マウストラッキングを用いたCAPTCHA方式の検討

立田 怜平[1]　山場 久昭[1]　久保田 真一郎[1]　朴 美娘[2]　岡崎 直宣[1]

概要：近年、無料メールサービスのようなアカウント登録をした上で利用するWebサイトに対して、ボットと呼ばれる自動プログラムを用いてアカウントを大量に取得したり、さらに取得したアカウントを使用した不正行為が頻発している。このような行為を防止するためにCAPTCHAと呼ばれる反転チューリングテストが広く利用されている。しかし、CAPTCHAを回避する手法として、インターネットの一般ユーザーや、低賃金労働者を利用してCAPTCHAを解読させるリレーアタックと呼ばれる攻撃がある。そこで、本研究では、リレーアタックを行った場合に生じる通信遅延に着目し、リレーアタックに耐性のあるマウストラッキングを用いたCAPTCHA方式を提案し、その有効性を実験により確認した。

A Study on the CAPTCHA Using Mouse tracking

Tatsuda Ryohei[1]　Yamaba Hisaaki[1]　Kubota Shinichiro[1]　Mirang Park[2]　Okazaki Naonobu[1]

Abstract: CAPTCHAs, which are reverse Turing tests, are used in many websites in order to guard them from bots attacks. However, there are many methods for breaking CAPTCHAs. Relay attack is one of such methods solving CAPTCHA using human solvers. We propose a CAPTCHA using mouse tracking to resist relay attack. We used delay time that is caused by communications needed in relay attack. We constructed an experimental enviroment that can simulate relay attack. A series of experiments was carried out to evaluate the performance of the proposed method.

1. はじめに

近年、無料メールサービスなどのWebサービスに対し、ボットと呼ばれる自動プログラムを用いてアカウントを大量取得したり、それらを用いてDos攻撃のような大量の不正をしたり、サービス要求を行うなどの不正行為が問題視されている。

このような問題を防止するために、CAPTCHA(Completely Automated Public Turing test to tell Computers and Humans Apart) と呼ばれる人間とボットを識別する方式が開発された [1]。CAPTCHAとはチャレンジ/レスポンス型テストの一種であり、人間には容易に解答できるがコンピュータには判別が困難である問題をユーザーに出題し、正解できたユーザーを人間と判断する技術である。

[1] 宮崎大学
Uniersity of Miyazaki
[2] 神奈川工科大学
Kanagawa Institute of Technology

しかし近年ではCAPTCHAを突破するリレーアタックと呼ばれる攻撃手法が登場している。既に、文字認識技術やパターン分類技術の発達によって文字列CAPTCHAや画像型CAPTCHAなどの既存のCAPTCHAは容易に突破されるようになってきており、その脆弱性が多くの研究者に指摘されている。特にリレーアタックは、インターネット上の一般ユーザーや低賃金労働者にCAPTCHAの問題を中継してそれを解読させる攻撃手法であり、人間がCAPTCHAの解読を行うため、プログラムを想定した対策では効果がなく、新たな対策が求められている。

そこで本論文ではリレーアタックを行った時に生じる、通信の中継による遅延時間に着目し、リレーアタックに耐性を持たせたCAPTCHA方式を提案する。提案方式は、動的なCAPTCHAであり、ランダムな位置に出現する複数の妨害オブジェクトの中から連続的に移動してその位置を変化させる移動オブジェクトを認識し、マウスカーソルで追跡できるか否かで人間か自動プログラムかを判別する。この動的な性質に加え、リレーアタックで生じるCAPTCHA

図 1　Google で利用されている CAPTCHA（文字列 CAPTCHA）
Fig. 1　CAPTCHA used by Google.

の問題を中継した際の遅延を利用してリレーアタックの防止を図る。以下、本稿では 2 章で既存の CAPTCHA とリレーアタック対策についての関連研究について述べる。3 章では、提案手法を説明し、4 章で実装の詳細と結果を示す。5 章では、提案手法のリレーアタックへの耐性の検証実験とユーザビリティ評価を行い、考察した後、6 章でまとめと今後の課題について述べる。

2. CAPTCHA とリレーアタック

CAPTCHA は、人間の高度な認識能力を利用して、画像や音声、文字列、動画を用いた、機械には難しい問題を出題し、その問題が解けるかどうかで人間と機械を区別している。

2.1 CAPTCHA の特徴

多くの Web サービス提供サイトで一般的に利用されている手法に、歪曲やノイズが付加された文字列画像を提示し、閲覧者がその文字列を正しく判読できるかどうか試す文字列型 CAPTCHA（図 1) や動物画像を複数提示して特定の動物の画像を選び出させる Assira と呼ばれる画像型 CAPTCHA などがある。

2.2 リレーアタック

リレーアタックは、攻撃者が正規サイトから CAPTCHA の問題画像を取得し、第三者の人間に CAPTCHA の問題画像を中継して解答してもらい、その解答を利用することで CAPTCHA を突破する手法である。問題画像の取得や第三者への問題の中継などは、攻撃者の作成したプログラムで自動的に行われる。

リレーアタックには、いくつか種類がある。例えば、攻撃者が運営するサイト（以下、リレーサイトと呼ぶ。）にインターネット上の一般ユーザーが訪問して来たら、正規サイトから取得してきた CAPTCHA の問題画像を提示し、リレーサイトのコンテンツを閲覧することと引き換えに解読させる手法や、攻撃者が低賃金労働者を雇って CAPTCHA の問題画像を労働者に転送し、報酬を与えて解読させる手法などがある。以降の説明では、リレーアタックに際して、CAPTCHA の解答を提供する（インターネット上の一般ユーザーや低賃金労働者) を幇助ユーザーと呼ぶことにする。

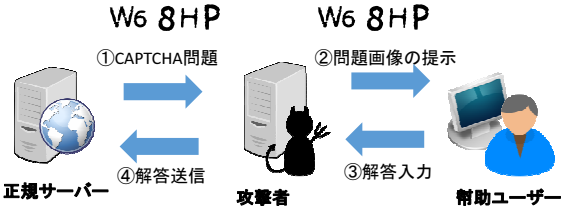

図 2　リレーアタックの一例
Fig. 2　example of relay-attack.

2.3 既存の CAPTCHA のリレーアタックへの耐性

この節では、既存の代表的な CAPTCHA の、リレーアタックへの耐性について述べる。

2.3.1 文字列 CAPTCHA

文字列 CAPTCHA は英数字の文字列を歪ませ、ノイズを付加して表示し、表示された文字列をテキストボックスに正しく入力できていたら解答者を人間と判断する。現在最も広く利用されている CAPTCHA である。

文字列 CAPTCHA の多くは静的な画像であるため、CAPTCHA の問題画像を取得すればリレーアタックは容易に成功できる。例えば、リレーサイトに幇助ユーザーアクセスして来たら、攻撃者の作成したプログラムが自動的に正規サイトにアクセスし、CAPTCHA の問題画像を取得する。そして、リレーサイトにアクセスしたユーザーに CAPTCHA の問題画像を提示して解いてもらい、その解答を利用して攻撃者のプログラムが CAPTCHA の解答入力を行うことで CAPTCHA を突破できる。

2.3.2 画像 CAPTCHA

画像 CAPTCHA は、文字列の画像を使用するものではなく、動物や食べ物など身の回りにある物の画像などを利用するものである。この画像 CAPTCHA の例として、Assira がある。Assira は、12 枚の犬と猫の画像を提示し、ユーザーが提示されている画像の中から猫の画像のみを選択できていれば、ユーザーを人間と判断する。

このような画像 CAPTCHA に対しても、リレーサイトを訪れた幇助ユーザーに対して問題画像を提示し、どの画像を選択したかを攻撃者の自動プログラムに通知することができれば、リレーアタックでの突破は可能である。

2.3.3 動画 CAPTCHA

動画ベースの CAPTCHA は、文字列 CAPTCHA や画像 CAPTCHA を動画へ応用したものである。動画

CAPTCHA の例としては、NuCAPTCHA が挙げられる。NuCAPTCHA は、カナダのソフトウェア企業 Leap Marketing Thechnologies が発表した CAPTCHA 手法である。NuCAPTCHA では、複数のフォントを用いたランダムな文字列が動画中に表示される。ユーザーは、動画上部に表示される色指定などを読み取り、動画中に流れる文字列中からそれに該当する文字列を解答として入力する。

NuCAPTCHA のような動画 CAPTCHA に対しては、画像キャプチャをくり返し行う。その結果、文字列が表示されている画像が取得できれば、文字列 CAPTCHA と同じようにリレーアタックで突破できる。したがって、リレーアタックへの耐性は、低いといえる。

2.4 リレーアタック対策

この節では、既存のリレーアタック対策やリレーアタックに耐性を持つ CAPTCHA について述べる。

2.4.1 IP アドレスの違いを利用した対策 [2]

リレーアタックでは、正規サイトにアクセスする PC とリレーサイトで中継された CAPTCHA を解く PC とが異なっている。この特徴を利用し、リレーアタックが行われていることを検知することができる。

