

Hello, and Welcome. My name is Tom Ritter, and I work for iSEC Partners. If you don't know who Zax is, you will by the end of this talk.

This talk is about an anonymity network that was started in the fledgling days of the Cypherpunk era – the early 1990s.

This book hadn't even come out yet – this is the second edition. But this is the first edition, and it had come out, and the US had ruled you while you could export the book itself, you couldn't export the floppy disk with the source code. The United States government was actively investigating Phil Zimmerman for violating the Arms Control Export Act, for making the first few versions of PGP available. Dan Bernstein and the toddler-aged EFF went on the offensive taking the US government to court and suing over the export controls on cryptography. Another group of people ultimately printed the source code for PGP, exported the book to Europe, scanned it in, and OCR-ed it in 1997 releasing a version of PGP that bypassed export controls

Alt.Anonymous.Messages was forged in the heyday of the cypherpunks, and really, overall, has changed very little in the intervening decade since it was last shaped in any major way.

And in that decade, what we have seen is a monumental focus of the nations spy

agencies on not what was thought to be the most critical piece of information to encrypt – the content itself. But instead....



The people who know won't talk, and the people who talk don't know. But the leaked court orders require Verizon to turn over call records local and abroad. Now, I'm talking here, so I don't know anything and am just speculating – but the most straightforward thing to do with this data is to build communication graphs. Analyze the metadata, looking for patterns. Identify people of interest, and figure out who they talk to.

The metadata around an encrypted channel tells volumes.



SSL, is the most widely used encrypted channel on the internet today. And even ignoring the numerous attacks we've seen on it in the past few years, and even ignoring how it breaks just about every cryptographic best practice there is – there is a wealth of information you can learn from observing an SSL session. There are protocol level leaks – SSL says a lot about what type of client you're using, and it's version. It also includes what you think the local time is.

SSL	
lf Who When	(You're communicating) (It's with) (You're doing it)
How much	ן (You've sent)

But from an information theoretic perspective, an adversary can see that you're sending packets, and communicating. That seems obvious, of course they can – but it's important to bear in mind for the future. Ideally, our adversary wouldn't even know if we're communicating.

Secondly, SSL makes no attempt at hiding who you're talking to. So the fact that you're on Facebook is straightforward.

And similarly, the adversary knows when you're on Facebook. And when you are sending data and when you are receiving data. The resolution on this goes down literally to the microsecond.

So they know exactly when, and they also know exactly how much data you receive. SSL doesn't have any real padding, and I don't know of any website that adds variable padding to frustrate length analysis.



So let's talk about Tor. Tor is an implementation of Onion Routing, where you pass messages along a chain, each node peeling off a layer of encryption, until an exit node talks to the intended destination. The destination responds, and it's routed back.



Onion Routing specifically aims to disguise Who is talking. An adversary observing you can't see that you're talking to a website (or a service), and an adversary observing that website or service can't see who is talking to that website.

But it doesn't stop an adversary from knowing you're talking to someone, knowing when you're talking, and how much you're saying.

Tor doesn't really do padding what little it does is not intended to be a security feature. Tor explicitly leaves _out_ link padding.



And if you stayed through Runa's talk, you know that Tor cannot protect you if the adversary can see the entire path of the circuit.

Let's say hypothetically, New Zealand, Australia, the US, Canada, and the UK were to, say, conspire secretly on some sort of spy program.

Well if your circuit went through those countries – Tor can't help you. The adversary can track your traffic, and find out who you're talking to. I'm not saying this is actively happening, I'm saying we've proved in papers that it's possible, and it's outside Tor's threat model.



And a slightly more difficult version of that attack is if the adversary can see you, and then see the last leg of the path later on, like say, you're in China visiting a Chinese website. Well, they can do a similar attack, and track you down. It requires a little bit more math, but again, we've proved it's possible, and it's outside Tor's threat model.

And this is particularly concerning seeing as I, like most of you probably, are in the US.... And so much of what we do online is hosted in the Virginia datacenter of EC2.

<text>

So if either of those two cases apply, we're basically back at SSL, because the adversary can tell who you're talking to.



And at this point, I think it's worthwhile to show a couple of attacks on metadata. IOActive built a proof of concept traffic analysis tool, that looks at your SSL session with Google, and figures out what part of google maps you're looking at – all based off the sizes of the tiles you're downloading over SSL.

It's worthwhile to note this is an attack on a client, on someone browsing google maps at that moment. I want to show an alternate example.



You're sitting on facebook, with facebook chat enabled – all over SSL. Heck, all over Tor.

Well Facebook chat acts as a _server_ - you are able to receive messages from people, and they will be pushed down to you. The *attacker*, not you, determines when you will receive a message. That's a pretty powerful capability, and it can lead to time-based correlation attacks. An adversary sends you a message, and then looks at all the people connected to Facebook, or Tor, and sees who recieves a message right after that.



And even easier, because Facebook chats tend to be small – it can lead to size-based correlation attacks. Now not only do I send you a Facebook chat, but I send you a HUGE Facebook chat. With only a couple of trials you can be pretty confident that the user whose internet connection you're monitoring is the same anonymous Syrian dissident you're messaging on Facebook.

