WebChaff: Chaff generation for Confidentiality of Instant Messenger conversations.

Faraz Shaikh

Carnegie Mellon University

fshaikh@cs.cmu.edu

***Abstract*— Chaffing and winnowing is a technique for achieving confidentiality for information without encrypting it. It works by introducing sufficient noise in the information so that an untrusted party cannot discern between the noise and real information. The lynchpin of this method is the ability of generating good quality noise that completely blends in with the transmitted information. We examine a practical method for using the Chaffing and Winnowing for securing Instant messenger (IM) conversations. The key contribution of this paper is the introduction of the "WebChaff" method to generate good quality noise for IM conversations. "WebChaff" generates chaff for IM conversation using the random results from Internet search queries for words in IM message. Also it will serve as practical demonstration of using the Winnowing and Chaffing technique for achieving confidentiality.**

***Index Terms*—instant messaging, privacy, chaffing and winnowing.**

I. INTRODUCTION

INSTANT messaging is one of the most popular ways of casual communications between computer users. IM is mostly provided as a service by ISPs that link a user's email account to his IM identity. Most IM service providers do not provide for encryption or other form secure transmission for IM messages. As a result IM messages are usually transmitted in clear text over the network making them highly vulnerable to eavesdropping. Encrypting IM messages will definitely achieve the goal of confidentiality. However, in this paper we investigate 'Winnowing and Chaffing' as alternative approach to encryption for achieving IM confidentiality. In the process we also describe the properties of a good chaffing scheme for IM messages. Also described are some of the common pitfalls of word level chaff generation. The remaining paper is organized as follows we discuss the related work associated with the CAW method in Section III, in this section we also have a brief look at CAW method itself an some of its known applications, Section IV describes the threat model we aim to protect against in the IM scenario, Section V describes the WebChaff method of generating chaff for IM messages. We conclude with conclusions and results drawn from an initial prototype of the WebChaff method.

Manuscript received October 9, 2001. (Write the date on which you submitted your paper for review.)

II. MOTIVATION

The Chaffing and Winnowing [1] (CAW) method was first described by Ronald Rivest for achieving confidentiality without encryption. Rivest's interest in alternatives to encryption was partly due to the debate that should the law enforcement agencies be given backdoor entries to access plain text of encrypted messages. One of the reasons CAW gets less attention is because it seems to be redundant in scenarios we can use encryption. CAW turns out to be a good solution in scenarios where encryption is not feasible. Again, application of CAW is challenging because it presents multiple challenges namely generating high quality chaff and tradeoff like amount of chaff to be used VS efficiency. By generating a very large and varied amount of chaff we have higher chances of getting strong privacy but at the cost of efficiency. A good implementation should achieve maximum privacy for information with minimum amount of chaff added as an overhead.

III. RELATED WORK

Rivest in his initial description [1] about the Winnowing and Chaffing method describes CAW as an alternative for attaining privacy without encryption or stenography. Here in this paper we discuss a brief summary of Rivest's description of the CAW method, this description is in no way the complete description of the original CAW method described by Rivest. Readers interested in getting the details and minutiae of the CAW method are suggested to read the original description of CAW by Rivest. The summary is added in this paper just for the sake of completeness.

A. *The Basic CAW algorithm*

Rivest describes a simple algorithm to implement chaffing where the sender and receiver share a secret session key. Further communication between the sender and the receiver is achieved by the sender adding arbitrary chaff/noise to with the original messages prior to transmission. All components of the transmitted message including the chaff are augmented with a machine access code (MAC) which is generated using the shared key and the message component itself. The chaff components however are added with a random MAC. So the transmitted can be described as

Message A is a random Interposition of
{
(Information Components + Valid MAC)
And
(Random Chaff Component + Random MAC)
}

Fig 1. Contents of a message transmitted using the CAW method.

At the receivers end the receiver re-computes the MAC for every received component using the shared key and drops the components for which there is a mismatch in the computed and transmitted MAC. An eavesdropper sees the message along with the chaff but is unable to discern between the original information contents and the chaff because he doesn't know the key used to generate the MAC for original data. Ideally, for the eve's dropper every component of the transmitted message is should seem equally probable of being part of the original message. Rivest suggests using the Diffie Hellman key exchange [2] algorithm for exchanging initial keys and using the using the HMAC-SHA1 [3] for computing the MAC.

