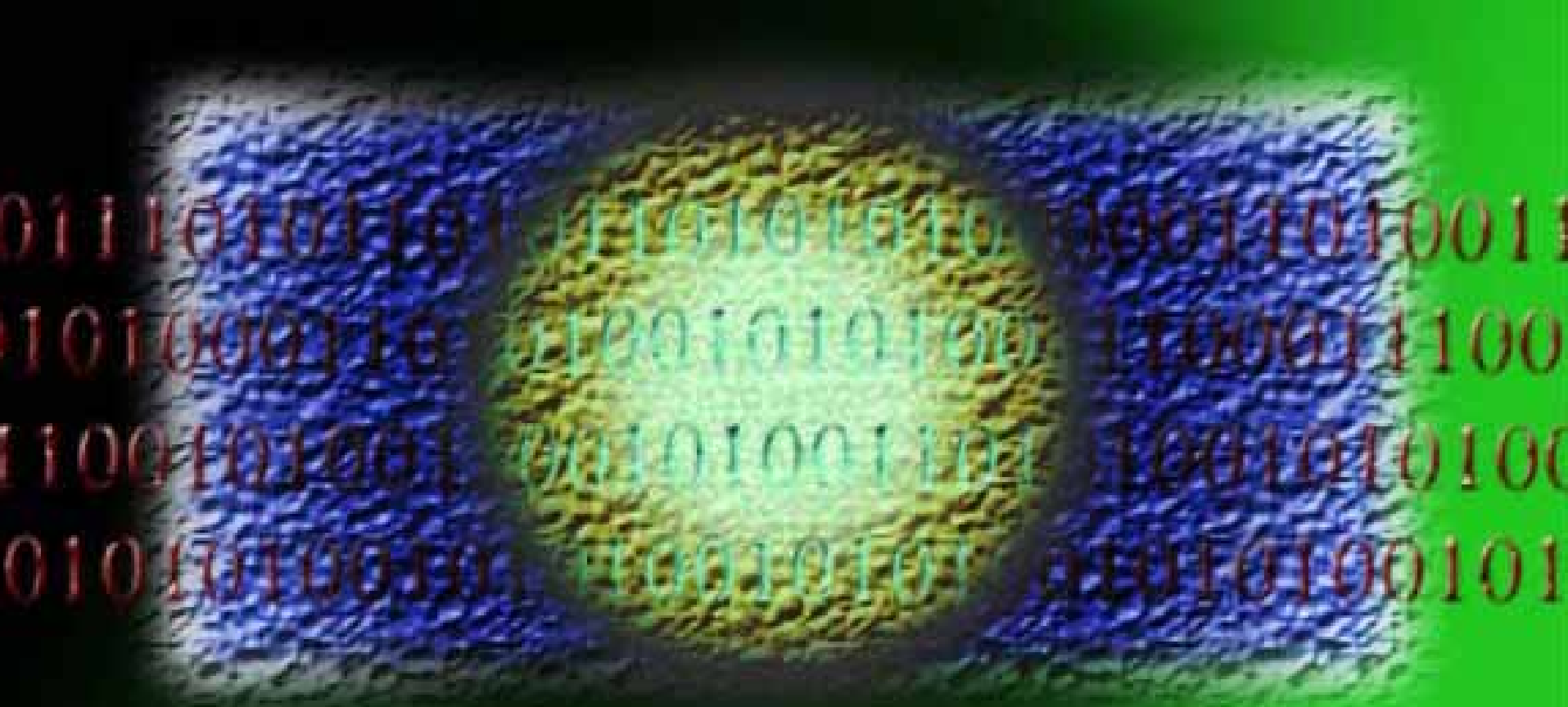


Team ISE

Image Selective Encryption

Team ISE

Image Selective Encryption

Joe Jarchow

Shinya Daigaku

Joseph Kadhim

Andrew Pouzeshi

Geoffrey Griffith

Sponsor

- **Tom Lookabaugh**
  - **Assistant Professor in Computer Science Department**
  - **Faculty Director of Interdisciplinary Telecommunications**
  - **Research into areas such as**
    - **Selective Encryption**
    - **Broadband**
    - **Multimedia and Distance Learning**