And it's interesting to note that a very similar attack was used to de-anonymize Jeremy Hammond, who is currently awaiting trial for allegedly dumping Stratfor's mailspools. The police staked out his home, watched him enter, saw some Tor traffic, and whoop – the username they thought was him, popped onto IRC. Classic traffic confirmation attack. And I've gotten some comments they also cut his Internet connection, and watched him drop off IRC, but I haven't seen the police logs from that side of things – if that is true, that's another type of traffic confirmation attack on a low latency connection.

http://arstechnica.com/tech-policy/2012/03/stakeout-how-the-fbi-tracked-andbusted-a-chicago-anon/



Now the good news is that even if the adversary can see the start and end nodes, or even the entire path, there is a way to disguise who you're talking to. And that's Mix Networks. Mix Networks introduce a delay, while they collect messages into a pool, and then fire them all out. Collecting the messages prevents an adversary who's observing the mix from knowing what message went where. It introduces uncertainty.

Mix Networks are a super important part of anonymous communication, that I want to encourage the growth of, so I want to take a quick minute to demonstrate it to you, live on stage.

Mix Networks (Remailers) If Who When How much

Alright, so Mix Networks demonstrated, we've gained back a certain amount of protection against figuring out who it is I'm communicating with. Given enough time, or a low enough traffic volume, an adversary can perform the same types of attacks I described against Tor – but it takes a lot more observation. And the easiest thing to learn, that takes no time or analysis, is the fact that I'm communicating, when I send a message, and how large it is – that is still apparent to someone observing my network connection.

Shared Mailboxes (AAM)

Date	From	Subject	Message
6/28/2013 4:15 UTC	Anonymous	f55d9f81fb5	BEGIN PGP MESSAGE
			hQ2134Aym
6/28/2013 5:32 UTC	Anonymous	e7bb5045c0	BEGIN PGP MESSAGE
			sdhQOAym
6/28/2013 5:56 UTC	Anonymous	B06c952e82	BEGIN PGP MESSAGE
			hQQOAym
6/28/2013 6:13 UTC	Anonymous	5465b12b5	BEGIN PGP MESSAGE
			MLwjVqd
6/28/2013 6:47 UTC	Anonymous	1ff4d96cf37	BEGIN PGP MESSAGE
			1JS0laiQp
5/28/2013 7:00 UTC Anony	Anonymous	2e6bd4698	BEGIN PGP MESSAGE
			FaW9hw9v

Enter Shared Mailboxes, and Alt.Anonymous.Messages. A shared Mailbox is what is sounds like. Imagine an email account where everyone in the room has the username and password – but it's read only access – you can't delete messages, or even send them from this mailbox.

All of the messages are encrypted, so what you do, as one of the people with access to this inbox, is download all the messages, and try and decrypt each message with your private key.

Shared Mailboxes (AAM)

6/28/2013 4:15 UTC			
	Anonymous	f55d9f81fb5	BEGIN PGP MESSAGE
			hQ2134Aym
6/28/2013 5:32 UTC	Anonymous	e7bb5045c0	BEGIN PGP MESSAGE
			sdhQOAym
6/28/2013 5:56 UTC	Anonymous	B06c952e82	BEGIN PGP MESSAGE
			hQQOAym
6/28/2013 6:13 UTC	Anonymous	5465b12b5	BEGIN PGP MESSAGE
			MLwjVqd
6/28/2013 6:47 UTC	Anonymous	1ff4d96cf37	BEGIN PGP MESSAGE
			1JS0laiQp
6/28/2013 7:00 UTC	Anonymous	2e6bd4698	BEGIN PGP MESSAGE
			FaW9hw9v

And a couple of those messages happen to be for you. The rest, you can't decrypt, so they must not be.



Well, someone watching this encrypted connection can tell that you're accessing the shared mailbox, and downloading all of the messages – that's certain. But they don't actually know if you've received messages – they only know that you downloaded all of the messages, not if you could decrypt any of them.

And because of that, they don't know when you've received a message, who it was from, or how large it was. All they know if that you're checking the mailbox.

At the cost of a lot of bandwidth, receiving messages via a Shared Mailbox provides an awful lot of security comparatively!



Now, shared mailboxes are an awesome anonymity tool, but the difference between an awesome anonymity tool and an anonymity tool that's actually used is the answer to the question: "Can I interact with the rest of the world?" Tor, wildly successful compared to other systems, because you can browse the actual internet with it. It's not a closed system where you only interact with hidden services.

So for a shared mailbox to actually be used, it needs to interact with normal email. That's where nymservs come in. The simplest nymserv, the newest and easiest to use, receives a message at a domain name, and post it immediately to alt.anonymous.messages. This is a nymserv written by Zax, and it's on github.



The much more complicated Type I or ghio nymservs can forward the mail to another email address, directly to alt.anonymous.messages, or route it through a remailer network to eventually wind up one of those two places. I'll talk more about this nymserv later on.



