Androidアプリの自動リンクにおける悪意のあるリンク生成リスクの検討

辻 知希[1,a)]　嶋田 創[2]　山口 由紀子[2]　長谷川 皓一[3]

概要：スマートフォンなど様々なデバイスに対応したサービスが普及した現代であっても，デバイス固有の仕組みは攻撃者の標的となりやすい．本研究では特に Android 向けのアプリケーションにおいて，URL 自動リンク機能での特殊文字の扱いにより URL が分割され，意図されたものと異なるリンクが埋め込まれる挙動に注目し，これによって悪意のあるサイトなどへユーザが誘導されるリスクについて検討する．また，Android 向けのアプリケーションと Web ブラウザ向けのアプリケーションの URL 自動リンク機能の差異により，信用された良性 URL が分割され自動リンクされることにより悪意のあるリンクが生成される仕組みを調査し，実際のアプリケーションにおいての発生状況を確認した．結果として，5 億インストールを超える複数の Android アプリケーションが本研究で示すリスクを有することが分かった．最後にそれら Android アプリケーションでのリスクを軽減する URL 自動リンクの実装について示す．

キーワード：悪性 URL，自動リンク処理，悪性 Web サイト，フィッシング，Android

Security Risk of Malicious Link Generation in Auto-Link Feature of Android Applications

Satoki Tsuji[1,a)]　Hajime Shimada[2]　Yukiko Yamaguchi[2]　Hirokazu Hasegawa[3]

Abstract: Services that support various devices such as smartphones are popular in recent days. However, device specific mechanisms are still easy targets for attackers. In this paper, we focus on behaviors of URL splitting in Android applications, especially when special characters are handled in Auto-Link feature. The URL splitting generates and embeds links that are different from intended ones. Thus, we discuss risks of users being directed to malicious sites. We surveyed differences in the Auto-Link feature among Android/Web applications to confirm whether malicious links are generated by splitting trusted benign URL or not. We confirmed that the malicious URL are generated by the splitting and Auto-Link feature generates links to them in real applications. We also confirmed that several Android applications with more than 500 million installs have above risks shown in this study. Finally, we present an implementation of the Auto-Link feature that mitigates above risks in Android applications.

Keywords: Malicious URL, Automatic link, Malicious Website, Phishing, Android

1. はじめに

今や私たちは個人で WWW を利用し，SNS や動画配信サービスにアクセスしない日はない．それぞれのサービスはハイパーリンクによって相互に連携しあうことで，外部コンテンツをインラインで埋め込んで参照したり，ユーザが自由に移動し情報を収集/発信することを可能としてい

[1] 名古屋大学 大学院 情報学研究科
Graduate School of Informatics, Nagoya University
[2] 名古屋大学 情報基盤センター
Information Technology Center, Nagoya University
[3] 名古屋大学 情報連携推進本部 情報セキュリティ室
Information Security Office, Information and Communications, Nagoya University
[a)] satoki@net.itc.nagoya-u.ac.jp

表 1 example.com へのホモグラフ攻撃の例
Table 1 Examples of homograph attacks on example.com

ドメイン名	対象文字	Unicode
example.com	l	U+006C
examp1e.com	1	U+0031
exampIe.com	I	U+0049
exampĺe.com	Í	U+013A
examp̨le.com	l̨	U+013C

る．また，サービスの提供形態も変化し，従来のパーソナルコンピュータに向けたアプリケーションや Web ページのみでなく，スマートフォン向けユーザインタフェースなどユーザの利用する環境に合わせてその見た目や機能を変えるよう設計されている．特に，近年では Web アプリやスマートフォン向けアプリの利用が広がっており，Web 版や Android アプリ版といった提供形態が広く利用されている．本研究ではこの提供形態のうち，Android アプリに対する自動リンクと Web 版のリンクの差異によって，ユーザが信用できると判断する URL から悪意のあるサイトに誘導されてしまう新たなリスクについて扱う．