B. *Factors affecting the degree of confidentiality provided by CAW.*

The basic CAW algorithm seems to be a very viable solution for achieving privacy only through initial authentication. The exact degree of confidentiality provided by the CAW algorithm depends on the following factors.

a. Quality of the MAC algorithm
b. How we break the original message into small components
c. Generation high quality chaff.

Rivest lays down the qualities of a good MAC algorithm to be used for CAW method. These qualities include appearance of the MAC algorithm as a random function to an adversary. Rivest also states that MAC algorithm used should not leak information about the message for which the MAC is generated. These qualities ensure that there is no relation that can be discerned by the adversary between random MACs for chaff and a valid MAC's for the original message.

The breakup of the original message into small components poses a quality VS efficiency tradeoff. With breaking the message into very small components we can achieve high confidentiality but efficiency decreases the overhead for MAC transmission increases. This is because each broken up component needs its own separate MAC.

If the introduced chaff itself has a pattern that can be discerned by an adversary the entire basis of the CAW method for providing confidentiality falls apart. Bellare et.al provide a formal analysis of the security/confidentiality provided by the CAW method in [4] and also suggested improvements to Rivest's basic algorithm. However, over the time very few if any applications have used CAW as a solution for providing confidentiality. One such application is described by Kiong et al in [5] where the authors describe a secure an electronic voting scheme using CAW.

IV. THREAT MODEL

We are trying to apply the CAW method as a solution for providing confidentiality of IM messages. It helps us to clearly state the attacks our solution is going to provide protect ion against.

According to Saltzer et. al in [7] Security attacks can be broadly classified as.

a. Attack on confidentiality via Interception
In the IM scenario it means an adversary can gain information to which he is not entitled, by intercepting IM conversations. This interception can by sniffing live IM traffic over the network or by compromising user account and going through the log of old message stored by the IM server. The second type of interception is becoming prevalent due to cases where chat logs are being used evidence in criminal proceedings.

b. Attack on availability via Interruption.
In the IM scenario it means denying legitimate user access to IM service by mounting any type of a DOS attack.

c. Attack on integrity via modification.
In the IM scenario an example of this type of attack would be modification of text messages as they are pass through the network.

d. Attack on authenticity.
In the IM scenario it would mean to fake the identity of an IM user. Handling this is typically in the domain of the IM service provider and we do not worry ourselves with it in this paper.

We specifically focus on protection against attack on

confidentiality via interception. Our solution protects against both an eavesdropper trying to intercept live IM communication between two users, and also attacks on confidentiality by gaining access to users chat logs.

Attack on integrity via modification is also handled to some extent by our solution. The CAW method makes it difficult for the attacker change the transmitted message by adding/deleting /updating information components. The addition and updating case is handled because an attacker cannot generate the correct MAC for newly added/updated message components. The attacker can also change to message by deletion of components from transmitted message, but again selecting correct components for deletion would be difficult. In some cases the attacker may decide to alter the transmitted message by deleting large parts of the transmitted message. This will include message parts containing both the chaff and the original message. Such an alteration to the transmitted message can be readily detected by end users because of the resulting illegibility of altered messages.

V. THE WEBCHAFF METHOD

In this section we first describe the basic problems statement for generating word level chaff for IM messages and the requirements for an acceptable solution. Then we describe the 3 solutions for the problem statement and compare them on the basis of whether they meet the requirements set forth by us for the ideal solution.

A. *Basic Problem Statement for IM chaff generation.*

Build an algorithm such that given an IM message X convert it into a new message X`. In an ideal algorithm X' should have the following properties.

P1. The words of message X should be a part of the message X'. *(Strongly required)*
P2. One should not be able to discern/extract X from X' simply by having access to X'. (*Strongly required)*
P3. All words in X should appear in X' in the same order as they as they appear in X. *(Can be relaxed)*
P4. To be practical the scheme should support major all major international languages. *(Strongly required)*
P5. X' should be as small as possible. *(Strongly required)*

The ideal algorithm should have the property.