So if we add in nymservs, Shared Mailboxes have awesome anonymity for the recipient. When you send a message to a nym that uses a shared mailbox, you're ideally using an Onion Router or a Mix Network (although you don't have to), and thus have those security properties – an adversary can see that you're sending a message, when you sent it, and how large it was

		ESSARES		
alt.anonymous.me	essage: ×	<u>କ</u> =		
+You Search	Images Maps Play YouTube News Gmail Drive Calendar Mo	же +		
Google	Search for topics	- Q SIGN IN		
Groups	NEW TOPIC C Filters ~	<u>2</u> ° - ♀ .		
alt.anonymou 60 of 785976 topic	s.messages	About (🔍		
Anonyr	eb187373c397#7979acee23dd409039e03b916 mous - 1 post - 0 views	5:06 PM		
a 2a36bd256 By Nobody	8dc7fb02baf0a8340dc7d1c92d13755e0c908ff / - 1 post - 0 views	4:44 PM		
By Nobody	c207b4060e09c19457cc792b6190053df7c44e4 / - 1 post - 0 views	4:44 PM		
A 7caf60a703	3f2b3d9d5fcdc447be03d412e44efd2b4ae31a2 - 1. post - 0. views	4:44 PM		
e653287ba By Anonyr	758279f5ec032ff9542f76a2736fd3809e050fd nous - 1. post - 0. views	3:59 PM		
C5b5bbcde	3dbe29ab14ee2364796ae4a91a2899cd908fa78	3:44 PM		
9d63e31de	ffa7c7d5d64d8dbb493ac29684eff3b3978f772	3:44 PM		
a 1a307c2de	2988dcdcd50814ba2d328dfc0d8a4a01797acfd	3.44 PM		
Dy Nobody	- 1 post - 0 views	5.44 PM		

So, now that I've walked through the security properties of the different types of anonymity networks, let's actually dive into AAM. It should have really strong security, afterall it's the most theoretically secure.

If you've never looked at it before, this is what it looks like in Google Groups. A bunch of hexadecimal subjects, posted by Anonymous or Nobody



And any individual message usually looks like a PGP Message that may or may not have a version string.



There's about 190 messages posted per day these days, but what's interesting is while the average certainly has decreased over the last decade, it's held somewhat steady in the last 5 years.



The dataset I worked off of was about 1.1 million messages from the last ten years.

Now we can already see some shortcomings here. Over half of the messages in my dataset go through nodes operated by two people. The network diversity is horrible, and the network itself would be thrown into disarray if either one of these folks got subpoena-ed, shut down, or retired. But, it's actually much worse than this slide.

603,844 / 1,128,312 = 53.5% Dizum: 416579 Zax: 192317



You see that 53.5% statistic was over the entire dataset. Today, these two folks make up virtually all of AAM.

That dip: 7,800 messages through Frell, which operates a remailer and a newsgateway

Subject: España busca que el consejo resuelva problema de activos tóxicos No unique headers, identical PGP signaturas of Tag types 10,3,9

I couldn't get much out of those messages, other than that someone sent out 7800 messages in a group, over a short timespan, and then stopped..

So, with network diversity pretty clearly abolished, let's take a look at the data, and see what types of analysis we can do.



I don't think I can say anything as ironic as this quote, which I pulled from literally, 1994.

Read it

And here we are, just shy of 20 years later.



So the first thing to do is to break it up by PGP or Not-PGP. And you can see it's overwhelmingly PGP messages.

So, really quickly, what are the non-PGP messages?



Well, I was trying to come up with a nice way to say crackpots – I'm not sure if I succeeded.

But there are several people who have and continue to post just... random... rants. About... I'm not really sure. And there are actually Frequently Asked Questions that have sprung up in response to the crackpots, because people were just getting flat out confused.



So, besides those there are some other non-PGP messages. I think most interesting is a set of about 10K messages with the subject 'SATANIC OPERATION', or OPERATION SATANIC. What's interesting about these messages is they're clearly ciphertext, but alphabetic. If you look at a single message, you almost think it's a Ceaser Cipher, or a Vigenere, or a polyalphabetic. But if you analyze the messages in whole, you discover a 16 letter alphabet with a perfectly even distribution.

In other words, I think it's a substitution cipher into Hexadecimal. And the even distribution implies it's ciphertext of some sort.

And there are other message clumps similar to this, so if you're into this sort of analysis, have at it!



So the next thing to look at is what percentage of messages were delivered to AAM via a nymserv or a remailer.

These numbers are going to be a bit off, since some of the PGP or Remailed messages are actually to nyms, and some of the PGP messages may be through remailers I don't know about. But it's something. We can see that a large portion are messages to nyms, which will be important when I eventually tell you how many nymservs are actually running.



All right, so those somewhat interesting statistics aside – let's start diving into all of those hundreds of thousands of encrypted messages. So if you didn't know, OpenPGP consists of packets, and each packet type does something slightly different. There's a packet type for a message encrypted to a public key, and a different packet type for a message encrypted to a passphrase.



So what are these packet types. These graphs show the popularity of each of the different packet signatures, i.e. packet 1, followed by packet 9

The top 5, the ones on the bottom, are the ones you'd expect to see.



1 is messages encrypted to a public key.



3 is Messages encrypted to a passphrase.