2. 先行研究

URL を対象とした攻撃手法として，人間の認識の穴をついた攻撃やアプリケーションの文字解釈の違いに関する攻撃が研究されている．人間の認識の穴をついた攻撃はホモグラフ攻撃などが知られており，ドメインの一部を誤認しやすい文字に変え，ユーザを悪意のあるサイトへ誘導する攻撃である．**表 1** に示す例では最上段の example.com が本来のドメインであり，下段はすべてホモグラフ攻撃での利用が想定されるドメインとなっている．特に国際化ドメイン名（IDN）によるホモグラフ攻撃は利用できる文字数が多く，ユーザがドメインを誤認しやすくなるとされる．鈴木らは各ドメインの文字列に対する編集距離を計算することで，IDN ホモグラフ攻撃を検出している [1].

文字の解釈違いに関する攻撃として，白倉らは URL の抽出や自動リンク機能（Auto-Link feature）に注目し，アプリケーション依存の処理による既存のセキュリティ機構のバイパスや，ドメイン名の誤った解釈によるセキュリティリスクについて研究を行った [2][3]．ここでいう自動リンク機能とは，プレーンテキストから URL などであると判断できる部分をリンクに変換する機能を指す．結果として，ドメイン名に半角英数字以外を利用した場合，PhishTank[4] にて有害と登録された URL に対するスパムフィルタリングツールの無効化などが確認された．さらに Web メールサービスなどでは不正な HTML が生成され表示が崩れることが示された．これは URL 中の絵文字をイメージタグでの画像へと書き換えることで発生し，CSS インジェクションやクロスサイトスクリプティングなどの脆弱性を生じる要因となりえる．

IDN では，マルチバイト非対応の処理系でも IDN を扱えるように，Unicode 文字形式で表示される U ラベルとそれに 1 対 1 で対応する ASCII 文字形式で表示される A ラベルが用いられる．この U ラベルから A ラベルへと変換する処理において，正規化と Puny coding を順に行う．Birch は正規化においてホスト名が分割され，本来想定されるものと異なるドメインへ機密情報が送信される脆弱性 HostSplit[5] を報告している．

表 2 Linkify によるリンク対象テキスト
Table 2 Target texts linked by Linkify

パターン	リンクされる対象
ALL	すべてのリンク可能であるテキスト
EMAIL_ADDRESSES	メールアドレスと判断されるテキスト
PHONE_NUMBERS	電話番号と判断されるテキスト
WEB_URLS	URL と判断されるテキスト

表 3 パーセントエンコーディングが必要な文字の例
Table 3 Examples of characters that require percent encoding

文字	Unicode	パーセントエンコーディング
|	U+007C	%7C
`	U+0060	%60
^	U+005E	%5E

3. Android アプリで用いられる自動リンク生成開発キットの挙動

本研究では主に Android アプリにおける自動リンクについて扱う．Android アプリでは，アプリ中のテキストを表示する要素である TextView において，テキストを解析し自動でリンクに変換する Linkify[6] と呼ばれる開発キットが使用される．本開発キットでは，リンクに変換するテキスト中のパターンとして**表 2** に示すとおり ALL, EMAIL_ADDRESSES, PHONE_NUMBERS, WEB_URLS などの指定や，自身で設定した正規表現を利用することができる．本研究では，ALL と WEB_URLS を扱い，テキスト中のすべてのリンク可能であるテキスト，および，URL と判断されるテキストをリンクに変換する 2 つの設定に関してのみ扱う．これはメールアドレスや電話番号についても本研究にて示すリスクは生じ得るが，フィッシングなどの攻撃へ発展するリスクが少なく，影響範囲が限定的であると考えられるためである．

URL に利用できる文字種は RFC3986[7] に定められている通り，予約文字 (Reserved Characters) および非予約文字 (Unreserved Characters) からなり，その他の文字はパーセントエンコーディングの必要性がある．本研究ではこのパーセントエンコーディングが必要な文字の例として，**表 3** に示す文字に注目する．本研究では，表 3 に示す文字を含む**図 1** の URL に対する，Android アプリの自動

```
https://www.google.com/search?q=satoki|example.com
https://www.google.com/search?q=satoki`example.com
https://www.google.com/search?q=satoki^example.com
```

