mailto スキームのエスケープについて

ＮＴＴ コミュニケーションズ株式会社
経営企画部
マネージドセキュリティサービス推進室
セキュリティオペレーション担当

2013 年 02 月 01 日

Ver. 1.0

1. 調査概要

1.1. 調査概要

Web アプリケーションにて、利用者から入力されたメールアドレスを用いる場合、SMTP Injection などに気をつける必要があるが、本文書では、mailto スキーム内に埋め込む際の注意点について検討した。

2. mailtoスキームでのエスケープ処理

2.1. 脆弱なWebページを想定する(掲示板)

以下のような Web ページを想定してみる

- 利用者からの投稿内容と共に、メールアドレスを受け取り、Web ブラウザ上に表示する
- その際に、メールアドレスには、mailto スキームでリンクを張る

サンプルとして、図 2.1-1を用意した。

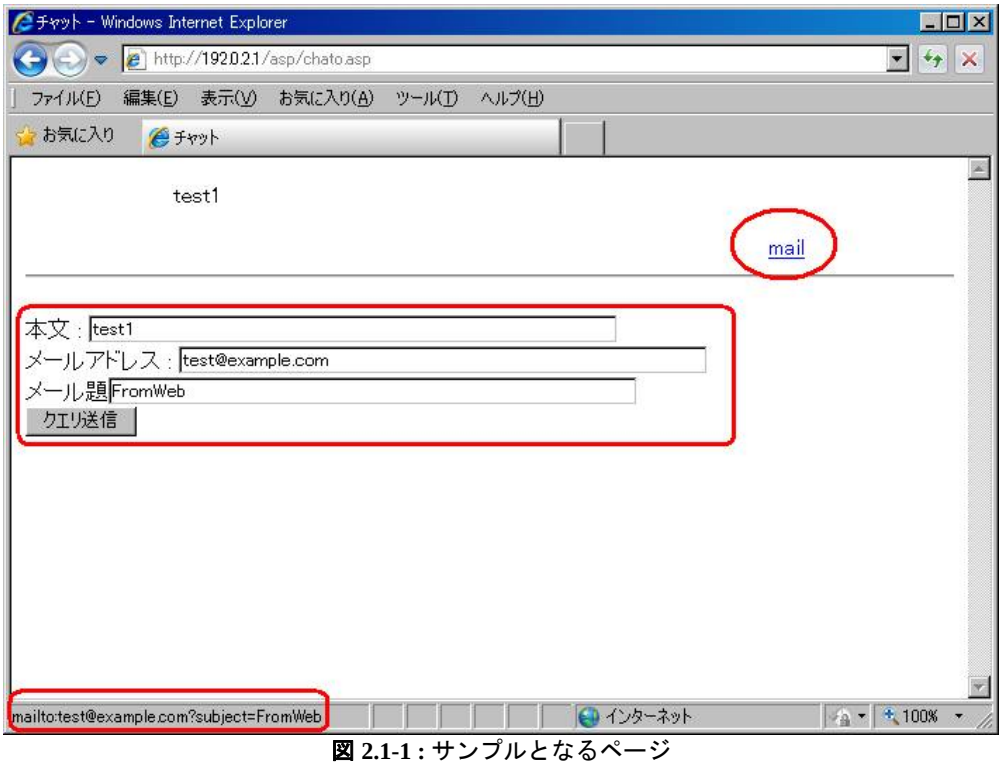

図 2.1-1 : サンプルとなるページ

「メールアドレス」と「メール題」いうテキストフォームに入力された情報が

mailto スキームのリンク先(画面上ではステータスバーに表示)として生成される

2.2. 想定して脆弱なWebページに不正行為を行う 1(メールアドレス)

図 2.1-1に対して、メールアドレスに「test@example.com?cc=test4@example.com&」を与えた結果が、図 2.2-1～図 2.2-2である。
入力データは、「@」が二つなど、メールアドレスとして不適切な部分があるため、入力値のバリーデーション時にエラーとなる可能性があるが、「?」のバリーデーションを考慮している Web ページは少ないのではないだろうか。
そのような場合、図 2.2-1、さらに図 2.2-2のようになり、Webブラウザからメーラー(MUA)に渡される際にこのWebサイトを構築したWebアプリケーション・プログラマが想定していないようなデータが渡されることになる(図 2.2-2)。