The actual ciphertext of a message is 9 or 18 for old-style or new-style. And I separated out the messages sent to a single public key vs. ones sent to multiple.



There are two that are weird. These are packet types you'd expect to see after you *decrypted* a message. These are plaintext packets. There are actually a small number of messages that look like OpenPGP data – they've got the BEGIN PGP MESSAGE ticker and they're base64ed – but they're actually plaintext. Just hiding in plain sight.



And if we look at packet type 8 – this is what we get. It really just is compressed plaintext data. Unfortunately, it's also nonsense. I don't know if there's a code there or not, but I didn't spend any time on it, I figured "Iran ongoing bizarre sabbatical" probably came out of some makov generator somewhere. So I moved on to...



The messages that were sent to public keys. It's super obvious to do analysis based on the public keys in the message. I promise you the analysis gets more complex later. But lets look at KeyIDs.



So obviously the KeyIDs are a pretty powerful segmenting tool. So I wanted to illustrate a couple of examples where the KeyIDs tell us more.

There was one KeyID that was messaged very reliably through a nymserv. Except for 2 messages sent through EasyNews. If you track down the very unique easynews gateway + User Agent, we find that that person also messages another KeyID. We can start making inferences across multiple types of metadata.



Now I mentioned that I separated the messages that were sent to a single public key from the ones sent to multiple. If a message was sent to a single key, we don't know too much about it, especially because usually they throw the key ID, so you can't tell what public key it was encrypted to. But if a message is sent to more than one public key ID, then...



You can draw communication graphs. Now it's not a strict communication graph in the sense that a message was sent from Alice to Bob, technically it's that Alice and Bob both received the same message. But in some, if not most, situations, people include themselves on messages they send... so they can read their own sent mail.



So a quick legend to these graphs, if a node is green, that means I was able to find the public key on the a keyserver.

If the node is a circle, that means that key received messages individually. And the size of the circle, and the width of the line, means how many messages they received.

So we have this very nice symmetrical 5-person graph here.



And then we've got these much larger communication networks here.



And then we've got this huge spiderweb of messages.



And we've got a couple of interesting graphs with central communication points.



And then we've got a couple of more interesting networks. And I think these are interesting because they imply that not everybody knows everybody else. This graph and the next one really may be a model of the actual Internet where people will email other people and in a complex, interconnected, but not fully connected way. This is a fairly low-volume network



While this one has quite a few higher-volume folks participating.



And then here's the rest of them the simpler, 2-person communications.



So I was working on the communication graphs after all the PRISM stuff came out, and I was feeling distinctly uncomfortable imagining that this is what the NSA is probably doing to me and my friends.

But the show must go on, so let's talk about brute forcing ciphertext. Now if you'll recall this graph, you saw that packet type '9' was by far the most common packet type found – over 700,000 of them. Now this packet type is interesting so let's dive into a little bit.



This packet is the actual ciphertext of the message. It is only, the encrypted data. It doesn't say what algorithm it is, and it doesn't explain how to get the key.



The key, is in another packet. It's in packet type 1 (for public keys) or 3 (for passphrases).



But if you'll recall from that graph, there aren't any packets that precede packet type 9. We've got a disconnect from what the spec says, and the data we see.



Well if we keep reading, we'll find this gem.

"the IDEA algorithm is used with the session key calculated as the MD5 hash of the passphrase"



Yea. The MD5 of the password.

This is absolutely legacy, and we've had better ways of doing this in OpenPGP since the late 90s. So while in the very beginning of AAM, this might have been excusable, the fact that my dataset was from 2003 onwards makes this a pretty horrible situation.

So we know how to MD5s really, really fast. But that's only half of this. We have to take the output and use it in an IDEA decryption. And then we have to detect if what we decrypted to was an actual plaintext, or just random. And while you can run randomness tests – they're slow, and we're brute forcing here – we want to go as fast as possible. So while I spent a lot of time at this point, wrote a lot of code and did a lot of optimizations, it doesn't play very well into the slides, so I'll just say that I wrote a lot of CUDA-powered code and brute forced these on GPUs for many months.

And one of the first results I got, actually a few dozen of these messages, was



This did not make me feel terribly good about myself. But I persevered.

Other Decrypted Messages

- HTML Pages
- Weird SMTP Logs
- Partial Remailer Messages
- But overwhelmingly...



More encrypted messages. Recursively encrypted PGP messages.



In fact, here's a breakdown of how many recursions I hit. I got about 10,000 decryptions into a public key message, and another 2200 into another password-protected PGP message. I was able to take 49 messages two layers deeper, and 5 messages 4 layers deep.

Now, for the number of messages I was trying to brute force, these numbers may not seem very impressive. While I certainly am not the best password cracker here at Defcon, I think it's worth bearing in mind that I am not trying to crack passwords, I'm trying to crack encryption keys used by some of the most paranoid people on the Internet. So I'm sure people can do better, but I don't feel too bad about these results.

But I haven't explained why there are so many recursively encrypted messages.



And to explain that I have to talk about Remailers. So how many have heard of Mixmaster and Mixminion.

Okay a good number of you. Well these tools have been dubbed Type II and Type III remailers. Which means there must be a Type I remailer somewhere. Well, Type I remailers are basically dead, but their protocol lives on in Mixmaster.