しかしこの手法ではリレーアタックを検知するために必要な IP アドレスを正規サーバーに通知する機能を、一般ユーザーの PC にインストールするプログラムとして実現している。そのため、低賃金労働者を利用したリレーアタックのように、不正であることを知った上でリレーサイトにアクセスしてくるユーザーがいる場合には、本方式の機能のプログラムをインストールしないことで対策の回避が可能となってしまう。

2.4.2 DCG-CAPTCHA[3][4][5]

DCG-CAPTCHA は簡単なミニゲーム形式の CAPTCHA である。ユーザーが与えられた指示に適するオブジェクトを選択し、その選択が正しければ、人間とみなすものである。例えば図 3 では、複数の異なる形状のオブジェクトの中から、青いエリアのオブジェクトと同じ形状のものを選択し、青いエリアのその形状のオブジェクトの位置にドラッグ＆ドロップで配置できれば、ユーザーを人間とみなす。また、DCG-CAPTCHA のオブジェクトは常に移動しているので、ユーザーが解答を行う際のリアルタイム性に着目し、ユーザーと CAPTCHA との間のインタラクションのタイミングを検査することでリレーアタックの検出を実現している。

しかし、同形状のオブジェクトを認識することや移動するオブジェクトをフレーム画像を解析してプログラムで追跡することは容易にできるため、自動プログラムによる攻撃への耐性は低いと言える。

図 3 DCG-CAPTCHA の例
Fig. 3 example of DCG-CAPTCHA.

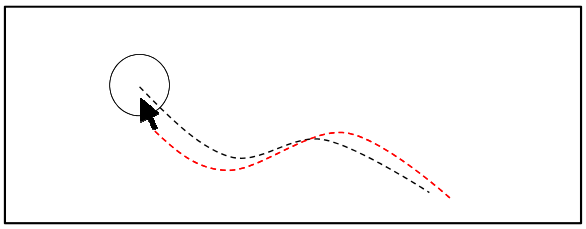

図 4 提案 CAPTCHA の例
Fig. 4 example of the proposed method.

3. 提案手法

3.1 目的と提案 CAPTCHA

本研究では、リレーアタックに対する耐性を持ちながら、プログラムによる自動化攻撃にも耐性を持つ CAPTCHA を作成することを目的とする。2.4 で述べたように、リレーアタックに対する対策はいくつか考えられているが、それらの手法は、自動プログラムへの耐性が弱くなっている。

提案する CAPTCHA は、動的な CAPTCHA 方式であり、図 4 に示すように移動する円形のオブジェクト（以下、移動オブジェクトとする。）をマウスカーソルで追跡できるか否かで解答者が人間か判断するものである。

次節にて、提案 CAPTCHA のリレーアタックとプログラムによる攻撃への対応について示す。

3.2 想定される攻撃に対する耐性

3.2.1 リレーアタックへの対応

図 5 に CAPTCHA に対してリレーアタックを行ったときの通信についてのシーケンス図を示す。

図 5 で用いている記号の意味を以下に示す。

Ox_t, Oy_t: 時間 t の移動オブジェクトの座標
$M^l x, M^l y$: 正規ユーザーのマウスカーソルの座標
$M^a x, M^a y$: 幇助ユーザーのマウスカーソルの座標
$\Delta t1$: CAPTCHA サーバーから中継 PC または正規ユーザーに移動オブジェクトの座標が送信されてくるまでの時間
$\Delta t2$: 中継 PC または、正規ユーザーから CAPTCHA サーバーにマウスカーソルの座標が送信されてくるまでの時間

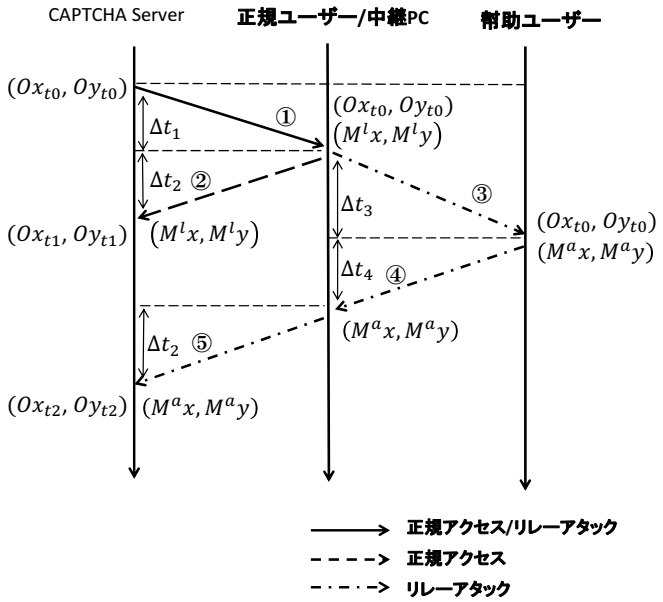

図 5　提案 CAPTCHA に対するリレーアタックのシーケンス図
Fig. 5　Sequence diagram of the relay attack.

$\Delta t3$**:** 中継 PC から幇助ユーザーに CAPTCHA のフレーム画像が送信されてくるまでの時間

$\Delta t4$**:** 幇助ユーザーから中継 PC にマウスカーソルの座標が送信されてくるまでの時間

正規アクセスと比較すると、リレーアタックでは、CAPTCHA のフレーム画像を幇助ユーザーに送信する処理と幇助ユーザーの解答を攻撃者の中継 PC に送信する処理が追加されている。この追加された通信によって、CAPTCHA サーバーに直接アクセスしている中継 PC で表示されているフレーム画像と幇助ユーザーの PC 上で表示されているフレーム画像には、時間のズレが生じる。このズレを生み出している遅延時間を利用してリレーアタックによる CAPTCHA の突破を防ごうというのが提案手法の基本的な考え方である。

まず、正規ユーザーが提案する CAPTCHA のサーバーにアクセスしているときの振る舞いを以下に示す。

(1) 時刻 $t0$ の移動オブジェクトの座標 (Ox_{t0}, Oy_{t0}) を正規ユーザーに送信する。
(2) 時刻 $t0 + \Delta t1$ で正規ユーザーには、(Ox_{t0}, Oy_{t0}) に移動オブジェクトがあるように見える。
(3) 正規ユーザーは、移動オブジェクト上にマウスカーソルを置く。このときのマウスカーソルの座標を $(M^l x, M^l y)$ とし、(Ox_{t0}, Oy_{t0}) と $(M^l x, M^l y)$ の位置は、非常に近い位置にあると仮定する。
(4) この座標 $(M^l x, M^l y)$ は、図 5 の ② の通信で CAPTCHA サーバーに送信される。
(5) マウスカーソルの座標 $(M^l x, M^l y)$ は CAPTCHA サーバーに時刻 $t1 = t0 + \Delta t1 + \Delta t2$ に到着する。この時、CAPTCHA サーバー上の移動オブジェクトの位置は (Ox_{t1}, Oy_{t1}) まで移動している。
(6) CAPTCHA サーバーでは、マウスの位置がおよそ $\Delta t1 + \Delta t2$ だけ前の移動オブジェクトの位置にあるように見える。

次に、提案する CAPTCHA にリレーアタックを行ったときの振る舞いを以下に示す。

(1) 時刻 $t0$ の移動オブジェクトの座標 (Ox_{t0}, Oy_{t0}) を中継 PC に送信する。
(2) 時刻 $t0 + \Delta t1$ で中継 PC では、(Ox_{t0}, Oy_{t0}) に移動オブジェクトが表示されている。
(3) 中継 PC は、(Ox_{t0}, Oy_{t0}) に移動オブジェクトが表示されているフレーム画像を取得し、幇助ユーザーに送信する。
(4) 時刻 $t0 + \Delta t3$ で幇助ユーザーには、(Ox_{t0}, Oy_{t0}) に移動オブジェクトがあるように見える。
(5) 幇助ユーザーは、移動オブジェクト上にマウスカーソルを置く。このときのマウスカーソルの座標を $(M^l x, M^l y)$ とし、(Ox_{t0}, Oy_{t0}) と $(M^a x, M^a y)$ の位置は、非常に近い位置にあると仮定する。
(6) この座標 $(M^a x, M^a y)$ は、④ で中継 PC に送信される。
(7) 中継 PC は、受信した座標 $(M^a x, M^a y)$ にマウスカーソルを移動させる。
(8) この座標 $(M^a x, M^a y)$ は、⑤ の通信で CAPTCHA サーバーに送信される。
(9) マウスカーソルの座標 $(M^a x, M^a y)$ は CAPTCHA サーバーに時刻 $t2 = t0 + \Delta t1 + \Delta t3 + \Delta t4 + \Delta t2$ に到着する。この時、CAPTCHA サーバー上の移動オブジェクトの位置は (Ox_{t2}, Oy_{t2}) まで移動している。
(10) CAPTCHA サーバーでは、マウスの位置がおよそ $\Delta t1 + \Delta t3 + \Delta t4 + \Delta t2$ だけ前の移動オブジェクトの位置にあるように見える。

以上より、正規ユーザーがアクセスした時のマウスの座標のずれより幇助ユーザーがアクセスした時のずれが $\Delta t3 + \Delta t4$ の分だけ大きくなっていることを利用して、判定を行う。

3.2.2　物体追跡技術への対処

提案手法の CAPTCHA では、移動する円形オブジェクトをマウスカーソルで追跡する解答方法をとっているため、物体追跡技術を用いて、移動オブジェクトをプログラムで自動的に追跡する攻撃が考えられる。

そこで提案手法では、移動オブジェクトと同じ形、大きさ、色の妨害オブジェクトを用いて、プログラムによる追跡が困難になるように設計した。

図 6 に示すように、提案する CAPTCHA は、移動オブジェクトのフレーム画像と妨害オブジェクトのフレーム画像を重ねて表示する。また、移動オブジェクトが 1 フレームごとに位置を更新するのと同時に複数の妨害オブジェクトがランダムに位置を更新する。

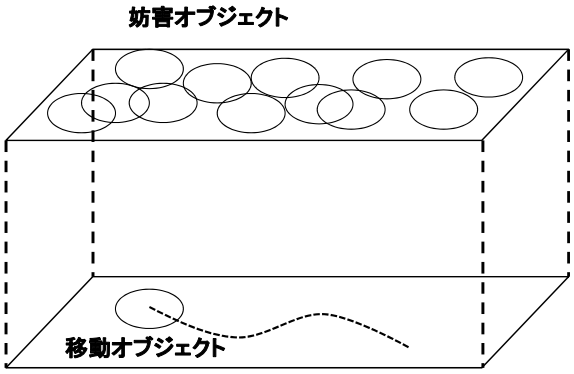

図 6　妨害オブジェクトの追加
Fig. 6　Additional interference object.

移動するオブジェクトを追跡する方法としては、提案 CAPTCHA のフレーム画像から背景画像を差し引くことで、前景の移動オブジェクトを抽出して追跡する方法や、追跡対象のカラーヒストグラムの特徴を元に追跡する方法などがある。この追跡を成功させるには、追跡対象を画像処理によって検出できること、または、追跡対象のカラーヒストグラムなどの色や形状などの特徴点をデータとして所持しておく必要がある。ところが、攻撃者が移動オブジェクトを自動的に追跡しようとフレーム画像を解析しようとしても、各フレーム画像は、同じ形状、色のオブジェクトがランダムに配置されているようにしか見えない。そのため、移動オブジェクトのみを検出することや、カラーヒストグラムなどの特徴点を利用した追跡は、困難になると考えられる。

3.3　認証手順

提案する CAPTCHA の認証手順を図 7 に示す。CAPTCHA が開始されて、移動オブジェクト上にマウスカーソルを乗せたら追跡開始とし、この解答時間の 10 秒間の間、移動オブジェクト上にマウスカーソルが乗っていた時間（移動オブジェクトの中心座標とマウスカーソルの座標との距離が移動オブジェクトの半径以下であった時間。以下、捕獲時間とする。）が設定した閾値よりも長い時間であれば、ユーザーを人間と判断する。追跡してもらう解答時間は、現在最も広く利用されている文字列 CAPTCHA の解読にかかる平均所要時間は 10 秒程度あるため、追跡開始から 10 秒間とした [6]。

4.　実装

4.1　基本設計

提案 CAPTCHA は、HTML5 の canvas 要素として描画する方法を取ることにした。ただしこの時、サーバーからクライアントに CAPTCHA のソースコードを送信し、それをクライアント上で実行する方式では、攻撃者にソースコードを解析される恐れがあり、セキュリティ上問題がある。従って、サーバーとクライアント間で CAPTCHA の描画に必要な情報（移動オブジェクトと妨害オブジェクトの座標）やユーザーの解答情報（マウスカーソルの座標）をリアルタイムで送受信することが可能な環境を構築することとした。

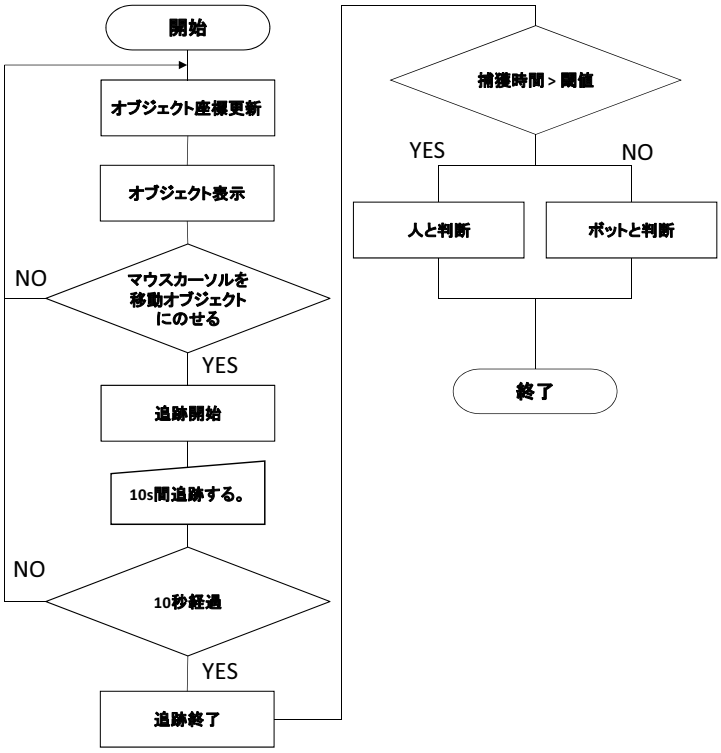

図 7　提案手法のフローチャート
Fig. 7　Flowchart of the proposed method.

4.2　開発環境

開発言語は JavaScript、CAPTCHA のサーバーを Node.js を用いて、CentOS6.6 上で実装した。

4.3　CAPTCHA システムの実装

4.1 節に示した実装方針に基づき、提案方式の CAPTCHA システムを実装した。サーバー側では、移動オブジェクトと妨害オブジェクトの位置座標の生成、各オブジェクトの位置座標をクライアントに送信する機能を実装した。クライアント側では、両オブジェクトの座標を受信するたびに移動オブジェクトと妨害オブジェクトを再描画する機能と、解答時間である 10 秒の間、0.1 秒ごとにマウスカーソルの座標とクライアント上の移動オブジェクトの座標を送信する機能を実装した。提案する CAPTCHA の実装に用いた各パラメータの詳細を以下に示す。

更新頻度:　サーバー上で、0.01 秒ごとに移動オブジェクト、妨害オブジェクトの位置座標を更新してクライアントに送信する。クライアント側では、移動オブジェ

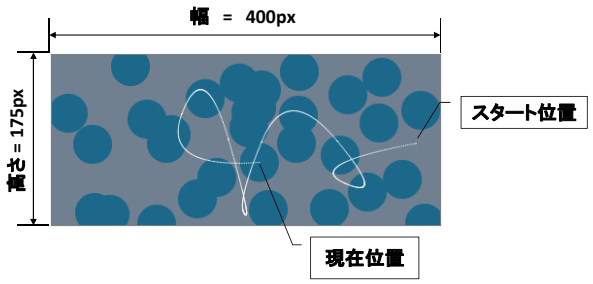

図 8　実装した CAPTCHA の 1 フレーム
Fig. 8　1 frame of the CAPTCHA.

クトと妨害オブジェクトの位置座標は、1 フレームごとに同時に更新する。

移動オブジェクト:　移動オブジェクトは、半径 20px の円である。

妨害オブジェクト:　妨害オブジェクトは、移動オブジェクトと同じ大きさ、色で半径 20px の円とする。これは、移動オブジェクトと異なる形状や色で表示すると、画像処理で移動オブジェクトを検出されてしまうのからである。

捕獲時間:　移動オブジェクトは半径 20px の円に設定したので、マウスカーソルの座標と移動オブジェクトの中心座標との距離が 20px 以下であったときの時間の総和を捕獲時間と呼ぶ。

ウィンドウサイズ:　高さ 175px、幅 400px

図 8 は、作成した CAPTCHA の 1 フレームを抜き取ったものである。スタート位置は、CAPTCHA が開始されたときの移動オブジェクトの位置であり、現在位置は、抜き取ったフレームでの移動オブジェクトの位置である。白線は、抜き取ったフレームまでに移動オブジェクトが移動した軌跡を表したものであり、移動オブジェクトは白線で示されている不規則な動きをする。

5.　実験と考察

今回は、提案 CAPTCHA のリレーアタックへの耐性の検証と提案 CAPTCHA のユーザビリティ評価を行い、CAPTCHA としての実用性について調査する。

5.1　リレーアタック耐性の検証実験

5.1.1　実験目的

実装した提案 CAPTCHA に対して実際にリレーアタックを行い、リレーアタックへの耐性を与えられるか確認する。

5.1.2　実験方法

実験は、宮崎大学工学部生の被験者 8 名に、正規アクセスで提示された CAPTCHA とリレーアタックで提示された CAPTCHA をそれぞれ 5 回ずつ解いてもらい、移動オブジェクトの捕獲時間の計測を行った。

表 1　実験結果
Table 1　experimental results.