図 1　パーセントエンコーディングが必要な文字を含む URL

Fig. 1　URLs that contain characters that require percent encoding

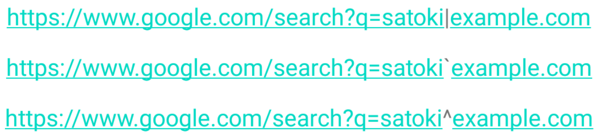

図 2　Android アプリの自動リンクにおける挙動

Fig. 2　Behavior of Auto-Link feature in Android applications

```
https://www.google.com/search?q=satoki`evil.example.com?
dummy=aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaa
```

図 3　セキュリティリスクとなりうる URL の例

Fig. 3　Example of URL that may pose a security risk

リンクにおける挙動について解析する．解析方法として，Android Studio 4.2.2[8] により Linkify を用いたアプリを作成し，対象の URL をテキストエリアに表示し，どのような自動リンク処理がなされるかを確認する．本実験では，リンクとそれ以外のテキストを区別するため，リンクの文字にのみに色がつくよう設定を行った． 結果は**図 2** に示す通りとなり，URL のパーセントエンコーディングが必要な文字で URL が切断され，2 つのリンクと解釈されたことがわかる．先頭部分をタップした場合は www.google.com へ，後半部分をタップした場合には example.com へのアクセスが行われる．このような URL の切断現象が発生する印字可能な ASCII 文字を調査した結果，予約文字以外では，表 3 に例として挙げたもののほかに複数種類確認された．

4.　自動リンクによる URL の悪性化のリスク

アプリ開発者自身が設定した URL がパーセントエンコーディングが必要な文字によって分割される場合では，意図された動作と異なるバグとなるが，セキュリティ上の問題は発生しにくい．一方，ユーザの入力に対して自動リンク処理を行う場合には，URL が分割されることによって，フィッシングサイトなど悪意のあるサイト（以降，悪性サイトと呼ぶ）へ誘導されるリリンクが生成されるスクが高くなる．例として，**図 3** に示す URL を使用する．この URL は，URL の前半部分が www.google.com，後半部分が evil.example.com で構成されており，パーセントエンコーディングが必要な文字によって連結されている．前半部分が良性のサイトであり後半部分が悪性サイトであるとし，後半部はテキスト量を多くするために dummy として

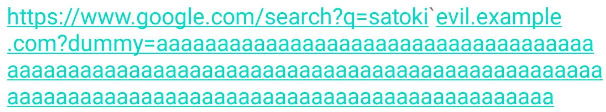

図 4　リンクの文字色設定あり

Fig. 4　Link text color enabled

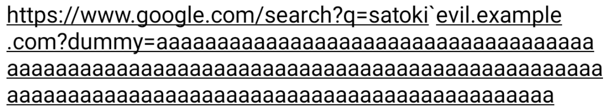

図 5　リンクの文字色設定なし

Fig. 5　Link text color disabled

長いクエリパラメータが設定されている．**図 4**，**図 5** に示す通り，リンクの文字色設定のある図 4 では，注意して見れば URL の分割が発生していることが分かるのに対し，図 5 ではリンクの文字色設定を行わないために，色によって悪性サイトを判断することが難しくなっている．さらに，後半部分に追加されたクエリパラメータによりタップされる面積が拡大されていることで，悪性サイトに誘導される確率が高くなっている．特に Android アプリでは，マウスオーバーなどによって遷移先が自動表示される機能が無いため，確認を行うことなくリンクをタップする場合が多いと予測される．

さらに URL の分割によるリスクが高くなる事例として，Android アプリ版と共に Web 版の提供を行っている場合に，Android アプリ版と Web 版の挙動の差異を原因としたユーザ誤認による，悪性サイトへのアクセスが考えられる．Web 版でも Android アプリ版と同様に URL が自動リンクの対象となっていることがほとんどであるが，この自動リンクは「元のテキストに HTML のタグの 1 つであり，リンクを表す a タグの，リンク先を指する href 属性に URL を指定することでのリンク（以降 a タグでのリンクと呼ぶ）を挿入する」処理が行われることが多い． Web 版を Android のブラウザなどで閲覧する場合，WebView を用いたページをアプリ内に組み込んだ場合，その他の環境で Web 版を使用する場合には a タグでの自動リンクが用いられることとなる．この自動リンクが Linkfy とは異なる挙動のライブラリを用いて実現された場合，URL はパーセントエンコーディングが必要な文字によって分割されず，URL すべてが www.google.com へのアクセスとなる．つまり，**図 6** に示すように Web 版と Android アプリ版によってユーザからの同一の入力を別のリンクとして解釈してしまう問題により，Web 版では良性リンクとなっているものが Android アプリ版では悪性化してしまうという挙動の差が発生する恐れがある．これにより，過去に Web 版で良性 URL を受け取りアクセスしたことのあるユーザが，同一の URL を Android アプリ版で受け取った場合，

図 6　自動リンクによる URL の悪性化
Fig. 6　Malicious link generation in Auto-Link feature

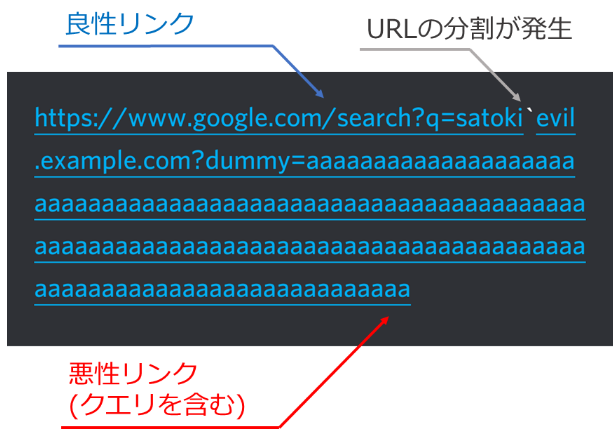

図 7　Android アプリ A のポップアップ表示
Fig. 7　Pop-up of Android application A

URL の分割によって後半部分が悪性サイトへのリンクとなる場合があるが，以前アクセスを行なった URL ということで信用してしまい，後半部分が悪性サイトへのリンクとなっていることを認識することがより難しくなると考えられる．

開発者の視点から見ると，複数の環境による挙動の違いを認識/確認することは難しく，一方の環境では良性リンクであるが他方では悪性リンクとなる本挙動を確認するためには，テストにおいて，URL が自動リンクされうるすべての個所をテストする必要がある．1 つの環境でのみテストを行いリンクの挙動を問題ないと判断してしまうケースも少なくないと考えられる．

5.　実際の Android アプリにおける調査

前節までにおいて，パーセントエンコーディングが必要な文字によって URL が分割されることによるセキュリティリスクについて述べた．本節では，実際に利用されているサービスについてのセキュリティリスクを確認する．対象とするサービスは，2021 年 8 月 19 日時点での Google Play で配布されている人気（無料）アプリランキング 100 位までの中から 5 億インストールを超えるチャットアプリを選定し，その中から，Web など他環境版も提供されている 3 アプリ（以降アプリ A，アプリ B，アプリ C と呼ぶ）を選択した．チャットアプリを選択した理由は，URL の投稿とその自動リンク結果の確認が容易であり，実際のフィッシング事例 [9] などが確認されているためである．前節で示した図 3 のセキュリティリスクとなりうる URL を例として用い，引き続き www.google.com を良性サイト，evil.example.com を悪性サイトであると仮定し，Android アプリ版と Web 版での自動リンクの挙動によって悪性サイトへの誘導が発生するか否かを確認する．また，これら自動リンクされた URL は，アプリ内でタップすることにより「アプリ自体で開く/外部ブラウザで開く」といった利用方法の他に，リンクを長押しすることで Android 標準搭載のポップアップを表示させ，URL をコピーし外部に張り付けるといった利用法が想定される．それらポップアップを独自で実装しているアプリも多くあるため，ポップアップにおけるリンクに対しても悪性サイトへの誘導の有無を確認する．

5.1　URL の分割が発生するが目視で確認できるアプリ

図 7 ではアプリ A に対して図 3 の URL を送信した後，長押しによりポップアップを表示した結果を示す．Web 版，Windows 版，Android アプリ版全てで a タグでのリンクを用いているようであり，通常のチャットエリアでは悪性リンクによるセキュリティリスクは顕在化しなかった．しかし，Android アプリ版において，本アプリ固有の実装として，リンクを長押しした場合に独自のポップアップが表示される仕組みとなっており，ポップアップの表示では本研究で問題とする URL の分割が確認された．また，リンクをコピーする機能では分割されていない URL がクリップボードへコピーされるため，外部の URL チェックサービスなどへの貼り付けた際には良性リンクと判断される．これらの観測結果から，チャット画面や Android 版以外で信用された URL は，ポップアップ表示されることで悪性化する危険性があると考えられる．さらに本アプリでは，表 3 で示したパーセントエンコーディングが必要な文字以外の日本語などのマルチバイト文字でも同様の現象が発生することが確認された．これは，Linkify のリンクに変換するテキスト中のパターンとして，表 2 に示したもの以外に独自のマルチバイト文字を検出するパターンを導入したことによる実装の不備と考えられる．幸いなことに，本アプリでは文字色と下線を付けてリンク文字列を表示する実装であるため，注意して確認することで URL が分割されていることを認識できる．

5.2　URL の分割が発生し目視で確認できないアプリ

図 8 ではアプリ B に対して図 3 の URL を送信した結

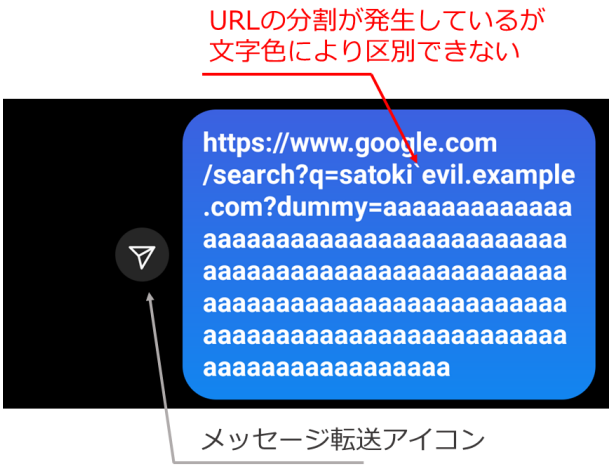

図 8 Android アプリ B の表示
Fig. 8 Android application B

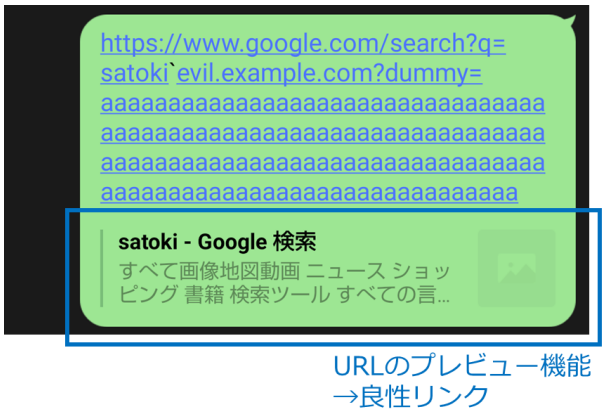

図 9 Android アプリ C での URL のプレビュー
Fig. 9 URL Preview in Android Application C

果を示す．アプリ B においても Web 版，Windows 版では a タグでのリンクとなっていたが，Android アプリ版では通常のチャットエリアにおいて Linkify を用いているようであり，本研究で示すセキュリティリスクが確認できた．アプリ内では，メッセージ横のアイコンをタップすることで，メッセージを同アプリの他のユーザに転送する機能が実装されているが，転送した先でも分割が発生していることを確認した．これにより悪性化した URL を意図せず伝播させてしまい，被害が広範囲にわたる危険性が高くなるとも考えられる．また長押しすることによるクリップボードへのコピーにおいては，アプリ A と同様に，URL は分割される前のものであった．アプリ B でのコピー機能は URL だけでなくメッセージ全体がコピーされるため，リンク以外の部分に注目してしまい URL が分割されていることに気づきにくいことに加え，コピーした時点の形では URL チェックサービスへの入力が行えないため，チェックを行わず他の場所へ張り付ける危険性が高くなる．さらにリンクの文字色の変化や下線の付与などの視覚上の変化が無く，ユーザが URL の分割による悪性リンクの生成を認識することが著しく困難となっている例である．なお，本アプリではマルチバイト文字での分割は発生しなかった．

5.3 URL の分割が発生し目視で確認できるがプレビュー機能で良性のリンク先のみ表示されるアプリ

図 9 ではアプリ C に対して図 3 の URL を送信した結果を示す．アプリ C においては，Android アプリ版および Web 版の双方にて本研究で示すセキュリティリスクが確認できたが，他のサービスと異なり，チャットエリアにリンクを先読みした結果の OGP 情報が表示されることが見て取れる．これは URL のプレビューと呼ばれる機能であり，URL の分割が発生した際にテキストに含まれている複数のリンク先の情報をユーザへ表示することができるためリンク先を確認する手段の 1 つとなりえる．しかし，本アプリ内では情報を取得できなかったリンク先については無視され，取得できたプレビューのみを表示しているため，かえってユーザの URL 全体が良性リンクという誤認を誘発する表示となってしまっている．URL 分割を悪用する攻撃者がこの挙動を認識し，アプリ C からの OGP 情報の取得リクエストのみをブロックするような悪性サイトを作成した場合，プレビューを表示が表示されず，URL 全体が良性サイトへのリンクに見えてしまう．また，本アプリにおいては，メモ機能など URL のプレビューが有効でない箇所でのセキュリティリスクも確認された．

6. リスクを軽減する URL 自動リンクの実装

本節では，悪性リンク生成のリスクを軽減するための対策や URL 自動リンクの実装を示す．ただし，主にアプリケーション側での対策であり，開発キット固有の問題を解決しているものではないため環境に即した実装の変更が求められる．

6.1 ユーザのリンク文字列の誤認を防止する対策

ユーザのリンク文字列の誤認を防止する対策の例として，初めに図 9 で用いられた URL のプレビュー機能を挙げる．プレビュー機能によりリンク先が表示されるため，複数の URL がリンクされていることや，悪性サイトの情報を認識することができる．また，プレビュー部分は表示されたサイトへのリンクとなるため，正確に目的のサイトへのリンクをタップできることで悪性 URL のクエリパラメータによるタップ面積拡大による効果を軽減できるとも考えられる．

一方，ユーザ入力が生じ得る箇所すべてでの実装が求められるといったコスト面でのデメリットや，URL のプレビューの個数に上限を定めている場合には，その個数以上の URL が含まれるテキストには依然として本研究に示すセキュリティリスクが残されるといった問題もある．さらにテキスト中の URL が多数である場合はプレビューを個

別に確認することが難しいため，良性サイトのプレビューを装った悪性サイトのプレビューによっての誘導も考えられる．また，リンク先の情報を取得する機能自体に対しての攻撃も確認されており，プレビュー機能に脆弱性問題が発生するという新たなセキュリティリスクも考えられる．例えば，Kinugawa は URL のプレビューと類似した埋め込みコンテンツ機能の不備などにより Discord デスクトップアプリ [10] の任意コード実行の脆弱性を報告している [11]．プレビューの実装に問題がない場合も，不適切なサイト内容をプレビューによってユーザに提示する Sexting のような悪用手法が考えられる．