PS1. It should generate distinct X' for the same input X for different of the invocations of the algorithm. (This is a very important property and is somewhat of an analogy to property of the PRNG where we don't want sequence to repeat in the output of a PRNG). (Strongly required)

PS2. The algorithm should not in the process of generating X' compromise or disclose the contents of X. (Strongly required)

With respect to CAW algorithm X' is the message that will be transmitted. Property P1 ensures that original message can be reconstructed at the receivers end. Property P2 ensures that an attacker cannot compromise confidentiality simply by looking at the message. Property P3 can be relaxed but that would complicate the reconstruction for X from X' and would require additional sequencing information to be added in X' before transmission. Properties P4 and P5 are properties required for the method to be practically usable and deployable across the multitude of IM users.

Once we have a solution to this problem X' can act as the chaffed message to use as part of the CAW approach to achieve confidentiality of IM messages. This also means that we are using a word in the sentence as the basic granularity for breaking the IM message into components.

Input:
Alice-> "Today is a good day"

After chaff generation should look like.

Output.
Alice-> India Today is a popular magazine. This is not a good introduction. A day has 24 hours.

Example 1. A totally made up ideal solution instance to the chaff generation problem.

One can come up with multiple X' that satisfy all properties P1-P5 for a given message X. This brings us to the desired properties of the chaff generation algorithm PS1. This property avoids an attacker from building a large table for translating every X to X' and then doing a reverse lookup to find a particular X from X'. If we consider the chaff generation algorithm as cryptographic hash function, property PS1 forces the solution to preimage resistant, second preimage resistant and collision resistant. (We would like to admit that analogies of the solution requirement PS1 to the requirements of PRNGs and cryptographic hash functions are extremely weak).

PS2 loosely translates to the notion that to protect an entity X we should protect X as well as the mechanism that protects X. This concept is very well explained in Saltzer's paper about protecting information is computer systems in [7].

Now with a clear crisp problem definition and requirement being set we can present and evaluate algorithms for solving the problem. We study three algorithms for generating chaff of IM messages.

B. *Basic WebChaff algorithm*

The algorithm is briefly explained below for generating webchaff using the internet search results.

```
Algo: Webchaff
Input: IM message
Outuput: chaffed IM message

String ChaffedMessage;
1. For each word in message.
2.     Queryresults = webSearch(word);
3.   Find a random result from query set
     that contains the 'word'.
4.   Append the random search results to
     ChaffedMessage.
5.  End for.
6.   return ChaffedMessage.
```

The algorithm for WebChaff is self explanatory and quite simple where each word in the IM message becomes part of the chaffed message in the form random search query results.

Some considerations while implementing WebChaff include selecting a bias for unpopular search results and encoding of words that result in 0 search results.

For selecting unpopular search results we request for higher numbered search pages from the search provider. Unpopular search results may not contain the word as part of the result description in verbatim. In this case we make finite attempts to search for the word in adjacent search results. We bail out after if we cannot find a result that contains the IM message word.

If encoding of a word fails due to it not being a part in verbatim of some query results one can try to merge this word along with adjacent words in the original IM message. For example if the original IM message is "Gift for Dexter", and we fail in finding suitable chaff for the word "for" we should try search for the word "Gift for" instead.

Well some words simple don't encoded using the algorithm if the search engine results no results for even after we try merging words. In this case we can either drop the word from the original sentence or let it be transmitted without the chaff. Dropping the word a notifying the user about is a good idea because adding single word in a chaffed message makes them stand out from the rest of the message.

C. WebChaff algorithm with securing the web search query channel.

This is an optimization for the basic Webchaff algorithm. This addresses the problem of the original message being transmitted in plain text without the chaff to the search engine. This meant that an eavesdropper can compromise privacy by monitoring the search requests we send to the search providers.

To avoid this we chaff the search stream itself. Words to be used in chaffing the search stream are part of and initial entropy word pool. This initial pool of words can be selected form static corpus of words, can be entered by the user or they could be part of the user's old IM conversations. This initial pool of chaff words is then constantly updated by adding new conversation context specific words to it from search results themselves. This forms the feedback that keep the word pool updated with the words specific to the context of the IM conversation.