Joe J

Problem:

- **Multimedia files are often very large**
- **Encrypting can require extensive processing time**
- **Can also increase the file size**
- **No current intelligent method for securing multimedia information without encrypting an entire file**

Joe J

Solution:

- **Selective Encryption**
- **Team ISE**

Joe J

Definition of Selective Encryption:

- **Selective encryption applies encryption to a subset of a file with the expectation that the entire file will be rendered useless to anyone who cannot decrypt that subset.**

Joe J

Successful Selective Encryption:

- **The right subset must be chosen or the file may not very secure**

Example:

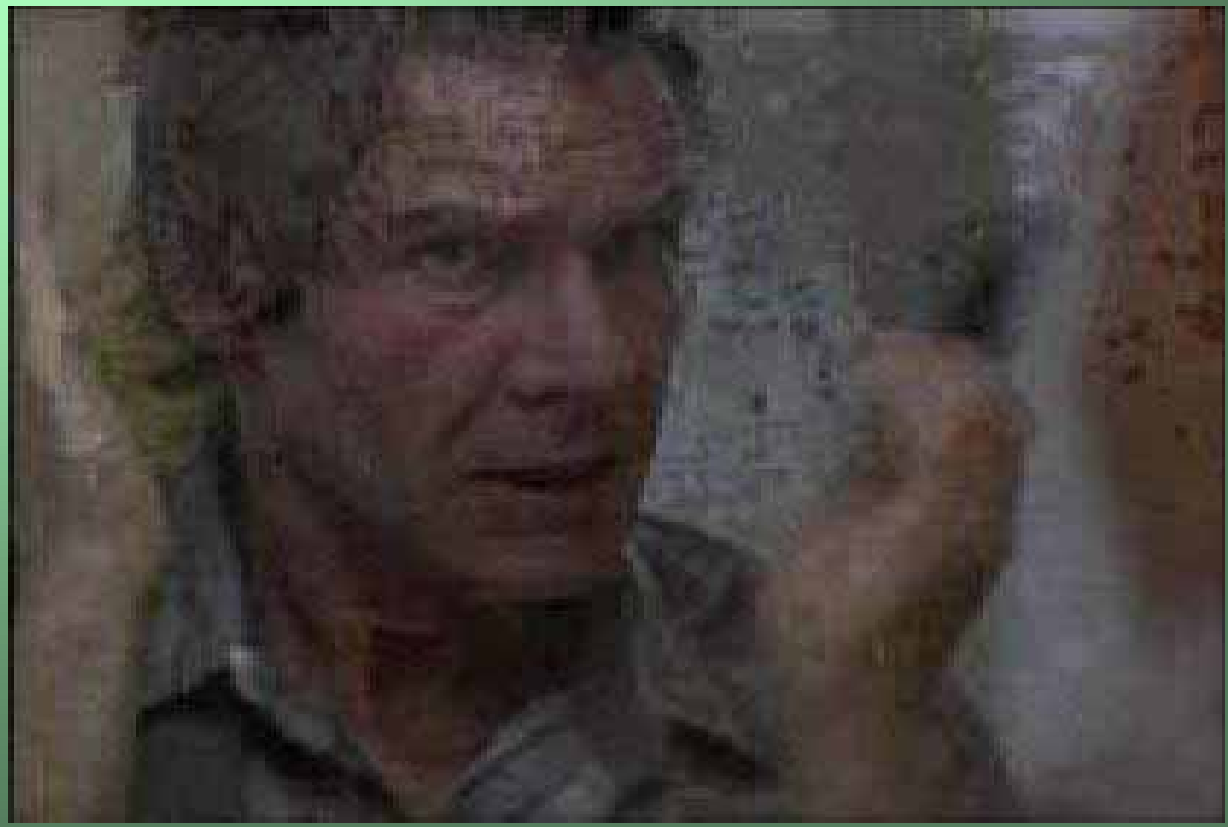
```
For ixstanxe, obsxurinx
everx fiftx lettxr of ax
Englxsh sextencx does xot
maxe it pxrticxlarlx hard
xo read.
```