And boy, what a protocol.

This is the manual of how to use most, but not even all, of the options supported by Type I remailers.



Some of the Type I directives are on the left. Now, what's the difference between Remail-To, Remix-To, Anon-To, and Encrypt-To? I sure as heck don't remember, and I've been studying this for a while. And to use a Type-I, you have to type each of these options out, yourself. There's usually no GUI here.

I had talked in the beginning about Type I nymservs? Well, Type I nymservs are the main recipient of these directives. You would string together a mix network chain of directives, encrypted to different nodes, and that would be your reply block. When someone emails your nym, the nymserv would basically execute your reply block, sending the message off through each of the steps, ultimately coming out to either your real email address, or a Usenet group like AAM.

We're still seeing these messages posted. But there are only 2 Type I Nymservs operating. One is Zax, of course, the other is paranoici. Paranoici is run by a group of Italian Hackers in Milan, they also run Autistici, Inventati – which you can think of as an Italian version of RiseUp.



So, in conclusion what are those nested PGP messages? They're Type I nymserver messages, where the keyID is the ultimate nym owner. If I don't have a keyid, there's another layer of symmetric encryption I didn't crack. When you download Type I nymserver messages, you know all the passwords, peel them off one by one, and then finally use your private key.

This is all the recipients with >5 messages. Pretty top heavy towards just a few nyms.



So Communication Graphs and brute forcing is really just the first, quarter, I would say, of the analysis I did on AAM. A majority of my time was spent doing Correlation. Even if I don't know who a message is to, or what it says, it's valuable to know that it's to the same person as another message, or was sent by the same sender.



And why is that valuable? Well, let's go back to this slide. You can't tell if someone has even received a message in a shared mailbox. But if I can correlate one message with another,



Then I can start determining that some unknown person _has_ received a message. And once I know these two message are related, well I can pay attention to the timestamp and the length. This goes even further,



because people tend to respond to messages they receive. And since I know If someone has _sent_ a message, it might be that they are replying to a message they just received.

So let's talk a lot about correlation, and more analysis on what's in AAM.



So first off, it's obvious that you can correlate messages that use a single, constant subject. But there are a lot of messages like these! Nearly half of all the messages posted to AAM!

They tend to be older, and have tapered off more recently. Which makes sense.



And if you've looked at AAM, what you've probably seen is the random Hexadecimal subjects. Those look random. Let's correlate them.

So there are two algorithms to generate these subjects. Esubs, or Encrypted Subjects, and Hsubs, or Hashed Subjects. And the point of these is to quickly identify which messages are for you, and which you should ignore. This saves you an expensive public key operation. Now personally, I think we're at the point we could probably cut this step out, but nonetheless, it's there.

So esubs have two secrets – a subject, and a password. Hsubs have a single secret, a password. If you want to brute force these, it's considerably more difficult to brute force the esubs – and I ran out of time. Now you'd think that esubs must be newer, but actually it's the hsubs.



Hsubs were created by Zax actually, and as his services are used more and more, they make up an increasing percentage of the subjects.

Now, hsubs have a random piece in them that you can think of as an Initialization Vector, or as a salt. While I could try to shoe-horn these into the existing SHA256 password crackers out there, it'd be really painful, because hsubs will truncate the output to match the length of esubs. So I had to write my own GPU cracker, again.



And I cracked about 3,500 hsubs. Better than the percentage of messages I brute forced, but again, not a great percentage. But keep in mind these are passwords of the most paranoid people on the Internet.

I found an interesting set of messages with the hsub DANGER WILL ROBINSON, which was used by some, but of all, of the messages to a couple of particular KeyIDs.

I cracked all the hsubs of another Key ID, with the two passwords testicular and panties.

If you don't know what schmegma is, don't urban dictionary it.


So if HSUBs and ESUBs are used to let a nym own identify their messages, can we do something similar? Let's say we want to target the nym Bob. Well, what we can do it send a particularly large message to Bob, full of nonsense. And then we wait for a large message to pop out into AAM. Zax's nymserv is instantaneous, so this size-based correlation is easy. Type I nymservs are not necessarily instantaneous, so they're a little more difficult, but it's not _too_ difficult. We can get a very good idea by keeping careful track of the size and maybe doing it a couple times.

And this works, easily and efficiently. And what we get is a specific message we know is to a particular nym, that we can then target for hsub cracking.



So I'm not done.

But unlike everything I've presented before, what I'm going to talk about now is probability-based attacks. That is, I come up with a hypothesis, that I can correlate messages with a probability better than random if I look at property X. Whatever it is. Well, if I don't have a control or test messages, I can't tell if that hypothesis worked, right? Well, I don't have controls. So what I'm doing is coming up with a hypothesis, running it across the dataset, and then looking at the clusters of messages that come out. And if I can figure out something _else_ that correlates them, I call it a success.

Like say – If a message header has a value of X - I think that's a unique sender. Only one person is sending those messages. So I run that analysis, and I get clusters of messages encrypted to a single public key. Well, if there was no correlation, I wouldn't get such nicely segmented public keys, would I? It would be a random distribution of the all the public keys in the dataset. And even though I could have found that cluster by just looking at the public key ids – this data implies that I could use that trick, that hypothesis, to find clusters of data when there _is_ no other distinguishing characteristic.