While the above method seems straight forward, its effectiveness depends upon selection of feedback words used to update the entropy word pool. Unfortunately our initial results show that it's difficult to come up with a method that keeps the pool updated with words specific to the context of the IM conversation. Although we tried only a few methods for updating the entropy pool we could find out some updating methods were better than others. We also identified some pitfalls that should be avoided.

Updating the entropy word pool every single time we generate a web search is a bad idea. This means that the pool never converges to the topic of conversation. Instead one should choose only a few result queries as candidates for containing words for entropy pool updating. Also, selecting all words from a single search result for pool update performs badly as compared to selecting words from many query results. If we include all words from a single search query we see that the pool is quickly polluted with common words like "the" "a" and "of".

We tried out only a few alternative methods to update the word pool; the best method was where we used random sampling of query results as candidates for updating the word pool. Once the search result set is are selected as candidate for updating the word pool we update the pool by taking words adjacent to search word from all of the results in the result set. This made sure that cache is not updated frequently and also that it's not polluted with common words. Below is an example of words selection for updating the word pool.

```
Web Search: Hammerschlag.
…
1. Motivational Keynote Speaker Dr.
Carl  HammerSlag,..
2.  Michael  Hammerschlag  is  a
journalist …

WebSearch: Meeting
1.  ...in web conferencing, online
meetings,     web     conferencing
service..
2.  ...in a meeting, two or more
people ...
```

Example 1. Selection of words for updating the word pool.

Example1 present the results for the words "Hammerschlag" and the words marked bold are the words that should be used for updating the word pool. They are the words that are adjacent to the searched word 'Hammerschlag'. The gist for

selecting the words for updating the word pool is simple and can be concluded as.

- Select only words specific to the context of the IM message.
- Avoid updating the pool too frequently and instead do a random sampling for selection of words.

VI. RESULTS

We implemented a prototype that lets us generate web chaff for a give message. The prototype used MSN Live search [8] for search results. The prototype also includes chaffing of the web search word as explained in this paper in section V. Updating the entropy word pool is also implemented using the methods described in section V. The code for this prototype is written such that different tunable parameter like the bias for selecting unpopular query results and frequency of updates to the entropy pool are easily configurable.

We get very good results for chaff for the original message. Some of the widely accepted private information gets chaffed nicely using web chaff. Example 1 shows the generated webchaff for an IM message consisting of the hypothetical credit card number.

IM:
5999 4000 4994 1343

Chaffed To:
ISS X-Force Database: encipher-recover-operator-cards(**5999**)**:** encipher operator card set "no-recovery" option The Kestrel **4000**, a hand-held wind meter and hygrometer, …. and more.ISS X-Force Database: ipfilter-firewall-race-condition(**4994**): IPFilter firewall race condition ... Description: IP Filter versions 3.3.15 and 3.4.3 and earlier are vulnerable to a ... RFC 1343 - A …. Mechanism for Multimedia Mail Format Information ... RFC **1343** - A User Agent Configuration Mechanism …

Words Selected for Cache Updating
Kestrel - nCiper – cards – RFC - A

Encoding time:00:00:05.4077760

Example 1: Webchaff for a hypothetical credit card number.

Example 1 shows the encoding for a hypothetical credit card number which many would believe is private information. As you can see the numbers of the credit card get mixed with parts of other context like port numbers and RFC numbers. Similarly other information like individual names and address get chaffed quite nicely.

IM:
John Doe lives at 15712 Carnegie Mellon

Chaffed To:
JohnBingham.com "The miracle isn't that I finished. The miracle is that I had the courage to start. "The **Doe** Fund, Inc. … Devinn Lane and her six roommates **lives** …D-**15712** 2-way Eyebrow board ..broadcasts and performance schedule for the resident Lincoln … Andrew **Carnegie** was born in Dunfermline…

Encoding time:00:00:06.4192304…

Words Selected for Cache Updating
Andrew - Roomates - 2-way - Bingham.com

Encoding time:00:00:05.4077760

Example 2: Webchaff for a hypothetical creadit card number.

Example 2 shows the encoding of message that contains a proper noun and an Address. As you can see first names usually get mixed and matched with last names and last names get matched with the first names in the Webchaff. Also numbers in addresses get matched with any of the numerical datum over the net in this case the version number of some product.