Joe J

Sponsor's current research into MPEG

- Target for degraded image rather than secretive
- Encryption of less than 10% of file
- Allow only a small reduction in efficiency (nominal increase in bandwidth)
- Provide solution to the cryptography community for review and testing

Joe J



Joe J

Joe J

Team ISE Presentation Outline:

- General introduction to compressed media
- Requirements for JPEG image approach
- Architecture and Design
  - C++ Production Code
  - Website
- Demonstrate research
  - C# JPEG Manipulator

Joe J

Compressed Multimedia Overview

Shinya

Multimedia file examples:

- **MPEG 1, 2, and 4 video**
- **MP3 (MPEG 1 Layer 3)**
- **JPEG, zip and voice**

Each uses a standard compression scheme

Multimedia File Components

- **Compressed files are partitioned into standard pieces**
- **Some parts are *Descriptive***
- **Others are *Mathematical***
- **These components are referred to as frames within a compressed media file**

Shinya

Frame:

- **Consists of a marker, header and data**
- **Example frame (piece of a JPEG file):**

```
ff e0
00 10
4a 46 49 46 00 01 01 01 00 48 00 48 00 00
```

Shinya

Marker:

- **Indicates what kind of data follows.**
- **Example marker:**

ff e0 (indicates Application Data)

Shinya

Header:

- **Describes the data to follow**
- **Size, parameters, descriptor, etc.**

- **Example header:**

00 10 -- (16 bytes of data will follow)

Shinya

Data:

- **The information itself**
- **Example data:**

4a 46 49 46 00 01 01 01 00 48 00 48 00 00

(16 bytes of information indicating what application created the file.)

Shinya

Real cryptography approaches a solution from both a white hat and black hat view

- **White hat -- Designs a secure system**
- **Black hat -- Attempts to break into the system**

Closed encryption method

- **NSA has been faulted in not publishing their algorithms**
- **Only people on the inside actually know the algorithm**

Open encryption method

- **Algorithm is published publicly**
- **People not involved in producing the algorithm can review and attempt to break the encryption**
- **If the encryption is broken, we can improve the algorithm**

Shinya

Why do we want to use selective encryption?

Drawbacks to encrypting an entire file

- **Takes time to encrypt the data**
- **Sometimes makes the file bigger**

Drawbacks to selective encryption

- **Slightly more complex than encrypting an entire file**
- **Have to find the right target to encrypt**

Shinya

- **Many multimedia compression algorithms concentrate critical information in a small portion of the bit stream**

- **Encrypting this portion could render the remaining information useless**

- **Selective encryption involves selecting which pieces of a bit stream to encrypt in order to minimize the amount of encryption applied and maximize the amount of damage**

JPEG Specification and Approach

Joseph K

JPEG Selective Encryption

- JPEG is a compressed image format consisting of frames (markers, headers, data)

- Through a process of analysis, we were able to find a target appropriate for successful selective encryption

- We are only required to handle Baseline JPEG Images, the most commonly used compression mode for JPEGs

Criteria For Bad Targets

- **Optional markers**
- **Markers not used in Baseline JPEG images**
- **Markers that contain information that does not affect visibility of the image**
- **Markers that contain information that can be easily guessed or forged by a hacker**

Markers immediately eliminated:

- **APP - Application**
- **COM - Comments**
- **DAC - Define Arithmetic Conditioning Tables(Not part of Baseline Compression)**
- **DHP - Define Hierarchical Progression (Not part of Baseline Compression)**
- **DNL - Define Number of Lines**
- **DRI - Define Restart Interval**
- **EOI - End of Image**
- **EXP - Expand (Not part of Baseline Compression)**