次に，リンクをタップした時点で遷移先の URL のみをポップアップにて表示し，遷移の可否を再び確認するといった対策が考えられる．これは主にオープンリダイレクトなどへの対策として使用されているもので，遷移先 URL のみを表示するため，本研究における URL の分割は生じえない．しかし，リンクの直感的な使用ができなくなる点において，ユーザの使い心地を損なうデメリットも考えられる．本対策ではドメインのみ，もしくは，URL の先頭部分を表示することがより効果的であると考えられる．理由として，チャットエリアにて URL の分割が発生していた場合に，長いクエリパラメータにより URL の後半部分を含めて表示してしまうと，ポップアップにおいての遷移先の再確認を正確に行うことが難しくなるためである．

6.2 悪性化する恐れのある入力を変更する実装

URL の悪性化を防ぐ仕組みとして，URL 自動リンクの実装においてスキームが欠落しているテキストを自動リンクする際に，画面表示内容に対してスキームを追加する手法が考えられる．これは URL の悪性化への根本的解決策とは言えないものの，ユーザに複数のリンクが含まれていることを明示できる点では有効であると考えられる．さらに，クリップボードへのコピーの際には，URL が 2 つであることが明確になるため，外部の URL チェックサービスでの検査が有効もしくはエラーとなることでの悪性 URL 発見の機会が増加する．ただし，元のテキストを変更してしまうため，情報の正確性を保証できないといった問題もある．

最も効果的なリスク軽減策として，自動リンクの対象と判断した文字列に対し，パーセントエンコーディングが必要な文字をすべてエンコードする処理を適用した後に，自動リンクを行うといった方法が考えられる．これによって URL が分割されることはなくなり，複数の環境でのリンク先が異なるものとなることも防止できる．しかし顔文字などの特殊な文字列を含むテキスト中に URL が混在する場合には，URL の後に続く顔文字などを自動リンクの対象であると判断してしまい，プレーンテキストとして表示したい箇所がエンコードされてしまうといった問題があるため，URL の後に続く文字列をどこまでエンコードするかの条件の決定が求められる問題も多い．特に日本語などマルチバイト文字をクエリパラメータに含む URL などは，日本語文字列中では URL 終端と本文とが区別できないといった問題もあり，実装の困難さが伺える．

表 4 各環境におけるセキュリティリスクの有無
A：Android アプリ版，W：Web 版

Table 4 Security risks for each feature
A：Android applications，W：Web applications

	アプリ A	アプリ B	アプリ C
（A）チャットエリア	×	○	○
（A）ポップアップ	○	未実装	未実装
（A）テキストコピー	○	○	×
（A）テキスト転送	○	○	○
（W）チャットエリア	×	×	○
（W）ポップアップ	未実装	未実装	未実装
（W）テキストコピー	×	○	×
（W）テキスト転送	未実装	未実装	未実装
セキュリティリスク：	○あり	×なし	

7. まとめ

本研究では，Android アプリの自動リンク機能を実現するために利用される開発キットにおける URL 中の文字の解釈に着目し，特殊な文字によって URL が分割されることによって悪性リンクが生成される，新たなセキュリティリスクについて述べた．また，Web 版と Android アプリ版を共に提供している場合に，各環境での自動リンク機能の差によって，特定の環境で信用された良性リンクが，別の環境で悪性化する問題を示した．さらに，5 億インストールを超える複数の Android アプリでの発生状況とその発生個所を調査し，結果として，数多くの Android アプリが本研究にて示したリスクを有し，数億人のユーザが悪性サイトに誘導される恐れがあることを明らかにした．発生個所の調査では，自動リンクされた URL の利用方法についても考察し，アプリ内に実装されているテキストの共有機能やコピー機能にかかわる部分においてのセキュリティリスクと実際の挙動についても調査した．Web 版と Android アプリ版におけるチャットエリアやポップアップにおける URL 悪性化の有無や，悪性 URL を含むテキストの共有やコピーにおいての伝播のリスクついて**表 4** にまとめる．各サービス間や Web 版と Android アプリ版の間での差異も大きく，自動リンクの開発者側で本問題について統一的な基準ができていないことが見て取れる．アプリ C においてのみ Web 版で URL 悪性化のリスクが確認されたが，これは Chrome 拡張によって Web 版が実装されていることに由来すると考えられるが，Web 版で利用されているライブラリなどの原因に関する調査は行っていない．

