Part II
Authentication Techniques
Authentication Codes

- Provides means for ensuring integrity of message
- Independent of secrecy - in fact sometimes secrecy may be undesirable!

[Diagram showing the process of authentication codes with nodes labeled Alice (Transmitter), Oscar, Bob (Receiver), and Authentic? with arrows representing the flow of X, Y, Y', and X']
Techniques for Authentication

- Achieved by adding redundancy
  - authenticator, tag, etc., or
  - structure of message

- In some sense like Error Correcting Codes

- Private Key - Public Key  \(<=>\) Authentication - Digital Signature
  - Digital Signatures also provide origin authentication.

- Attacks
  - Substitution
  - Impersonation
  - Choice of above
Authenticating Multimedia Content

Proliferation of digital multimedia content and Ease with which digital content can be manipulated

Need for multimedia (image) authentication.

- Is the problem any different from traditional authentication?
- Some new issues do exist.
Examples of hash functions used for digital signatures are:

- 20-byte **secure hash algorithm (SHA-1)** that has been standardized for government applications.
- 16-byte **MD2, MD4, or MD5** developed by Rivest.

![Diagram of hashing function and image hash](image)
A **digital signature** is created in two steps:

- A fingerprint of the image is created by using a one-way hash function;
- The hash value is encrypted with the private key of a public-key cryptosystem. Forging this signature without knowing the private key is computationally infeasible.
Digital Signature Verification

Verification Software

Public Key

Image being authenticated

11001011100001010010
01000010100100100101
10000111101010010
10001010001………….

Hashing function

Hash of the original image

1000101000
0101000010

Hash of the image in question

1000101000
0101000010

Decryption

Digital Signature

1000101000
0101000010

Yes or No
Digital Signature Authentication

Original

Hash

Private key

Digital Signature

Fail!

Forgery

Public key
New Issues

- Authentication of "content" instead of specific representation -
  » Example - JPEG or GIF image.
- Embedding of authenticator within content
  » Survive transcoding
  » Use existing formats
- Detect local changes
  » Simple block based authentication could lead to substitution attacks
- Temporal relationship of multiple streams
Authentication Using Digital Watermarks

- A number of authentication techniques based on digital watermarks proposed in literature.
- A Digital watermark is a secret key dependent signal “inserted” into digital data and which can be later detected/extracted in order to make an assertion about the data.
- A digital watermark can be
  » Fragile
  » Robust
Fragile Watermarks

Detests and localizes any change to watermarked images.
Authentication Watermark by Wong

Block of watermark Logo

User key $K$

Image width $M$

Image height $N$

Compute Hash $H(K, M, N, \tilde{X}_r)$

Set LSB's to Zero

Output Block $X'_r$

Insert $C_r$ into LSB of $\tilde{X}_r$
Limitations of Fragile Watermarks

- Essentially same as conventional authentication – authenticate representation and not “content”.

- The differences being –
  - Embed authenticator in content instead of tag.
  - Treat data stream as an object to be “viewed” by an human observer.
  - Computationally efficient?
Content-based Authentication

- Number of techniques proposed -
  - Schneider and Chang (1996)
  - Kundur and Hatzinakos (1998)
  - Xie and Arce (1998)
  - Fridrich (1998)
  - Fridrich (1999)
  - And many more in subsequent years …
Feature Authentication

Image → Feature Extraction → Hash → Encrypt → Authenticator

Embed in perceptually irrelevant part of image

Private Key
Feature Authentication (contd.)

1. **Image** → **Feature Extraction**
2. **Feature Set** → **Hash** → **Same?**
   - **Hash Value**
   - **Hash of Feature Set of Original**
3. **Authenticator** → **Decrypt** → **Public Key**
4. **Yes, Authentic**
5. **No, Not Authentic**
Visual Hash Generation (Fridrich 99)

- Low frequency DCT coefficients of an image cannot be changed without changing the image itself.
- Projection onto N random smooth patterns.

```
\text{Image Block (B)} \rightarrow \text{Projection(D)} \rightarrow \text{D > 0 ?}
```

Yes

```
\text{i^{th} Hash bit = 1}
```

No

```
\text{i^{th} Hash bit = 0}
```

\text{Secret key}
Overview of visual hash based image authentication system

- We have shown that collisions can be generated for the underlying hash function. (RXM 2002).
## Performance of Fridrich’s Algorithm

<table>
<thead>
<tr>
<th>Forgery attack</th>
<th>Probability of miss $P_m$</th>
<th>Signal processing attacks</th>
<th>Probability of false alarm, $P_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>d B</td>
<td>Substitution</td>
<td>No attack</td>
<td>Smoot h</td>
</tr>
<tr>
<td>38</td>
<td>0.6%</td>
<td>1.1%</td>
<td>98.9%</td>
</tr>
<tr>
<td>41</td>
<td>1.0%</td>
<td>1.6%</td>
<td>98.3%</td>
</tr>
</tbody>
</table>

And this was the best performance from a large collection of schemes studied by Sankur et. al.!!
Limitations of feature authentication

- Difficult to identify a set of definitive features.
- Set of allowable changes has no meaningful structure – certain “small changes” may not be allowed but the same time “large” changes may be allowed in other situations.
- “Strong” features facilitate forgeries.
- “Weak” features cause too many false alarms.
Difficulties with content authentication of images

- Content is difficult to quantify.
- Malicious (benign) modifications are difficult to quantify.
- Images considered as points in continuous space means there is not a sharp boundary between authentic and inauthentic images.

authentic inauthentic

authentic and inauthentic images which are similar to each other
Distortion Bounded Authentication

- Problem 1: allow flexibility in authentication to tolerate small changes
- Problem 2: to characterize and quantify the set of allowable changes
  - Bound the errors
  - Perceptual distortion or pixel value distortion
- Provide “guarantees” against substitution attacks.
- Approach – bounded tolerance authentication
  - (semi-fragile)Watermarking techniques offer flexibility but most do not offer bounds
Distortion Bounded Authentication

- Quantize image blocks or features prior to computing authenticator.
- Quantization also done prior to verifying authenticity of image.
- Enables distortion guarantees – image considered authentic as long as change made does not cause quantized version to change.
- Can be used in many different ways
Distortion bounded authentication – example.

- **Private Key K'**
- **Image width M**
- **Image height N**
- **Block of watermark Logo**
- **Compute Hash**
  \( H(K, M, N, \tilde{X}_r) \)
- **Quantize block**
- **Set LSB’s to Zero**
- **Encryption**
  \( W_r \)
- **Insert** \( S_r \)
  into LSB of \( X_r \)
Limitations

- Distortion added to “original” image.
- Similar problems as feature authentication, though to a lesser degree.
- Significant changes may indeed be possible within specified set of allowable changes.
- How to define set of allowable changes?
A Better Approach?

Chai Wah Wu - 2000

Fuzzy region: authenticity of image is uncertain.
A framework of content-based image authentication

• Given a source image I, an authentication tag T is generated from I. Tag T is much smaller than I (data reduction)
• I is changed to I' in order to make it authenticatable (authenticability distortion)
• T is appended to I' resulting in an authenticatable image (I', T).
Authentication of \((I', T)\)

- \(T\) is extracted from \((I', T)\).
- A second tag \(T'\) is computed from \(I'\).
- \(T\) and \(T'\) are compared. If they compare favorably \((R(T, T') \text{ is true})\), image is authentic.
- Examples of \(R(T, T')\):
  - \(d(T, T') < \varepsilon\)
  - \(T = T'\)
Some parameters to optimize

- $D$, the maximum authenticability distortion.
- The size of tag $T$ compared to the size of source image $I$.
- Parameters $\beta_a$ and $\beta_m$: given $(I', T)$ generated by system, if $|x| < \beta_a$, then $(I'+x, T)$ is authentic. If $|x| > \beta_m$, then $(I'+x, T)$ is inauthentic.
- Size of fuzzy region is determined by $\Delta \beta = \beta_m - \beta_a$. 
Fuzzy region: authenticity of image is uncertain.
Extract features and check for similarity

• Tag is generated from some inherent features of image such as the location of edges.
• Given \((I',T)\), a second tag \(T'\) is calculated from \(I'\), and image is authentic if \(d(T,T') < \varepsilon\) for some metric \(d\). Therefore similar images should generate similar tags (smoothness).
• Authenticability distortion \(D\) can be small.
Extract features and check for similarity

- Data reduction implies existence of forged images. Given a map $f$ from $[0,1]^n$ to $[0,1]^m$, $n \gg m$, there exists points $x, y$ such that $d(x, y)$ is close to 1, but $d(f(x), f(y))$ is arbitrary small.

- Smoothness in generating the tags indicate a lack of diffusion and can lead to methods to forge images, e.g. edges do not contain color information. In cryptography, preimage resistant functions generally avoid such smoothness.

- $\beta_a$ and $\beta_m$ difficult to determine.
Generate hash and check for equality

- Authenticability distortion is applied to source image I to generate I'.
- Tag T is generated from I' by a cryptographic hash/digital signature scheme.
- To authenticate (I', T), generate T' from I', and image is authentic if T = T'.
Generate hash and check for equality

- Inherit security from cryptographic hash and digital signatures.
- One can prove that $D \geq b_a$. Thus authenticability distortion is at least as large as the benign distortion that can be tolerated. In Memon et al., $D = b_a$.
- But, some applications require $D=0$ or $D < b_a$. 
Hash-based image authentication with no authenticability distortion

• Robustness to minor modifications using quantization functions.
• Utilize multiple quantization functions.
• Each point is quantized by a quantization function whose quantization boundaries are away from the point.
• The choice of quantization function is stored in an index vector.
• Quantized value is used to generate tag.
Hash-based image authentication with no authenticability distortion

Choice of q is stored in one bit of index vector. With the proper q, small changes will not affect the quantized value and the tag.
Generate authenticatable image

- Select appropriate quantization functions for each data point.
- Store choice of quantization functions in index vector.
- Quantize data point with chosen quantization function.
- Sign quantized data + index vector.
- Append lossless compressed index vector to signature and form tag.
Authentication of images

- Extract index vector and signature from tag.
- Choose quantization functions according to index vector.
- Quantize data point with chosen quantization function.
- Verify signature against quantized data + index vector.
Example

Lena RGB image in TIFF LZW format: 646KB.
Authentication tag (index vector + signature): 5.6KB.
Authenticatability distortion $D = 0$.

After JPEG compression: image is **authentic**.
After minor brightening of image: image is **authentic**.
Extra strands added to hat area: image is **inauthentic**.
Features of the scheme

- Inherit security from cryptographic hash and digital signatures. Finding forged images which are $\beta_m$ away from the original is at least as hard as breaking the underlying digital signature scheme.
- D can be smaller than $\beta_a$.
- $\beta_a$ and $\beta_m$ can be explicitly determined.
- Tradeoff D against size of authentication tag T.
- Tradeoff D against $\beta_a$ and $\Delta \beta$. 
Flowchart of procedure to generate authenticatable data

1. Source data \( I \)

2. Feature vector generation
   - Choose quantization functions and generate index vector \( V \)
   - Lossless compression of index vector \( X \)

3. Digital signature generation
   - Append index vector to quantized feature vector \( W \)

4. Tag generation
   - Append/insert tag into \( I' \)
   - Apply authenticability distortion

5. Authenticatable data \( I_a \)
Flowchart of Authentication Procedure

1. Extract authentication tag from T
2. Extract signature from tag
3. Extract compressed index vector from tag
4. Decompression
5. Quantization of feature vector V according to X
6. Append X to quantized feature vector
7. Signature verification
   - success
   - failure
8. Data is authentic
   - Data is not authentic
Key Management

- In watermarking applications, especially authentication, it is convenient to
  - Use same key for watermarking different images or
  - Use same key to insert different watermarks in different images.

- We have shown on multiple occasions that above can be insecure, depending on watermarking technique and watermark inserted.
Image Dependent Key

- Attacks are possible only because same key used for seeding the pseudo-random sequence for each image.
- Can be avoided if we use different keys for different images.
- This can lead to key management problem.
- We have proposed two solutions
  - Image dependent key - Derive key from bits extracted from image itself.
  - Unique Salt for each image – Salt used to derive master secret. Salt stored in the clear.
Summary and Conclusions

- Multimedia content poses some new challenges for design of authentication techniques.
- We need a formal framework similar to conventional authentication.
- Bounded tolerance – trading off flexibility and quantifiable errors (comfort zone) is a good approach.
- Many schemes become vulnerable with same key used to mark multiple objects. Image dependent keys or salting offer good mechanisms to prevent such attacks.