Markers immediately eliminated:

- **JPG - Reserved for Future Extensions**
- **RES - Reserved**
- **RST - Restart (Not part of Baseline Compression)**
- **TEM - Temporary (Not used in Baseline Compression)**

- **Markers themselves are predictable**
- **Scan Header easily reconstructed**

Remaining Target List for Selective Encryption

- **Encoded Data Stream**
- **Quantizer Tables**
- **Huffman Tables**

Target Analysis

- Two C++ programs were written for target analysis:

    Convert and Analyze

    Convert
    - Binary to Hexadecimal
    - File information for a single JPEG image

Joseph K

This is an ASCII representation (in hexadecimal) of the binary values found in the file : Dust.jpg

Markers Found:===============

ff d8 -- Start of Image

ff e0 -- Application Data -- 00 10 -- (16 bytes) -- 4a 46 49 46 00 01 01 01 00 48 00 48 00 00

ff db -- Define Quantization Table -- 00 43 -- (67 bytes) -- 00 06 04 05 06 05 04 06 06 05 06 07 07 06 08 0a 10 0a 0a 09 09 0a 14 0e 0f 0c 10 17 14 18 18 17 14 16 16 1a 1d 25 1f 1a 1b 23 1c 16 16 20 2c 20 23 26 27 29 2a 29 19 1f 2d 30 2d 28 30 25 28 29 28

ff db -- Define Quantization Table -- 00 43 -- (67 bytes) -- 01 07 07 07 0a 08 0a 13 0a 0a 13 28 1a 16 1a 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28

ff c0 -- Huffman Table -- Baseline DCT -- 00 11 -- (17 bytes) -- 08 01 cb 02 4a 03 01 22 00 02 11 01 03 11 01

ff c4 -- Huffman Table -- 00 1f -- (31 bytes) -- 00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09 0a 0b

Joseph K

Analyzer

- Compute file information for multiple JPEG images
- Average file size
- Average number of Huffman tables
- Average size of Huffman tables combined
- Average number of Quantizer tables
- Average size of Quantizer tables combined