So that's how I try and preserve some semblance of the scientific method, while not actually having controls.



So my first example was message headers, and that's a big one. Let's look at these.



There are a few headers that are in nearly every message, but a long tail of headers that in only a few.



But those mostly-unique message headers are not necessarily the goldmine you might think they are.

And that's because headers can be added by the client, by the exit remailer, by the mail2news gateway, or by the Usenet peer.

Ν	Aessage Headers
- Headers injo - Headers injo - Headers injo	ected by the Usernet Peer ected by the mail2news gateway ected by the exit remailer
Left With:	Headers specified on the client

So to really go after the distinguishing headers, that is the headers added by the client – I have to subtract out the headers that were added by all the other parts of the path.

Client Message Headers

- User-Agent
- X-Post-Type-ID
- Author-Supplied-Address (sometimes)
- Encrypt-Subject (this is user error)
- X-No-Archive
- x-archive-no

And here are some great examples of headers specified by the client.

X-Post-Type-ID X-Post-Encrypted X-Post-Priority

These strange headers all formed a distinct clump of messages, with the unique subject "Weed Will Save The Planet". An easy example of how the idea of unique message headers can correlate messages.



X-No-Archive – this means, don't save it in Usenet. It's a client request that most usenet servers will obey.

It's also not the word on the screen. This is a misspelling of the header. And there is one person, at least I'm claiming one person, who has messed this up, and completely distinguishes their messages. All 17,300 of them.

~17,300 messages with this header Every one of them w/ subject of 'forforums'



So this is what you want, right?

No. Capitalization matters, and this is not the correct capitalization.

What's interesting about this one is that it shows up on several long running threads on AAM, composing nearly 28,000 messages. Now initially, I thought each of these threads was relatively independent from each other – but after finding this bit of information – I'm starting to seriously doubt that.



This one isn't right either.

1500 messages posted with this header Including test messages posted with someone's real name



This is the correct version, and about 135,000 messages had it, or a little more than 10%. Which makes it distinguishing in and of itself.

Encrypt-Subject			
Encrypt-Key: deceptionisanart Encrypt-Subject: theartofdeception Subject: message for chrome Date: Thu, 3 Jun 2004 03:40:32	Encrypt-Key: deceptionisanart Encrypt-Subject: theartofdeception Subject: message for chrome Date: Thu, 3 Jun 2004 02:52:27	Date: 7 Jun 2004 14:09:43 -0000 Subject: message for chrome ** BEGIN PGP MESSAGE Version: 2.6.2	
lets see if this works also :-/	lets see if this goes :-/	pgAABdWtU+e9qeX58OyVtesj/T4x oKYVUhVrFF88QSFDv5D+Rxhmv0 +OdcfaS2Ua	
Encrypt-Key: deceptionisanart Encrypt-Subject: theartofdeception Subject: message for chrome Date: Thu, 3 Jun 2004 03:40:32 lets see if this works also :-/	Encrypt-Key: deceptionisanart Encrypt-Subject: theartofdeception Subject: message for chrome Date: Thu, 3 Jun 2004 02:52:27 lets see if this goes :-/		

So how about Encrypt-Subject?

So Encrypt-Subject is an directive for Type I remailers that should be processed by the remailer – it should never make it's way into Usenet. This is a bug, this is a client messing things up. And I can't blame them, because Type I is so horribly difficult.

Over 10000 messages like this. And when you reuse the subject, like these, you make messages without Encrypt-Subject stand out

Encrypt-Subject			
Encrypt-Subject: ews4sRo+x	Posted Date	Newsgateway	Esub
Subject: ieoyue	10/23/2005 22:23	dizum.com	6cbfd
Date: Mon, 1 Nov 2004 23:00:05	10/23/2005 23:14	bananasplit.info	28e6c
	10/23/2005 23:35	dizum.com	934be
	10/24/2005 0:28	nym.alias.net	dec98
BEGIN PGP MESSAGE	10/27/2005 22:37	bananasplit.info	cfb1a
Version: PGP 8.1	10/27/2005 23:03	dizum.com	038ab
	10/31/2005 1:21	bananasplit.info	372da
	11/7/2005 0:22	dizum.com	94498
3R/107TDk8wKidyT0III myd077oINK	11/7/2005 1:34	bananasplit.info	46a10
DIxVm	11/13/2005 22:50	dizum.com	99935
	11/13/2005 23:11	bananasplit.info	0621d
	11/14/2005 1:23	nym.alias.net	1b1d9

Or even worse, mess up once, and then figure it out and reuse it...

I can identify 52 esub messages that were otherwise secure because of subject/password reuse



And then there's Encrypt-Key. Another header that should never make it's way into Usenet, but does because Type I remailers are so hard to use.

There are over 10,000 of them.



Let's look at another header. Newsgroups. Just list mailing lists, you can post a message to more than one newsgroup. But if you do, you're wildly in the minority, and that segments you.



