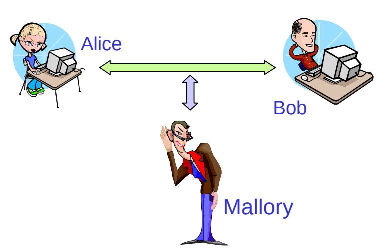
# Authentication of LZ'77 compressed data

*Stefano Lonardi*
University of California, Riverside

joint work with *M. J. Atallah (Purdue U.)*

## Problem

- Alice sends a document *T* to Bob
- She wants to make sure that what Bob receive is
  - Authentic
  - Integral




- Mallory monitors the communication and he will attempt to tamper with *T* and impersonate Alice

# Signatures

- Signature requirements
  - Authentic/Unforgeable
  - Not reusable
  - Cannot be repudiated
- The signed document should be unalterable *(integrity)*
- Typical solution involves PKC

**Stefano Lonardi**
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Fragile watermarks

- An alternative way to authenticate a document and ensure that it reaches the destination in a integral state is to use a fragile watermark
- A *fragile watermark* is a watermark designed to break as soon as the content of the document is changed

**Stefano Lonardi**
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Rationale

- Textual data is difficult to watermark
- Lossless compression is very common nowadays (compress, gzip, (win)zip, (win)rar, lzh, bzip2, etc.)
- Since we are sending the document over the network and it is likely that we are going to compress it anyway, why not watermark the compressed file?

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Notations

- *T:* document
- *k:* secret key
- *W:* (fragile) watermark
- *T':* watermarked & compressed document

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

## Specs

- *T=T'* (or semantically equivalent)
- Unless *k* is known
  - it is very hard to retrieve *W* from *T'*
  - it is very hard to add *W* to another text and pretend to be Alice
- The presence of *W* in *T'* shuld hold up in court (false positives are extremely rare)
- The security of the process should be based solely on the secrecy of the key *(Kerckhoffs' principle)*

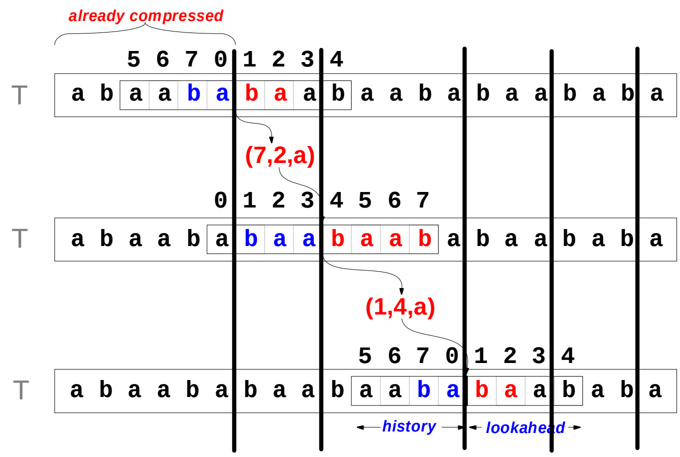
Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

## Approach

- We propose a method that hides *W* (the digest of *T*) directly in the compressed file as a fragile watermark
- Advantages
  - transparency (and therefore backward compatibility)
  - does not require to send separately the signature (authentication is embedded)
- We also satisfy all the previous requirements

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Lempel-Ziv 77 (gzip)