	最長捕獲時間	最短捕獲時間	平均捕獲時間
正規アクセス	8.8s	4.1s	6.5s
リレーアタック	2.3s	0.1s	0.6s

今回は、リレーアタックを再現するためのソフトウェアとして VNC(Virtual Network Computing) を使用した。VNC はネットワークを通じて接続された他のコンピューターの画面を遠隔操作するソフトウェアである。

正規アクセスでは、被験者に CAPTCHA サーバーにアクセスしてもらい表示された提案 CAPTCHA を解いてもらった。リレーアタックでは、中継 PC で VNC サーバーを起動しておき、CAPTCHA サーバーにアクセスして提案 CAPTCHA を表示する。次に、学校 PC の web ブラウザから中継 PC の VNC サーバーに接続し、中継 PC の CAPTCHA が表示された画面が学校 PC に表示されるので、被験者に学校 PC に中継された CAPTCHA を解いてもらった。正規アクセスのときに、CAPTCHA サーバーにアクセスする PC とリレーアタックのときに VNC サーバーを起動して、CAPTCHA サーバーにアクセスする中継 PC は、同じものを利用し、この中継 PC のインターネット接続は、ポケット WiFi を利用した。

5.2　実験環境

本実験の環境は、以下のとおりである。

CAPTCHA サーバー:　さくら VPS

中継 PC:　Dynabook R731/E26ER(メモリ 4.0GB, OS windows7 Home Premium)

VNC:　ThinVNC

ポケット WiFi:　Pocket WiFi GP02（受信最大 21Mbps / 送信最大 5.7Mbps）

5.2.1　実験結果

正規アクセスとリレーアタックのそれぞれの環境で被験者 8 人に 5 回ずつ提案 CAPTCHA を解いてもらったときの 40 個のデータから得られた、移動オブジェクトの最大捕獲時間、最小捕獲時間、平均捕獲時間について表 1 に示す。

この捕獲時間は、移動オブジェクトにマウスカーソルが乗ってからの 10 秒間の解答時間の間でマウスカーソルが移動オブジェクトに乗っていた（移動オブジェクトの中心座標とマウスカーソルの座標の距離が 20px 以下）時間を測定したものである。

表 1 より、リレーアタックの最長捕獲時間が 2.3 秒であり、正規アクセスでの最短捕獲時間である 4.1 秒にも達していなかった。このことから、今回の提案 CAPTCHA の捕獲時間の閾値を 4 秒以上に設定（4 秒以上の捕獲時間であれば、CAPTCHA を成功とする。）すると、実験環境おいて、提案 CAPTCHA をリレーアタックで成功させるこ

表 2　成功率と所要時間
Table 2　success rate and required time.

	成功率	平均所要時間 [秒]
ユーザー 1	3/3	11.3
ユーザー 2	3/3	12.0
ユーザー 3	3/3	14.8
ユーザー 4	3/3	13.3
ユーザー 5	3/3	11.7
ユーザー 6	3/3	13.4
ユーザー 7	3/3	12.6
ユーザー 8	3/3	13.0
ユーザー 9	2/3	12.7
ユーザー 10	3/3	15.9
平均	96.6%	13.0

表 3　アンケート結果
Table 3　a questionnaire result.

	(1)	(2)	(3)	(4)	(5)
ユーザー 1	5	5	5	4	提案 CAPTCHA
ユーザー 2	5	5	5	5	提案 CAPTCHA
ユーザー 3	5	5	5	5	提案 CAPTCHA
ユーザー 4	4	4	5	4	提案 CAPTCHA
ユーザー 5	5	5	4	4	提案 CAPTCHA
ユーザー 6	4	4	4	3	提案 CAPTCHA
ユーザー 7	3	4	4	4	提案 CAPTCHA
ユーザー 8	5	5	5	5	提案 CAPTCHA
ユーザー 9	5	5	5	5	提案 CAPTCHA
ユーザー 10	5	5	5	5	提案 CAPTCHA
平均	4.6	4.7	4.7	4.4	

とはできないと考えられる。

しかし、リレーアタックの遅延時間は通信環境に依存するため、今後は、適切な閾値を効率良く発見できる手法を検討する必要がある。

5.3　ユーザビリティ評価

ユーザビリティ評価で、提案 CAPTCHA の正答率や所要時間の測定と被験者に対するアンケート調査を行い、提案 CAPTCHA の実用性を確認することを目的とする。ユーザビリティ評価は、宮崎大学工学部生 10 名を対象に行った。被験者には、提示された提案 CAPTCHA の移動オブジェクト上にマウスカーソルが乗り続けるように追跡することを指示した。捕獲時間の閾値は、リレーアタック耐性の評価実験での正規アクセスで得られた最小捕獲時間が 4.1 秒であったため、捕獲時間が 4 秒以上であれば追跡できているとみなし成功とする。各被験者には練習を行った後、提案 CAPTCHA を 3 回ずつ解いてもらい、各解答の成否と解答にかかった所要時間を記録した。また、被験者に 1 点〜5 点の評価でアンケート回答してもらった。アンケートの質問項目を以下に示す。

(1) 簡単に解けたか（簡単であれば 5 点）
(2) 面倒だと感じたか（面倒でないなら 5 点）
(3) CAPTCHA は、使いやすかったか（使いやすいなら 5 点）
(4) web サービス上で使いたいか (使いたいなら 5 点）
(5) 実際の web サービスの場面で CAPTCHA を解くことが要求されたときに、文字列 CAPTCHA と提案 CAPTCHA のいずれかを選ぶことができた場合どちらを選ぶか。

ユーザーごとの提案 CAPTCHA の成功率と平均所要時間をまとめた結果を表 2 に、アンケート結果について表 3 に示す。全ユーザーの平均正答率は、96.6% であり、平均所要時間は、13.0 秒である。一般的な文字列型 CAPTCHA の平均所要時間は 10 秒程度であるため、提案 CAPTCHA は文字列型 CAPTCHA と同程度の時間で解ける CAPTCHA であると考えられる。また、アンケート結果から、全体的に 1 点、2 点の評価をした回答がないため、提案方式に大きな負担を感じた被験者はいないと考えられる。質問 (6) では、すべての被験者が文字列 CAPTCHA よりも提案 CAPTCHA を選択していた。これらのことから、提案 CAPTCHA は実用的であるといえる。

6.　まとめと今後の課題

本研究では、リレーアタックを行った時に生じる、通信の中継による遅延時間に着目し、リレーアタックに耐性を持たせた CAPTCHA を提案した。また、提案方式の CAPTCHA を実装し、リレーアタックを再現する実験環境を構築して、リレーアタックへの耐性の検証実験を行った。実験の結果、提案方式が実験環境でのリレーアタックに対して耐性を持つことが可能であることを示した。また、ユーザビリティ調査を行い、提案 CAPTCHA の実用性を確認した。

今後は、今回確認することができなかった、自動プログラムによる攻撃への耐性について検証実験を行い、提案方式の自動プログラムへの耐性について引き続き検討していきたい。

参考文献

[1] L. von Ahn, M. Blum, N. Hopper, and J. Langford, “CAPTCHA: Telling humans and computers apart,” Advances in Cryptology, Eurocrypt’03, vol.2656 of Lect. Notes Comput. Sci.,pp.294-311, 2003.
[2] 鈴木徳一郎, 山本匠, 西垣正勝. (2010). リレーアタックに耐性をもつ CAPTCHA の提案. 情報処理学会研究報告. CSEC,[コンピュータセキュリティ], 2010(21), 1-8.
[3] Mohamed, Manar, et al. ”A three-way investigation of a game-CAPTCHA: automated attacks, relay attacks and usability.” Proceedings of the 9th ACM symposium on Information, computer and communications security. ACM, 2014.
[4] Mohamed, Manar, et al. ”Dynamic cognitive game

captcha usability and detection of streaming-based farming.” the Workshop on Usable Security (USEC), co-located with NDSS. 2014.

[5] Gao, Song, et al. ”Gaming the game: Defeating a game captcha with efficient and robust hybrid attacks.” Multimedia and Expo (ICME), 2014 IEEE International Conference on. IEEE, 2014.

[6] 可児潤也, 鈴木徳一郎, 上原章敬, 山本匠, 西垣正勝. (2013). 4 コマ漫画 CAPTCHA. 情報処理学会論文誌, 54(9), 2232-2243.

[7] 藤田, 真浩, 池谷, 勇樹, 米山, 可児, ... 西垣正勝. (2014). SNOW NOISE CAPTCHA: 無意味な情報を利用した動画 CAPTCHA の提案. 研究報告コンピュータセキュリティ (CSEC), 2014(29), 1-7.