Like this newsgroup. There are 34 messages posted with this newsgroup, and than you so much comcast for making your users extremely distinguishable.

34 messages every one of them:

> subject: mlw0lj2b9HBP7EURCn0PdCvyyatVk8i Adam S. Toline uniquehsig: 8, 40 40 User-Agent: Xnews/03.04.11



Well what about this value. AAM with 4 commas at the end.

I thought this was a correlation attack – but after tracking it down, it was actually caused by a bug in 'remailer.org.uk' for a week in January '06. Random trivia I pulled out of this dataset.

1/21 - 1/29 2006



How about this one, with duplicated newsgroups. These were sent through a large variety of remailers and have no obvious correlation besides this value, and that they have english subjects.

So the English subjects was the control I used to confirm that using a unique newsgroup is a bad idea.

Lot of

- ATTN T-Boy
- Pariser Wasser
- Fresh fish from China

All of these have these two newsgroups, sent through many different remailers: - melontraffickers, frell, cypherpunks.to, dizum, rebleep, tatooine, paranoici, firenze

7743 reece.net.au	reece.net.au	Re: So much for freedom of the Internet
7277 news.mixmin.net	mixmin.net	Re: So much for freedom of the Internet
7175 reece.net.au	reece.net.au	Re: So much for freedom of the Internet
8806 mixmaster.it	news.bananasplit.info	Re: So much for freedom of the Internet
7089 news.mixmin.net	mixmin.net	Re: So much for freedom of the Internet
8696 reece.net.au	reece.net.au	Re: So much for freedom of the Internet
7085 fleegle.mixmin.net	mixmin.net	Re: So much for freedom of the Internet
8544 fleegle.mixmin.net	mixmin.net	Re: So much for freedom of the Internet
8464 reece.net.au	reece.net.au	Re: So much for freedom of the Internet
8272 www.ecn.org	dizum.com	Re: So much for freedom of the Internet
8166 reece.net.au	reece.net.au	Re: So much for freedom of the Internet
8064 news.mixmin.net	mixmin.net	Re: So much for freedom of the Internet

Humans are creatures of habit, and as flaky as remailers have been, a lot of people find a configuration that works for them, and then they stick with it. Well, if I partition people by the remailer and the newsgateway they use – that's what the colored squares are – what was previously an "anonymous" discussion thread suddenly makes it very easy to pick out who is saying what, and if they're arguing with or supporting themselves.



And if I add in the header signature at the far right, it's even easier!

Redundant Messages

Posted Date	Sender Domain	Subject
2003-07-21 09:07:04	nym.alias.net	soggy
2003-07-21 09:33:56	nym.alias.net	soggy2
2004-12-10 14:08:47	nym.alias.net	i4pPm1nona
2004-12-10 14:24:17	nym.alias.net	i4pPm2nona
2006-11-23 17:51:48	nym.alias.net	soggy2
2006-11-23 17:51:50	nym.alias.net	soggy

And then here's a really interesting pattern I observed. There are a host of messages who have subjects with a 1 or 2 in them. Like soggy / soggy2.

Well I looked at those, and found they were being posted together, really close together. And then I realized – one of the options in Type I remailers is to duplicate a message for redundancy. Send the same message down two different remailer chains, just in case one becomes unavailable. And while this gains you some measure of availability – it's also distinguishing.

You could target a nym, like I described earlier, with a huge message – and if you see two huge messages appear, you know that that nym's reply block duplicates messages. Look for all possible duplicate messages, and you've got a candidate list of messages to that nym – even if you're unsuccessful doing an hsub or esub crack.

Response Patterns?

Posted Date	Sender Domain	Newsgateway	Unique Header Sig
2007-03-06 15:06:08	anonymous.poster	panta-rhei.eu.org	12 114 115 116
2007-03-06 20:10:07	dizum.com	dizum.com	71 114 115 116
2007-05-20 19:46:19	anonymous.poster	panta-rhei.eu.org	12 114 115 116
2007-05-20 23:10:06	dizum.com	dizum.com	71 114 115 116
2007-05-22 03:00:36	anonymous.poster	panta-rhei.eu.org	12 114 115 116
2007-05-22 08:00:24	dizum.com	dizum.com	71 114 115 116
2007-07-07 06:29:05	anonymous.poster	panta-rhei.eu.org	12 114 115 116
2007-07-07 12:00:02	dizum.com	dizum.com	71 114 115 116
2007-07-29 09:53:43	anonymous.poster	panta-rhei.eu.org	12 114 115 116
2007-07-29 14:20:03	dizum.com	dizum.com	71 114 115 116
2007-08-30 19:03:38	anonymous.poster	panta-rhei.eu.org	12 114 115 116
2007-08-30 23:30:02	dizum.com	dizum.com	71 114 115 116

And a similar pattern I saw was these. Look at each pair of messages with the slightly different backgrounds. The second message comes out of dizum about 5-6 hours later from the one that comes from panta-rhei.

I don't know what this means, but it did stand out and is distinguishing.

Subject: "Weed will save the planet"

Also, messages from frell were mixed in, with no obvious correlation to other messages



So there were a number of other hypotheses I tried that did not turn up interesting data, and there are more queries that could be run across this dataset. But I need to start wrapping up. It all comes down to Metadata.