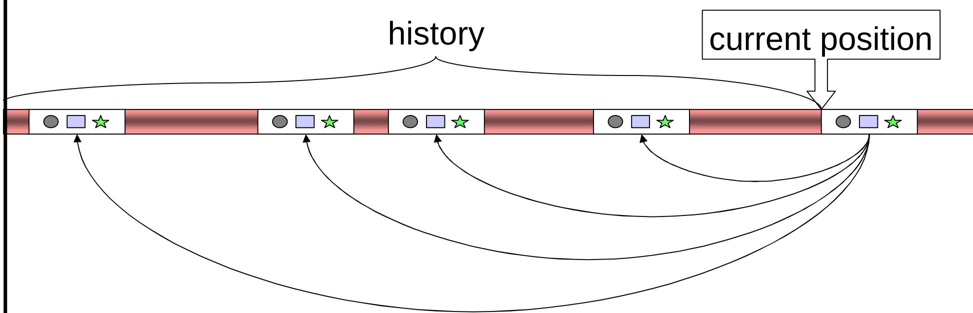

The LZ processing induces a parsing of *T* into *phrases*


Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside


# Watermarking LZ'77

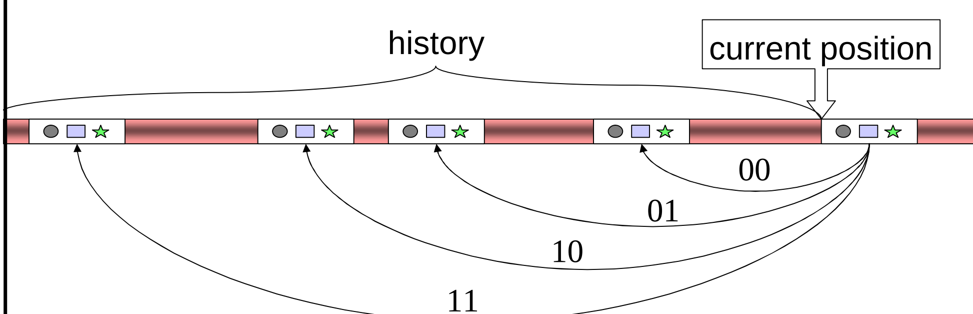

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Which of these pointers do we choose?

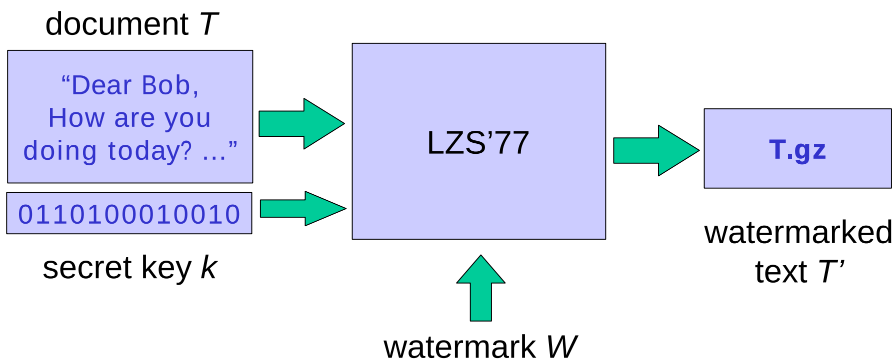



Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

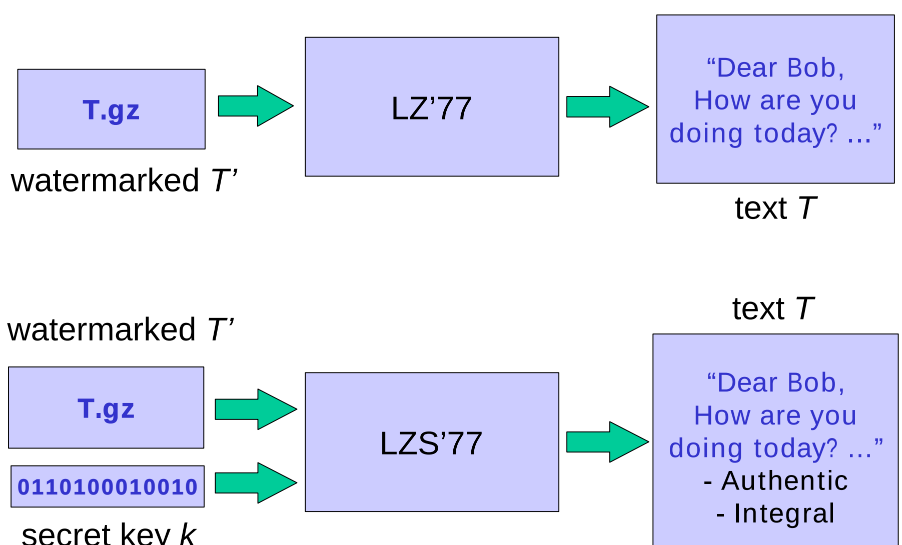
By choosing one of these pointers we are "hiding" two extra redundant bits. Note that we are not changing LZ'77




Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside




Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside




Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

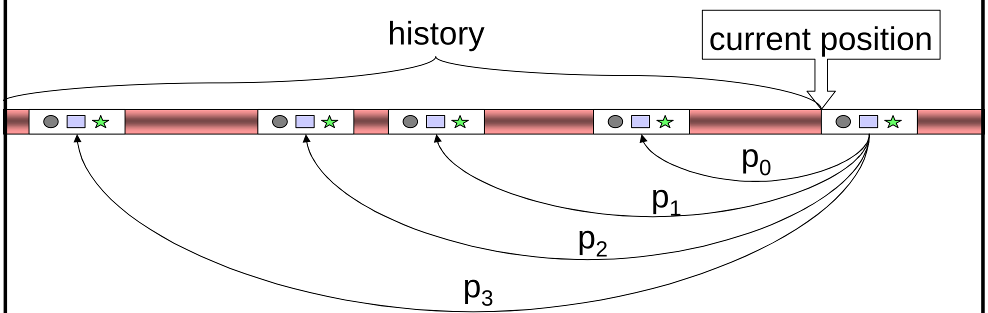
# Method

# Multiplicity

- Definition: a position *i* in the text *T* has *multiplicity* *q* if there exists exactly *q* matches of the longest prefix of *T[i,n]*
- Given a position with multiplicity *q*, we denote by $p_0, p_1, \ldots, p_{q-1}$ the *q* choices for the pointer
- We can embed about $\lfloor \log_2 q \rfloor$ bits

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Multiplicity $q=4$





Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside


# Encoding

- For each phrase $i$ with multiplicity $q>1$
  - Initialize the seed of a random number generator with $H(k,i,p_0,p_1,\ldots,p_{q-1})$
  - Generate a uniformly distributed random permutation $R$ of the set $\{0,1,\ldots,q-1\}$
  - Reorder the pointers based on $R$, *i.e.*, $p_{R[0]}, p_{R[1]}, \ldots, p_{R[q-1]}$
  - Assign each pointer $p_{R[i]}$ the binary code $i$
  - Choose the pointer which binary code matches with the next bits of $W$


Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Security

- Recovering the watermark is at least as hard as breaking the pseudo-random generator
- Finding the key requires to be able to invert a one-way hash function

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Security

- Using some crypto-secure RNG, like BBS [Blum, Blum, Shub 86], the pseudo-random sequence *cannot* be reproduced in a reasonable amount of computing time without the knowledge of the seed $H(k,i,p_0,p_1,\ldots,p_{q-1})$

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Experiments

# Prototype

- We implemented a suffix tree-based LZ'77
- We measured
  - the numbers of bits embedded vs. the length of the text
  - the multiplicity of pointers
  - the length of the phrases

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Number of bits embedded

| *# of bits embedded* | *length of the prefix of* `paper2` |
|---|---|
| 128 | 1,149 |
| 256 | 1,692 |
| 1,024 | 4,778 |

| *# of bits embedded* | *length of the prefix of* `progc` |
|---|---|
| 128 | 863 |
| 256 | 1,729 |
| 1,024 | 4,401 |

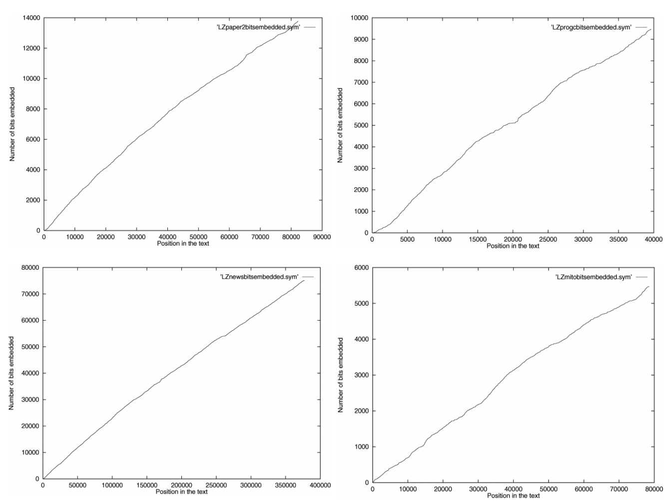
| *# of bits embedded* | *length of the prefix of* `news` |
|---|---|
| 128 | 1,115 |
| 256 | 1,825 |
| 1,024 | 5,195 |

| *# of bits embedded* | *length of the prefix of* `mito` |
|---|---|
| 128 | 1,488 |
| 256 | 3,078 |
| 1,024 | 14,310 |