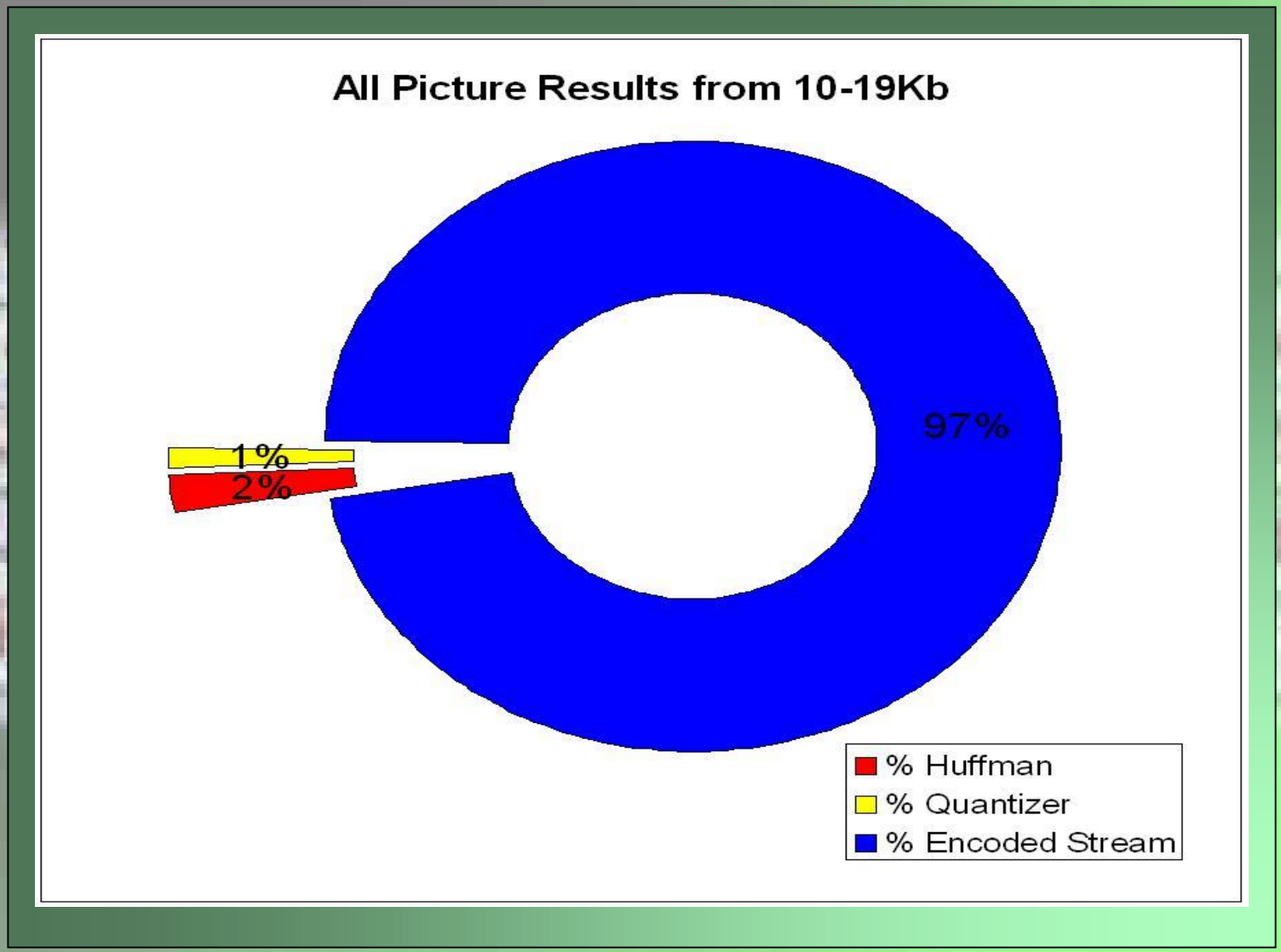
Analyzer

- Average size of the encoded stream
- Average number of markers
- Number of files processed

- Percent of the file dedicated to:
  - Huffman tables
  - Quantizer tables
  - Encoded Stream

Test Cases for JPEG Analysis

- Over 2500 JPEG images randomly selected from the Internet
- Digital Photograph vs. Manmade
- Size ranges: 10-19KB, 100 KB, 1 MB, and larger
- Resolution Ranges: 320x240, 640x480, and 800x640 Pixels


All Picture Results from 10-19Kb
97%
1%
2%
% Huffman
% Quantizer
% Encoded Stream

Joseph K

Encoded Data Stream

- SOI (start of image) marker
- Compressed data stream
- Takes up a large portion of the file
- Averaged 90% of the file!

Quantizer Tables

- DQT (Define Quantization Table) markers
- Averaged 0.88% of the file
- Unpredictable affects on image
- Might not visually damage the image at all!
- Could be replaced with another Quantizer table!

Huffman Tables

- DHT (Define Huffman Table) markers
- Averaged 1.84% of the file
- Considerable damage to image
- Mathematically derived from the image
- This makes the Huffman Tables a perfect target for Selective Encryption

Architecture and Design

Andrew

ISE ARCHITECTURE

- Invocation
- ISE Class Inheritance
- User Interface
- High-Level Modular Decomposition
- File Description

Andrew

ARCHITECTURE: INVOCATION

- Test Suite
  - Application designed to aid the research into JPEG images
  - Easy to test the effects of manipulating each type of frame
- Production Code
  - Final release of C++ package implementing selective encryption for successful targets

Andrew

ARCHITECTURE: INVOCATION

Test Suite:

- **Invoked as a windowed Version 1.1 .NET application**

Andrew

ARCHITECTURE: INVOCATION

Production Code:

- **ISE ENCRYPTOR and ISE DECRYPTOR**
  - **Input File Name and Path**
  - **Output File Name and Path**
  - **Encryption Key**
  - **Flags**

Andrew

ARCHITECTURE: ISE CLASS INHERITANCE

ISE Super Class

- The ise class will define our process and will be inherited by our specific encryption classes:

```
class ise
 {
     public:
     virtual int selectiveEncryption(. . .) = 0;
     virtual int selectiveDecryption(. . .) = 0;
};
```

Andrew

ARCHITECTURE: ISE CLASS INHERITANCE

JPEG Encryptor Class

- The jpeg_file class is the immediate goal of project ISE

```
class jpeg_file : public ise
{
    public:
    virtual int selectiveEncryption(. . .);
    virtual int selectiveDecryption(. . .);
};
```

Andrew

ARCHITECTURE: ISE CLASS INHERITANCE

MP3 Encryptor Class

- This class will be an extension to the project.

```
class mp3_file : public ise
{
  public:
    virtual int selectiveEncryption(. . .);
    virtual int selectiveDecryption(. . .);
};
```

Andrew

ARCHITECTURE: USER INTERFACE

- Test Suite
- Production Code

Andrew

ISE JPEG Manipulator - Version 1.0.1 - C:\Dust.jpg

File Help

New Project
Open Project
Save Project
Open Picture
Update Picture
Exit

Picture | Manipulated Picture

Project | File Information | Encoded Data | Quantizer Table | Huffman Tables 1 | Huffman Tables 2 | Application Data | Misc

Huffman 1: ffc0
08 01 cb 02 4a 03 01 22 00 02 11 01 03 11 01

Huffman 2: ffc4
00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00 00
01 02 03 04 05 0f 0f 0f 0f 0f 0f

Original 1:
Restore

Original 2: ffc4
00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00 00
01 02 03 04 05 06 07 08 09 0a 0b
Restore

Huffman 3: ffc4
10 00 02 01 03 03 02 04 03 05 05 04 04 00 00 01 7d 01
02 03 00 04 11 05 12 21 31 41 06 13 51 61 07 22 71 14

Huffman 4: ffc4
01 00 03 01 01 01 01 01 01 01 01 01 00 00 00 00 00 00
01 02 03 04 05 06 07 08 09 0a 0b

Original 3:
Restore

Original 4:
Restore

ARCHITECTURE: ISE USER INTERFACE

Production Code:

- Utilized through API
  - Encryptor API

```
int selectiveEncryption(
    ifstream &input_file_stream,
    ofstream &output_file_stream,
    char* key_material,
    char* encryption_flags,
    int num_flags, int quality);
```

Andrew

ARCHITECTURE: ISE USER INTERFACE

Production Code:

- **Utilized through API**
  - **Decryptor API**

```
int selectiveDecryption(
    ifstream &input_file_stream,
    ofstream &output_file_stream,
    char* key_material,
    char* encryption_flags,
    int num_flags, int quality);
```

Andrew

FILE DESCRIPTIONS

Input Files:

- **Encryptor:**
  - **JPEG file with ‘.jpg’ extension**
- **Decryptor:**
  - **Encrypted file with ‘.ise’ extension**

Andrew

FILE DESCRIPTIONS

Output Files:

- **Encryptor:**
  - **Encrypted file ending in '.ise' extension**
- **Decryptor:**
  - **JPEG file with '.jpg' extension**

Andrew

ISE WEBSITE

- Website is temporarily Available at:
  - ucsub.colorado.edu/~pouzeshi/ISE
- Includes links to:
  - Sponsor
  - Documentation
  - Test Suite/Production Code
  - Other Relevant sites
  - Team ISE info

Andrew

PROJECT EXTENSIONS/CODE MODIFICATION

Compression Type Extension:

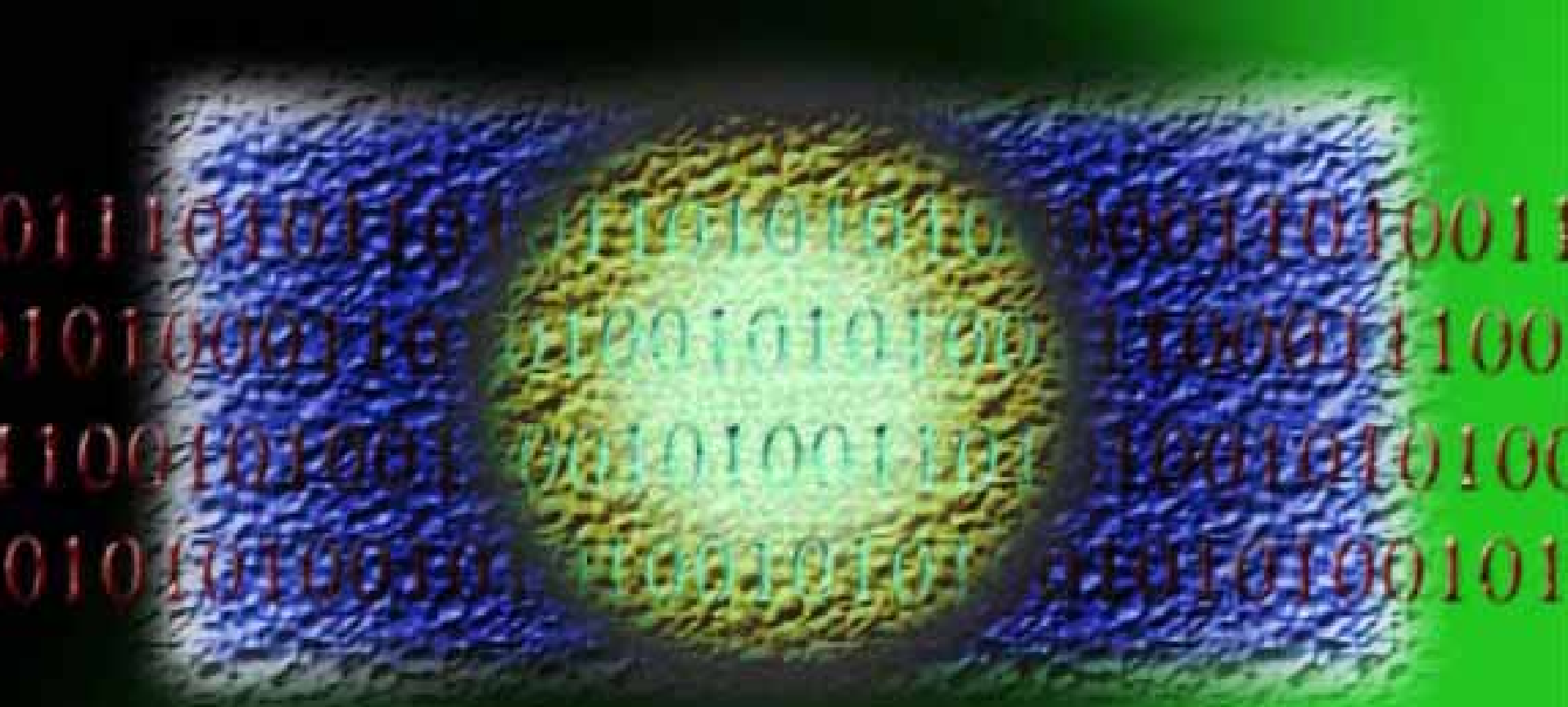
- **Time Permitting, the project will be extended to include:**
  - **MP3 Compression**
  - **ZIP Compression**

- **The Modular Design of the JPEG selective encryptor makes these extensions possible.**

Andrew

Manipulator Demonstration:

Geoff



Team ISE

Image Selective Encryption