Users generally don't like slow delivery of IM messages. This is because of the interactive nature of the IM conversation. For the Webchaff method to be usable it should perform the encoding of messages in quick time. For this we decided to measure the time required to generate chaff for a sentence depending upon the number of words in that message.

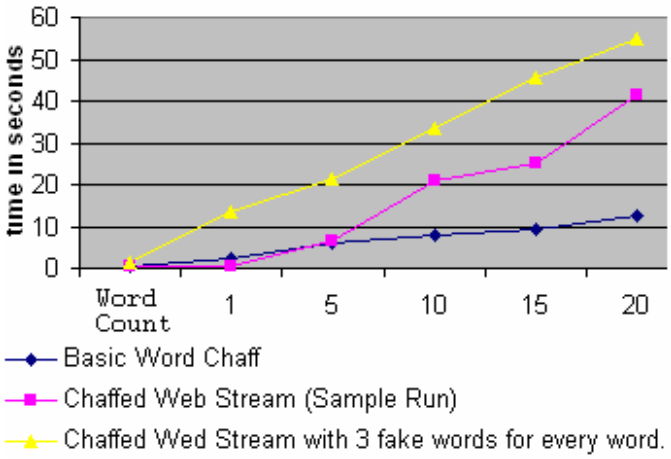

Fig 1. Timing results for generation web chaff for IM messages.

Fig 1 shows the time required for encoding messages with N number of words. The number of words in a sentence is represented along the X axis. X axis shows the time in seconds. This time heavily depends on the net connection used to get results for a web query. The basic chaff method scales quit well as the number of words in a message increase.

The WebChaff method with chaffing of the web search stream does shows requires significant time for coding as number of words in the message increase. For this we measure the worst case coding time where for coding every word we issued 3 dummy chaff words search before the searching for the actual word. This is the worst case times for generating Webchaff with 3 fake web searches per IM word case. In reality, we employ the random number to get the number of fake search to be performed for each IM word. The upper bound for the number of words can be fixed. The graph in figure 1 show a time required a sample run with 0-3 fake word queries per IM word in the "Chaffed Web Stream Sample Run" data set.

VII. CONCLUSION

Connecting over the internet to generate chaff for messages generation presents additional challenges to secure the contents connection itself. The chaff generated by search web pages gives good security for the contents of the IM transmitted messages.

Although, the entropy word pool method for described in this paper that chaffing the search stream is not that effective. This is mainly because this stream is generated by randomly selecting random words from the entropy pool. This random selection of words does not logically show up in sentences but the words in an IM message are logically connected as part of a sentence. This difference makes is easier to guess the transmitted messages as part of the web search. We have suggested some method to overcome the problems posed for selection of messages for updating the entropy pool.

The dictionary method suffers from the same problem of as of chaffing the web stream. Randomly selected words from the dictionary do not usually appear in logical sentences and this makes this easily distinguishable from words in the original sentence.

Thus it would be fair to conclude that word level chaff generation for IM messages is best done using selecting random results from the internet search queries and intelligently updating the entropy word pool.

REFERENCES

[1] R. Rivest, "Chaffinng and winnowing: Confidentiality without encryption," http://theory.lcs.mit.edu/~rivest/publications.html. (visited Dec-2007)

[2] New Directions in Cryptography W. Diffie and M. E. Hellman, IEEE Transactions on Information Theory, vol.

[3] Krawczyk, H., Bellare, M., and R.Canetti, "HMAC: Keyed-hashing for Message Authentication," RCF2104, February 1997.

[4] The Security of Chaffinng and Winnowing. Mihir Bellare, Advances in Cryptology ASIACRYPT '2000

[5] Kiong, N.C. Samsudin, A. Incoercible secure electronic voting scheme based on chaffing and winnowing, 9th Asia-Pacific Conference on Communications.

[6] Jennings, R.B., III Nahum, E.M. Olshefski, D.P. Saha, D. Zon-Yin Shae Waters, C. A study of Internet instant messaging and chat protocols, IEEE Network Volume 21.

[7] Jerome H. Saltzer, Michael D. Schroeder, the Protection of Information in Computer Systems.

[8] Microsoft Live Search API,msdn.microsoft.com/live/ (visited Dec-2007)