**Remark:** more bits can be embedded relaxing the greediness


Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

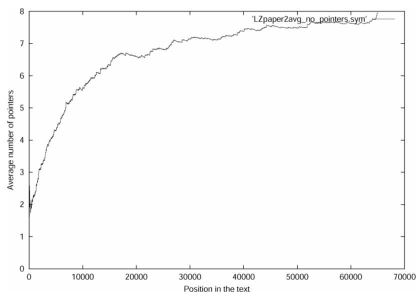

# Number of bits embedded

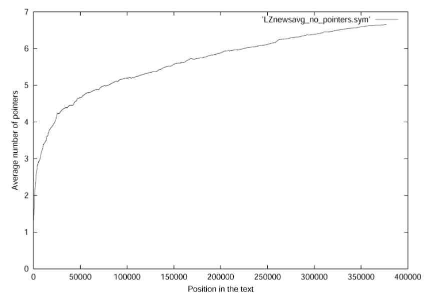




Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# Average multiplicity




Theorem: The average multiplicity ? *O(1)*, as *n? 8* (DCC'03)

**Stefano Lonardi**
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# gzip

- Open source implementation of LZ'77
- gzip issues pointers in a sliding window of 32KB (typically)
- The length of phrases is represented by 8 bits (3-258)
- Phrases smaller than 3 symbols are encoded as literals

**Stefano Lonardi**
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# gzip

- gzip always chooses the most recent occurrence of the phrase
- We modified gzip-1.2.4 to evaluate the potential degradation of compression performance due to changing the rule of choosing always the most recent occurrence
- As a preliminary experiment, we simply chose one pointer at random

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

# gzip vs. gzipS

| *file size* | gzip | gzipS | *file* | *bytes embedded* |
|---|---|---|---|---|
| 111,261 | 39,473 | 39,511 | bib | 1,721 |
| 768,771 | 333,776 | 336,256 | book1 | 14,524 |
| 610,856 | 228,321 | 228,242 | book2 | 10,361 |
| 102,400 | 69,478 | 71,168 | geo | 4,101 |
| 377,109 | 155,290 | 156,150 | news | 5,956 |
| 21,504 | 10,584 | 10,783 | obj1 | 353 |
| 246,814 | 89,467 | 89,757 | obj2 | 3,628 |
| 53,161 | 20,110 | 20,204 | paper1 | 937 |
| 82,199 | 32,529 | 32,507 | paper2 | 1,551 |
| 46,526 | 19,450 | 19,567 | paper3 | 893 |
| 13,286 | 5,853 | 5,898 | paper4 | 249 |
| 11,954 | 5,252 | 5,294 | paper5 | 210 |
| 38,105 | 14,433 | 14,506 | paper6 | 738 |
| 513,216 | 62,357 | 61,259 | pic | 3,025 |
| 39,611 | 14,510 | 14,660 | progc | 736 |
| 71,646 | 18,310 | 18,407 | progl | 1,106 |
| 49,379 | 12,532 | 12,572 | progp | 741 |
| 93,695 | 22,178 | 22,098 | trans | 1,201 |

336,256-
333,776=
-----------
2,480

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

## Conclusions

- Authenticity and integrity for LZ'77 files can be obtained efficiently and elegantly
- The degradation of the compression due to the embedding is almost negligible (about 2% when re-shuffling randomly *all* pointers)

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

## Some open problems

- About LZ'77
  - Can we design a steganography system for it?
  - Can we design a robust watermarking method for it?
- What about the other types of lossless compression?

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside