肖像権侵害抑止を目的とした画像配布システムの試作

宮元　章*・河野 哲也**・堤 祐樹***・脇山 正博

A prototype on an image distribution system for preventing infringement of portrait rights

Akira MIYAMOTO, Tetsuya KAWANO, Yuki TSUTSUMI, Masahiro WAKIYAMA

Abstract

This paper discusses the making of a prototype on an image distribution system which, using steganography, prevents infringement of portrait rights. In recent years, with the spread of mobile terminals such as smart phones, it is easy to share photos taken by oneself on social networking services. However, there are a significant number of photos taken and uploaded without permission. These infringe the rights of publicity. If we use image-sharing systems such as Picasa and OneDrive, our photos will not be seen of by the general public which thus reduces the risk of photos being diffused. In a situation whereby a malicious user spreads photos, it is difficult to determine the user in question. The current image systems are unable to prevent infringement of portrait rights. Therefore, we propose a means to distribute image files, using steganography, to identify the malicious users.

Keywords : steganography, image distribution system

1. 緒言

昨今，スマートフォン，タブレットPC等のモバイル端末の普及により，個人で撮影した写真をSNSやブログ，ウェブサイトにアップロードすることで簡単に不特定多数の人とその画像を共有することができる。しかし，それらの写真の中には被撮影者の承諾なく撮影・アップロードされたものであったり，無断転用や不正利用されたりするものが少なからず存在する。そうした一旦拡散してしまった写真は完全に削除することは難しく，肖像権侵害に繋がる問題が多く発生している。

そのため，写真共有のための方法の一つとしてPicasa(1)やOneDrive(2)等，予め許可されたユーザのみが画像をダウンロードできるシステムを利用する方法が挙げられる。そのシステムを利用することで不特定多数のユーザから写真を閲覧されることは無く，大量拡散するリスクは低減させることができる。しかし，悪意あるユーザによってこれらの写真を意図的に拡散されてしまったとしてもそのユーザを明らかにすることができず，肖像権侵害抑止としては不十分である。

そこで本研究では，ステガノグラフィを用いて共有写真にユーザ情報を埋め込んだ状態でダウンロードでき，悪意を持ってその写真を拡散したとしてもそのユーザを特定することができる画像配布システムの試作を目的とする。

2. システム概要

2. 1. 画像管理者側の処理

図1に本システムにおける画像管理者側の処理の流れを示す。まず始めに ①カメラ等により写真を撮影する。次に ②撮影した写真をPCに取り込む。その後 ③PCから画像保存用ネットワークアタッチドストレージ（以下，NASと略す）にアップロードする。Webアプリケーションサーバ（以下，アプリケーションサーバと略す）は起動時にNASをマウントするよう設定してある。最後に ④アプリケーションサーバにおいてNAS上の画像を縮小し，画像選択用Webサイトでサムネイルとして利用する画像を作成する。サムネイルを作成することでWebサイトの読み込み時間を短縮できるのみでなく，仮にサムネイルをダウンロードされたとしても，画像自体の解像度が低いため写っている人物の特定は難しく，肖像権の侵害防止としても有効な手段である。

2. 2. 画像を要求するユーザ側の処理

図2に本システムにおける画像を要求するユーザ側の処理の流れを示す。まず始めに ①Googleアカウント認証により画像管理Webサイトにログインし，必要な写真を選択する。次に ②アプリケーションサーバにおいてステガノグラフィの最下位ビット置換法を用い，選択した画像にユーザアカウント・名前等のユーザ情報を埋め込む。その後 ③埋め込んだ画像をGoogle Driveサーバ上のユーザ専用フォルダにアップロードする。そして ④Google Drive上のユーザ専用フォルダを共有し，ダウンロードできるように設定する。その後 ⑤Google側から共有設定が完了し画像をダウンロードできる準備ができた旨のメールをユーザに送信する。最後に ⑥ユーザがGoogle Drive上から選択した画像をダウンロードする。

このシステムは一見すると婉曲的な方法を取っているように思われる。Webサイト上のリンクをクリックした際に画像をダウンロードする方がよほどシンプルである。しかし，本システムではステガノグラフィの最下位ビット置換法を用いてユーザ情報を埋め込むが，それは画像の総てのピクセルに対して行うため膨大な時間を要する。デジタルカメラで撮影した画像の標準サイズは大きくなる傾向にあるため，今後は更に時間を要すると推測される。そのため逐次処理を行うシステムとした。

* 教育研究支援室 機器分析技術グループ
** 制御工学専攻 2年　　*** 電子制御工学科 4年

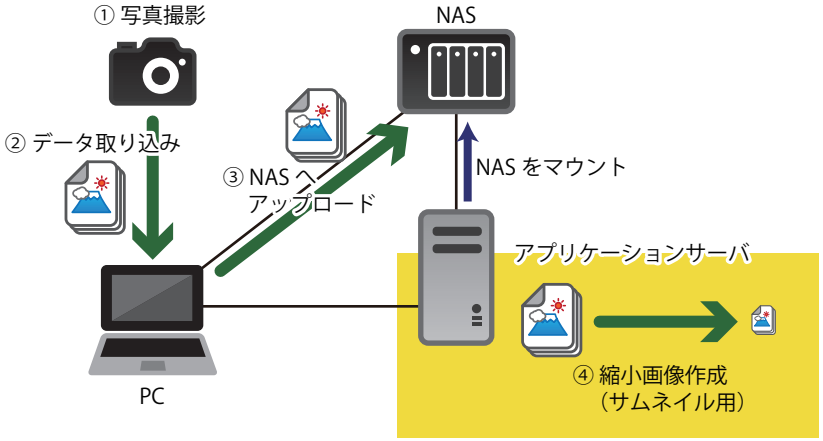

図1　画像管理者側の処理

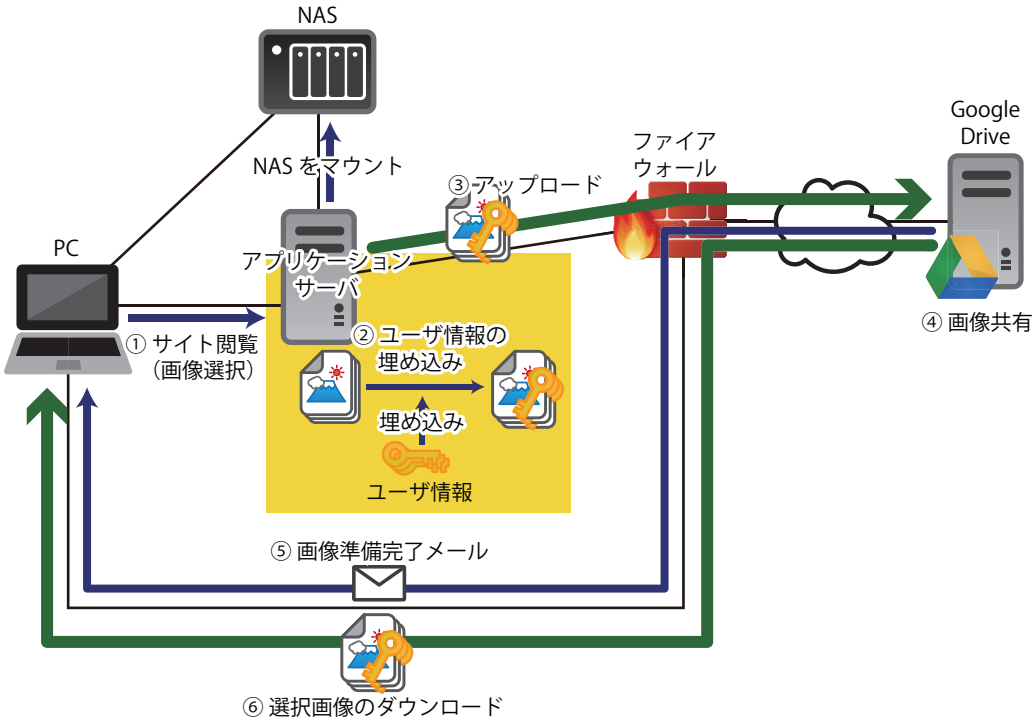

図2　画像を要求するユーザ側の処理

3. ステガノグラフィ

3. 1. ステガノグラフィ技術

ステガノグラフィ(Steganography)[3]とは，情報ハイディング技術の一つで，秘密にしたい情報を別のありふれたものに秘匿する技術である。そうすることで第三者からは秘密情報の存在自体を隠すことができる。

同じく情報ハイディングの1つであるクリプトグラフィ（Cryptography）と大きく異なる点は，クリプトグラフィは見ただけですぐに暗号化していることを悟られてしまうため，無尽蔵に時間をかけることで必ず解読され，秘密にしたい情報を保護することはできない。それに対し，ステガノグラフィは秘密情報が隠されていること自体を第三者に気付かせないようにするものであるため，秘密データを盗まれるという危険が極めて低いと言える。

一般的には，音声や画像などのデータ（カバーデータ）に秘密データ（シークレットデータ）を埋め込み，情報を秘匿したデータ（ステゴデータ）を作成する。このとき，カバーデータにもシークレットデータにも種類の制限はなく，画像，音声，

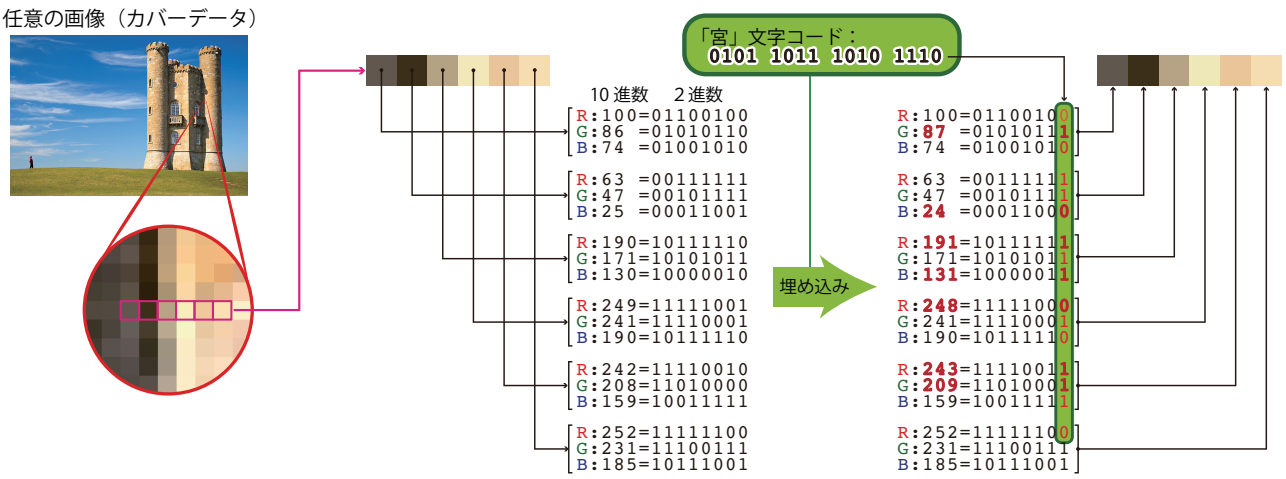

図3　画像ファイルの最下位ビット置換法による埋め込み例

動画，テキストなど，形式を問わず様々なものに埋め込むことが可能である。また，最下位ビットや離散コサイン変換，RAW画像の圧縮を用いるなど様々なアルゴリズムが考案されている。

3．2．最下位ビット置換法

最下位ビット置換法はステガノグラフィの埋め込みアルゴリズムの1つで，カバーデータの各単位の最下位ビットにシークレットデータのビット列を順に埋め込む方法である。

図3に示すようにカバーデータとして24ビット画像を例に取ると，画像の1ピクセルにあるR，G，Bの階調はそれぞれ8ビットで表すことができる。仮に「宮」という文字を埋め込む場合，その文字コード（Unicode）を2進数で表すと，”0101 1011 1010 1110”となる。文字コードの先頭から1ビットずつをR，G，Bの最下位ビットと置換することで埋め込みを行う。実際には複数の文字を埋め込む必要があるため，文字列の最後に終了文字を埋め込むことでシークレットデータを読み出す際の目印となる。最下位のビットのみの変更しか行われないため，データの劣化を少なくすることができる。実際に埋め込まれた画像は，元の画像と比べてもほとんど画像劣化を感じさせることはない。本システムではこの特徴を利用し，ユーザ情報を各ピクセルに埋め込んだ。

4．開発方法

4．1．RubyおよびRuby on Rails

本研究では，主としてステガノグラフィを用いたユーザ情報の埋め込みおよび後述するRuby on Railsの開発を行うためオブジェクト指向のスクリプト言語であるRubyを用いた。RubyにはRubyGemsというRubyの強力なパッケージマネージャが標準で搭載されており，様々なライブラリをインターネット経由で簡単にインストールすることができ，開発コストを大幅に下げることができる。また，ウェブアプリケーションの開発にはオープンソースのWebアプリケーションフレームワークであるRuby on Rails（以下，RoRと略す）を用いた。RoRは厳格なModel-View-Controller（以下，MVCと略す）パターンを採用し，機能ごとの独立性をより確保しながら開発できるよう設計されたフレームワークである。本システムの共同開発においてもそれぞれMVCを考慮した分業を行った。

4．2．Google API

Google APIとはGoogle社が提供するアプリケーションであるGoogle Appsの持つ機能をインターネットを介して外部から利用するための手続きをまとめたAPIである。本研究では，認証部分にGoogle Contacts API，画像を共有するGoogle Drive利用のためGoogle Drive APIを利用した。新規に認証部分や画像共有の仕組みを開発しないためそのコストが大幅に削減できるのみでなく，よりセキュリティが高く効率のよいシステムを利用することができるという利点がある。

5．画像配布システムの操作例

図4(a)～図4(e)に画像配布システムの操作画面例を示す。

5．1．ログイン画面

ログイン画面を図4(a)に示す。Google APIを利用した認証システムのため一般的なGoogleのログイン画面と同一のものと

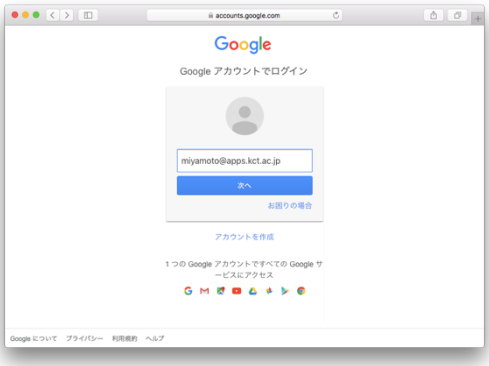

図4(a)　ログイン画面

なる。ユーザ名とパスワードを入力しログインする。

5. 2. 画像配布システムトップ画面

ログイン直後の画像配布システムのトップ画面を図4(b)に示す。必要な写真を選択後，ダウンロード予約を行うことができる。予約後，アプリケーションサーバにより選択画像に対しユーザ情報を埋め込まれる。

図4(b)　画像配布システムトップ画面

5. 3. 画像配布準備完了メール

画像配布準備が完了した後，Googleから届く閲覧への招待メールの画面を図4(c) に示す。メール本文中の[開く]をクリックするとGoogleドライブ上の共有フォルダを開くことができる。この画面を図4(d)に示す。その後画像をダウンロードすることができる。

図4(c)　閲覧への招待メール

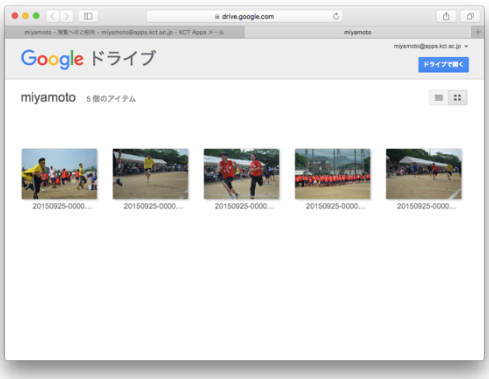

図4(d)　Google Driveの共有フォルダ

6. 結言

本研究では，ステガノグラフィを用いた画像配布システムの試作を行った。ユーザが画像選択用のWebサイトにログイン後画像を選択し，その画像に対しステガノグラフィを用いてユーザ情報を埋め込みGoogle Driveにアップロードを行う。その後Google Drive上の画像を共有することでダウンロードが可能となる。このシステムを利用することで不特定多数のユーザによって画像をダウンロードされることは無く，悪意あるユーザによって画像を拡散させられたとしても画像に埋め込まれたユーザ情報を抽出することでそのユーザを特定することができるようになった。結果，おおよそ目的通りに作動することを確認できた。

現状ではアプリケーションサーバは開発に使用しているPCのローカル上にインストールしているものを便宜的に利用している。今後は専用サーバを構築しその上で動作するよう本番環境を整え，本校の学生会の学生サービスの一環として体育祭などのイベントで撮影した画像を学生全員で共有し，自由にダウンロードできるシステムの構築を検討している。

参考文献

(1) Google Picasa
https://www.google.com/intl/ja/picasa/

(2) Microsoft OneDrive
https://onedrive.live.com/about/ja-jp/

(3) 脇山正博・田中義人・田中宏，情報セキュリティのステガノグラフィ，技術士 IPEJ Journal 2007年11月号，pp.8-11，2007

（2015年11月9日　受理）