先に述べた問題に対して，対策となりえる機能の提案や

リスクを軽減する複数の URL 自動リンクの実装方法を示すとともに，自身が提案した対策や実装の問題点についても考察した．本研究では，自動リンク機能とその周辺実装の今一度の見直しが必要であることを示唆するとともに，各環境での機能の差異がユーザに重大なセキュリティリスクを発生させ得るといった問題点を浮き彫りにした．

今後，IDN が広く普及することも考慮し，IDN を含めた自動リンク機能に対しての各開発元独自の実装や利用環境に左右されることのない統一されたライブラリの開発が求められると考えられる．さらに自動リンク機能について，扱う文字やリンクとする範囲などの基準が明確でないため，様々な実装の間で仕様が異なっていたり，あいまいであることもセキュリティリスクを発生させる要因の 1 つであると考えられる．種々の自動リンクの間で実装の仕様やテスト仕様の基準を明確化することにより本リスクは解消されると考えられる．

なお，本研究で明らかとなった情報は Linkify 開発元，各サービス提供元へ報告し，脆弱性に該当しない，または，フィッシング情報は脆弱性として扱わないとの回答を得ていると同時に，一部アプリでは報告後の修正を確認している．

参考文献

[1] 鈴木宏彰, 森達哉, 米谷嘉朗. IDN ホモグラフ攻撃の大規模実態調査：傾向と対策. コンピュータセキュリティシンポジウム 2018 論文集, pp. 249–256, Oct. 2018.

[2] Shirakura Taiga, Hasegawa Hirokazu, Yamaguchi Yukiko, and Shimada Hajime. Potential Security Risks of Internationalized Domain Name Processing for Hyperlink. *IEEE International Workshop on Network Technologies for Security, Administration and Protection*, pp. 1092–1098, Jul. 2021.

[3] 白倉大河, 長谷川皓一, 山口由紀子, 嶋田創. 国際化ドメイン名の自動リンク処理等におけるセキュリティリスクの検討. コンピュータセキュリティシンポジウム 2020 論文集, pp. 29–36, Oct. 2020.

[4] Cisco Talos Intelligence Group. PhishTank | Join the fight against phishing. `https://phishtank.org/`. (Accessed on 08/19/2021).

[5] Jonathan Birch. Host/Split: Exploitable Antipatterns in Unicode Normalization. `https://i.blackhat.com/USA-19/Thursday/us-19-Birch-HostSplit-Exploitable-Antipatterns-In-Unicode-Normalization-wp.pdf`. (Accessed on 08/19/2021).

[6] Android Open Source Project. Linkify | Android Developers. `https://developer.android.com/reference/android/text/util/Linkify`. (Accessed on 08/19/2021).

[7] IETF. RFC3986. `https://datatracker.ietf.org/doc/html/rfc3986`. (Accessed on 08/19/2021).

[8] Google Developers. Download Android Studio and SDK tools | Android Developers. `https://developer.android.com/studio`. (Accessed on 08/19/2021).

[9] Google Developers. SNS におけるネット詐欺の手口と対策 | トレンドマイクロ is702. `https://is702.jp/special/3817/`. (Accessed on 08/19/2021).

[10] Discord Inc. Discord | Your Place to Talk and Hang Out. `https://discord.com/`. (Accessed on 08/19/2021).

[11] Masato Kinugawa. Discord デスクトップアプリの RCE. `https://masatokinugawa.l0.cm/2020/10/discord-desktop-rce.html`. (Accessed on 08/19/2021).