Covert Channels

Towards a Qual Project

Rachel Greenstadt

Harvard University

Overview

- About covert channels
- Example channel: TCP timestamps
- Problems with the example channel
- Directions in covert channel research

What’s a Covert Channel?

- A channel transfers information in a way that violates a security policy
- This comes from military literature
- Alternately, consider ...

Alice and Bob in Jail

- Alice and Bob plan to escape
- But the Warden monitors their messages!
- If the warden suspects -> solitary confinement

Isn’t that a bit subversive?

- Well, yes...
- But censorship resistance
- And privacy
- And freedom
- Ok, how do we start?

Threat Modelling: Know Your Warden

- Watch traffic over channel
  Attempt to detect suspicious activity
- Close off potential channels through filtering
- Allow *legitimate* communication.

Covert Channel Properties

- Undetectability
  - Plausible (legitimate cover)
  - Open functionality
  - Encode the message to match channel statistically
- Robustness
  - Message survive natural/malicious lossiness
  - Indispensable

Example Channel

- My first publication!
- joint work with John Giffin, Peter Litwack, Richard Tibbetts
- Broken in some ways

Why TCP Timestamps?

- TCP ubiquitous - plausibility
- Possible to modify the timestamp/delay packets
- Slow connection - low order bits random
- Encryption produces random bits
- Seems simple, encrypt message,
  hide it in low order bits

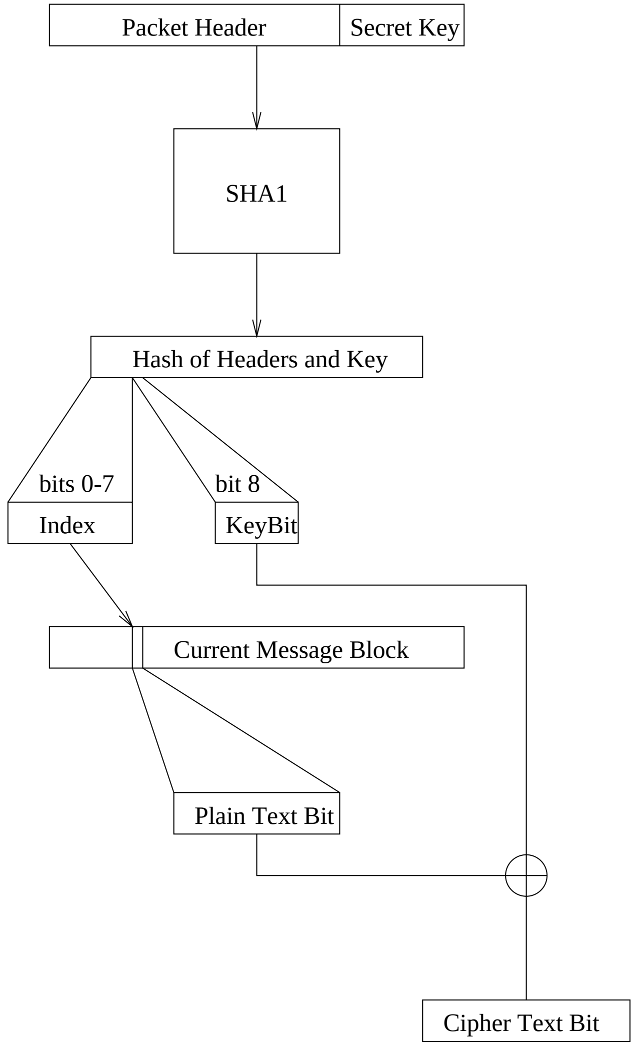
Robustness???

- Don’t get TCP reliability if you use the timestamps!
- Bits delivered out of order
- Bits dropped randomly
- Data acknowledged, not packets, can’t get reliability there.
- Timestamps must increase
- Timestamps are an option, can be replaced/squashed.

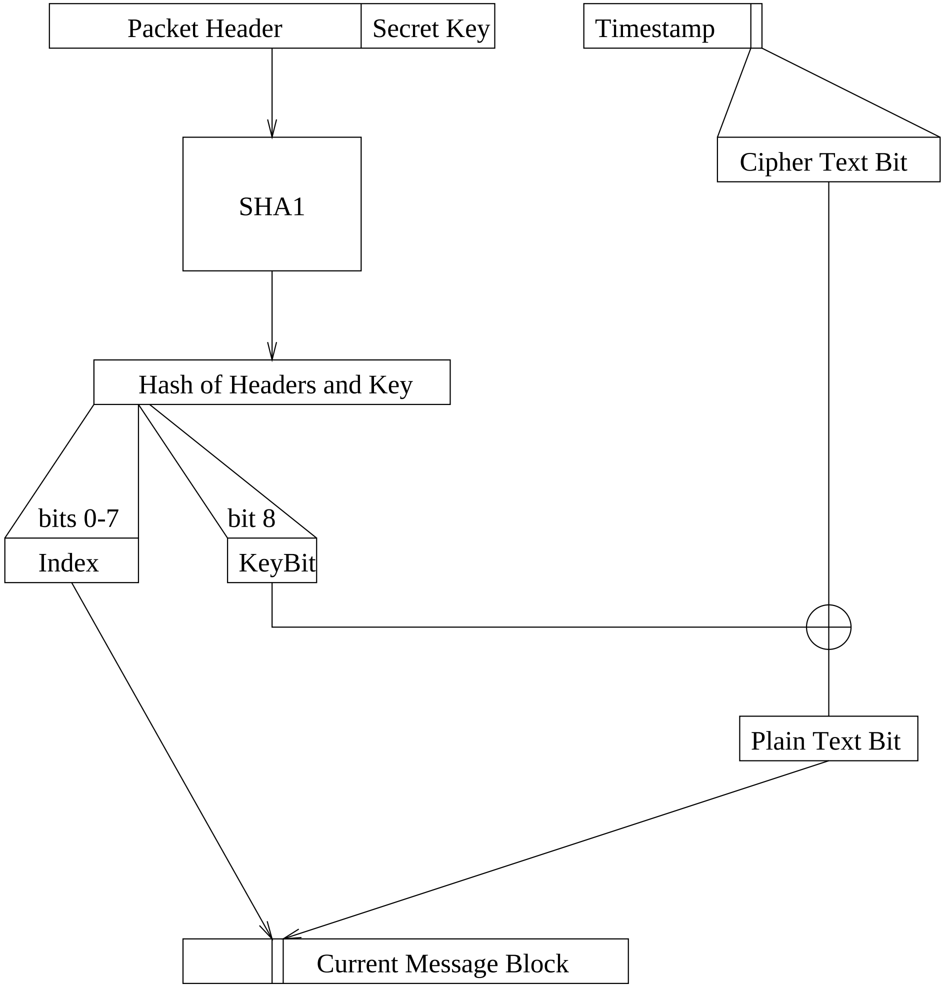
How to get reliability?

