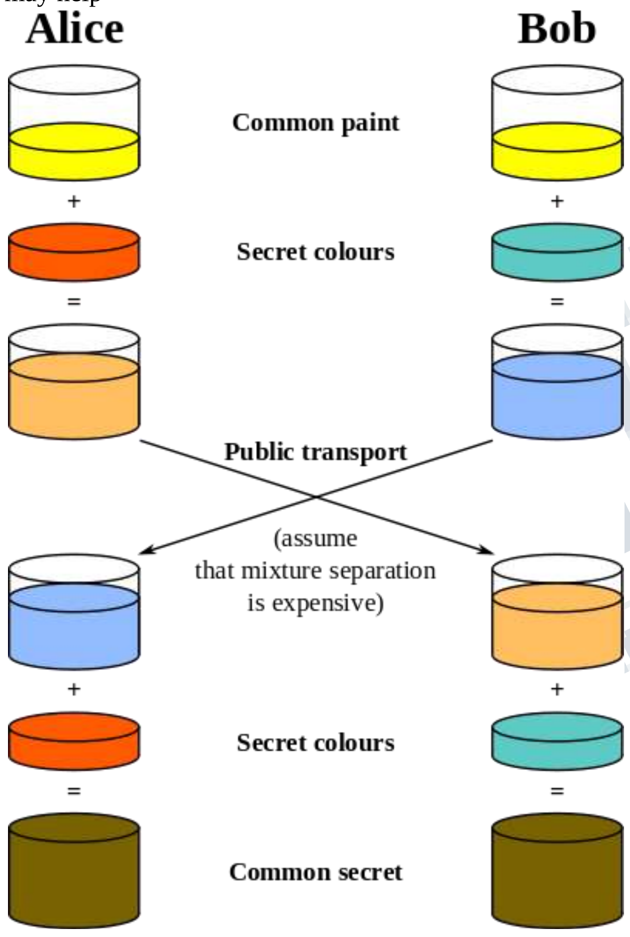
Strictly off the record, is Michael the only person who can approve this?

Confidentially, I'm not happy with it myself, but I still have to go through with it.

Let's go and have a cup of coffee. I want to give you some background information that may help.



may help



OTR uses the Diffie-Hellman key exchange algorithm. In short, two parties are able to exchange secret keys in such a way that they can derive a shared AES (Advanced Encryption Standard) key that is impossible for an eavesdropper to decipher. The following diagram conveys the general idea. Of course real world Diffie-Hellman doesn't use paint, but rather some complex math that relies on the difficulty of discrete logarithm problems.

IV. DISCUSSION

Of course nothing is really off the record, and in many negotiations you can never fully trust the other party. However, it can be very useful. When you place something off the record, for example, you imply that you are trusting them and, by reciprocation, they should trust you. This also puts them in an awkward position, where they are socially obliged to you not to repeat things, yet still are obliged to their employer or other associates. If they accept your obligation, you will have moved them towards you, and they may hence be more open to other suggestions.

Going off the record is also useful when things are getting stuck in the negotiation and a bit of honesty can help get things back into motion. By declaring that you are being truthful or open, you make what you say next more credible and also suggest you are willing to give as well as take.

V. OTRENGINE

Definition:

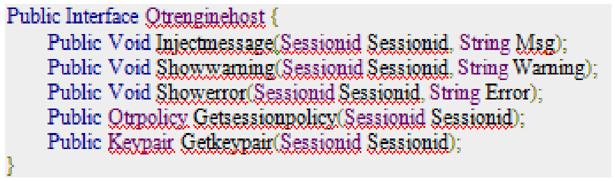
```
Public Interface Otrengine {

    Public String Transformreceiving(Sessionid Sessionid, String Content);
    Public String Transformsending(Sessionid Sessionid, String Content);

    Public Void Startsession(Sessionid Sessionid);
    Public Void Endsession(Sessionid Sessionid);
    Public Void Refreshsession(Sessionid Sessionid);

    Public Void Addotrenginelistener(Otrenginelistener L);
    Public Void Removeotrenginelistener(Otrenginelistener L);
    Public Sessionstatus Getsessionstatus(Sessionid Sessionid);

    Public Publickey Getremotepublickey(Sessionid Sessionid);
}
```

OtrEngine defines the Off-the-Record functionality and OtrEngineImpl is an implementation of OtrEngine. It is used to:

- Manage the encrypted session (startSession, endSession, refreshSession)
- Encrypt outgoing & decrypt incoming messages (transformSending, transformReceiving)
- Get the session status (getSessionStatus) and inform about session status changes (through the OtrEngineListener).