What we saw is that AAM had the obvious mistakes we'd expect.

And it also suffers a bit because it hasn't taken into account the lessons we've learned since it was developed.

But I do think there's some traffic analysis lessons we haven't codified as best practice, that we probably should.

AAM – Future?

Security of a well-posted message is good****

So what does the future hold for AAM? Well, the security of a well-posted message is good.



With a lot of caveats. If you use uncrackable passphrases, only use servers that output key stretched packets, post through remailers, with no distinguishing characteristics, and you're willing to be a very small anonymity set.

I don't know how many people are using AAM, but it's not a lot. What that means is if the government asked for a list of everyone who uses AAM – they would get a very short list of names. Probably small enough to dig pretty deeply into each of their lives.

AAM – Future?

- Security of a well-posted message is good****
- But...
 - Relies on remailers & newsgateways
 - Relies on Usenet
 - Extremely limited bandwidth

And AAM crucially relies on Remailers and Newsgateways. And these services are dying. Remember that 2 people, Zax and Dizum, post more than 98% of the traffic to AAM.

AAM is also text-based – very limited bandwidth.



And the nymservs themselves are pretty crappy, architecturally speaking. We give single-hop proxies like VPNs and UltraSurf a lot of shit because their Architecture is not as strong as Tor's.

But nymservs are in that exact same category of "Trust this guy not to roll over on you"

I feel compelled to mention that the alternative is to use Tor, which you trust, to send email via throwaway accounts on a service you do not trust. While this is a practice everyone in this room has used or at least thought of – it's still a really shitty architecture.



Now the good news is we have something better. We have a very strongly architected Nymserv.

Pynchon Gate was designed by Len Sassaman, Bram Cohen, and Nick Mathewson, and it uses Private Information Retrieval instead of a Shared Mailbox. It exposes less metadata, and resists flooding or size-based correlation attacks.

However, it's not built. It's been started, but it's got a very long way to go. It also requires a remailer network to operate.



And we don't really *have* a remailer network. What we've got is Mixmaster and Mixminion. Now Mixminion is a bit better than Mixmaster, which doesn't have any Link Encryption, has known attacks, uses old crypto with no chance of upgrading.



But both of these services suffer from the fact that we don't have a good solution to remailer spam or abuse, we don't have good documentation about them, and they both have horrible network diversity.

Remailers - Mixminion

- New Packet Format (Sphinx)
- Improve TLS Settings
- Distributed Directory
- Cryptographic Implementation Upgrades
- Pinger
- General Love
- <u>https://github.com/nmathewson/mixminion</u>

So if we like Pynchon Gate, the path forward also involves fixing Mixminion. And mixminion needs love.

Mixminion is currently unmaintained, but we have a TODO list that includes the items I've got here. Some of them are extremely complicated, like moving to a new packet format. Others are relatively straightforward, like improving the TLS settings, and others give you the opportunity to practice writing crypto, designing a distributed trust directory, or writing a complete standalone pinger in any language or style you want.

So if you're interested, there's a lot of pretty cool opportunities here.

High-Bandwidth, High Latency

But what I keep coming back to is the fact that we have no anonymity network that is high bandwidth, high latency. We have no anonymity network that would have let someone securely share the Collateral Murder video, without Wikileaks being their proxy. You can't take a video of corruption or police brutality, and post it anonymously.

Now I hear you arguing with me in your heads: Use Tor and upload it to Youtube. No, youtube will take it down. Use Tor and upload it to MEGA, or some site that will fight fradulent takedown notices. Okay, but now you're relying on the good graces of some third party. A third party that is known to host the video, and can be sued. Wikileaks was the last organization that was willing to take on that legal fight, and now they are no longer in the business of hosting content for normal people.

And you can say Hidden Service and I'll point to size-based traffic analysis and confirmation attacks that come with a low-latency network, never mind Ralf-Phillip Weinmen's amazing work the other month that really killed Hidden Services. We can go on and on like this, but I hope you'll at least concede the point that what you are coming up with are work-arounds for a problem that we lack a good solution to.



So if I've been able to entertain you, I am glad, if I've been able to inspire you to work on anonymity tools, I am overjoyed. And if you want a place to start, I will point you here.

Thank You.






What Are The Tools?

- Networks
 - Mixmaster
 - Mixminion
 - Newsgateways
- Nymservs
 - Type I Nymserv
 - 2 operating
 - Nymserv by Zax
 - 1 operating
 - (Tor+Webmail)

- Tools
 - Mixmaster
 - Mixminion
 - QSLite
 - Omnimix
 - QSAam
 - Aam2mail
 - Aamfetch





Nymservs

- Type I Nymserver
 - Paired with Type I remailer messages
- Zax' nymserv
 - Considerably updated
- Both are single hop, single operator

Remailer Clients

- Mixmaster
- Mixminion
- 'Official' clients
- Command Line
- Bit obtuse

- QSLite
 Win32 GUI Client
- Omnimix
 - Win32 GUI Client
 - Closed Source

AAM Clients

• QSAam

- Aam2mail
 Linux client by Zax
- Aamfetch
 Java client