- Divide data into blocks
- Use a hash of the headers to tell receiver which bit is in timestamp
- Encrypt that bit
- Make sure you send each bit *o* times
- Assume the receiver will get the block, then move on
- The receiver keeps a checksum to tell when to move on to next block

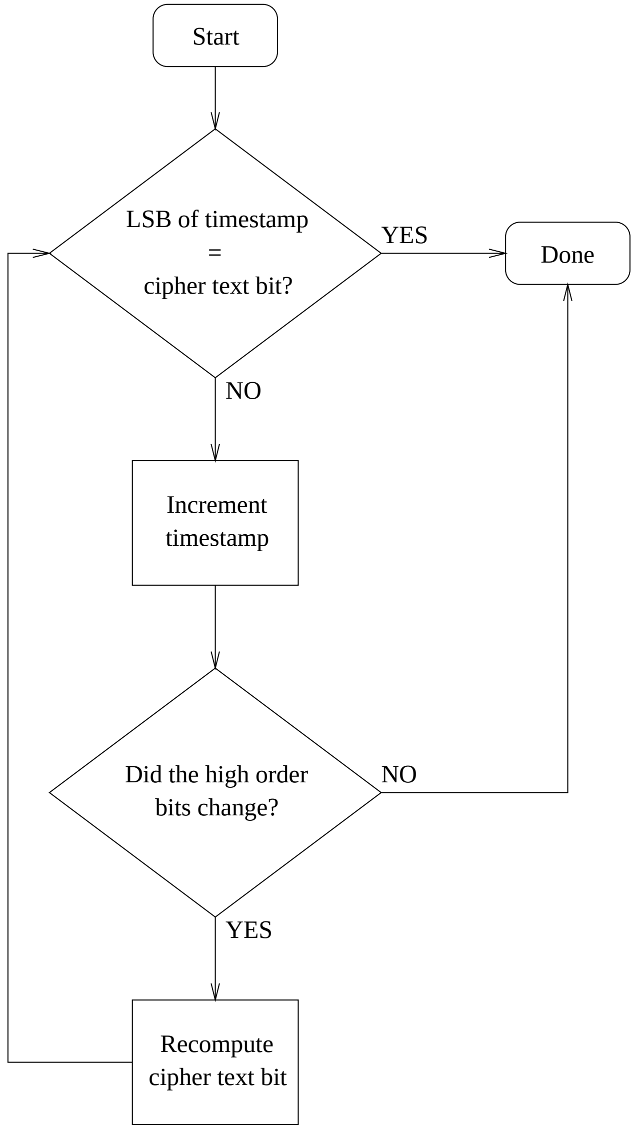
Sending Data



Receiving Data



Rewriting the Timestamp



Detecting the TCP Timestamp Channel

- Drew Hintz, Defcon 10
- Problem: Low order bits aren't cryptographically random
- Algorithm:
  - Record all the low bits of the timestamp
  - Put them through a complex randomness test
  - If very random, then covert channel used

Can This Idea Be Saved?

- Increase the occupation number
  - (or use some less braindead error correcti scheme)
- Model the distribution of timestamps
- Remove some packets to lower the entropy of the channel
- Arms race?

Should This Idea Be Saved?

- Complex, low bandwidth channel.
- Easy to remove anyway - timestamps are an option you could strip them from the packets or modify them.
- Maybe better off with another channel (say TCP initial seq numbers)
  - Are they really random?
  - Removable with a 32 bit offset

Security Through Obscurity?

- Can you have a widespread covert channel?
  - example: break the Chinese firewall?
- In crypto, algorithm public, key secret
- But known channels are closeable
- Should the channel be secret too?
- 3 can keep a secret if 2 of them are dead.

Solutions?

- Superiminal channels.
- More generalized covert channel scheme
  - Easy to apply to new channels
  - In band method of channel rotation.

Back to Randomness

- Maybe hard if limited to using true cryptographic randomness
- Need to encrypt to arbitrary distributions
- Maybe use ECCs and the rejection method
  - Graph desired distribution,
  - Pick uniform distribution which is larger
  - Remove anything which doesn't fit

Potential Directions

- Come up with a flexible covert channel scheme which can be used in many channels
- Create a protocol for jumping between multiple covert channels.