VI. OTRENGINEHOST

Definition:



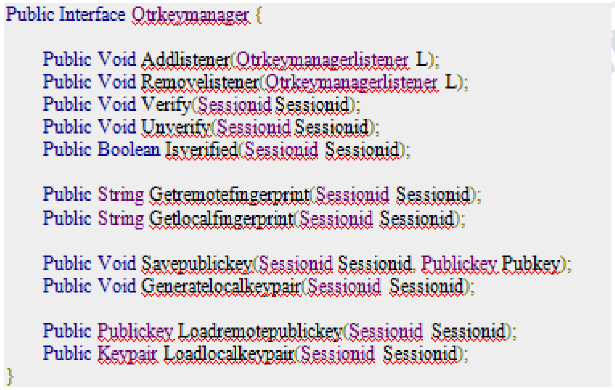
```
Public Interface Otrenginehost {
    Public Void Injectmessage(Sessionid Sessionid, String Msg);
    Public Void Showwarning(Sessionid Sessionid, String Warning);
    Public Void Showerror(Sessionid Sessionid, String Error);
    Public Otrpolicy Getsessionpolicy(Sessionid Sessionid);
    Public Keypair Getkeypair(Sessionid Sessionid);
}
```

OtrEngineHost defines the functionality the host application must provide to OtrEngine. You have to pass an OtrEngineHost as an argument to the OtrEngineImpl constructor. OtrEngineHost is used by OtrEngine to:

- Inject messages, for example during the Authenticated Key Exchange.
- Notify the host application about warnings and errors.
- Get the session Off-the-Record policy.
- Get the session long-term KeyPair.
- In this method implementation you will want to call OtrKeyManager.loadLocalKeyPair() and if that returns null, call OtrKeyManager.generateLocalKeyPair().

VII. OTRKEYMANAGER

Definition:



```
Public Interface Otrkeymanager {

    Public Void Addlistener(Otrkeymanagerlistener L);
    Public Void Removelistener(Otrkeymanagerlistener L);
    Public Void Verify(Sessionid Sessionid);
    Public Void Unverify(Sessionid Sessionid);
    Public Boolean Isverified(Sessionid Sessionid);

    Public String Getremotefingerprint(Sessionid Sessionid);
    Public String Getlocalfingerprint(Sessionid Sessionid);

    Public Void Savepublickey(Sessionid Sessionid, Publickey Pubkey);
    Public Void Generatelocalkeypair(Sessionid Sessionid);

    Public Publickey Loadremotepublickey(Sessionid Sessionid);
    Public Keypair Loadlocalkeypair(Sessionid Sessionid);
}
```

OtrKeyManager defines the key management functionality and OtrKey Manager Impl is an implementation of OtrKeyManager. It is completely decoupled from OtrEngine, so you can implement an alternative OtrKeyManager that stores/loads keys in a specific way. It is used to:

1. Verify/Unverify sessions (through their respective methods).
2. Get the session verification status (is verified) and inform about verification status changes (through OtrKey Manager Listener)
3. Get fingerprints for the host application to display (get Remote Fingerprint, get Local Fingerprint)
4. Save public key of remote parties. The host application has to call this method when a session status changes to encrypted. (This can be improved; it shouldn't be the host application responsibility to save the public key).
5. Load local key pairs or generate new KeyPairs (this is usually done in OtrEngineHost.getKeyPair(), as described above).
6. Load remote public keys. When a session goes encrypted the host application will want to check if the remote public key equals the one stored by the OtrKeyManager, and if yes to check if that key is verified or not.

VIII. FUTURE ASPECTS (CONCLUSION):

The OTR Method can be as successful as the End to End method Used in many messenger services nowadays. Most IM platforms are built for consumers where security takes a lower priority With privacy being the key issue in the server technologies, the OTR method can be best used to restrict external server control as the privacy is being constrained between two authenticated users. Such a terminology would help in justifying the cipher techniques used in social messaging services nowadays.

REFERENCES

Book

[1] B.H Briefings, Off-the-Record Messaging Or, When not to use PGP (2002)

Others

[2] Changing Minds: http://changingminds.org/disciplines/negotiation/tactics/off_record.htm

[3] http://www.bitcoinnotbombs.com/beginners-guide-to-off-the-record-messaging/