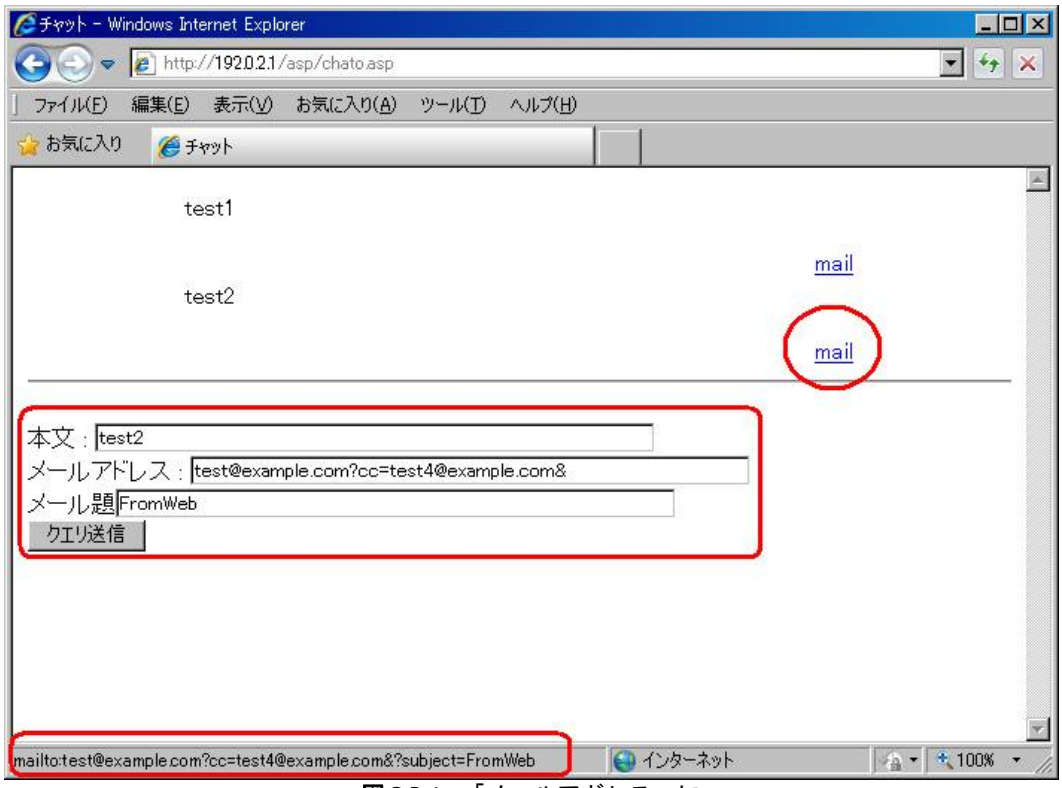

図 2.2-1 :「メールアドレス」に

「test@example.com?cc=test4@example.com&」というデータを与えた

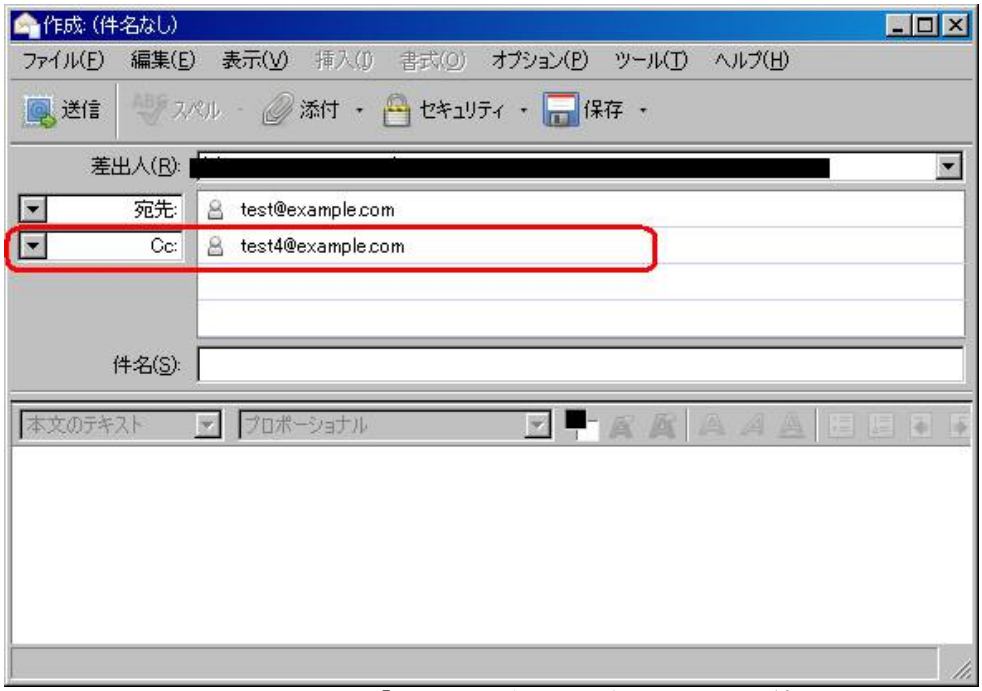

図 2.2-2 : 図 2.2-1の「mail」というリンクをクリックした結果

2.3. 2.2のエスケープ処理を実装する

「2.2 想定して脆弱なWebページに不正行為を行う 1(メールアドレス)」での問題点は、メールアドレスに「?」を与えることができ、かつmailtoスキームでエスケープされていなかったことである。
ということで、「?」をURLエンコードする処理を追加した結果が、図 2.3-1～図 2.3-2である。

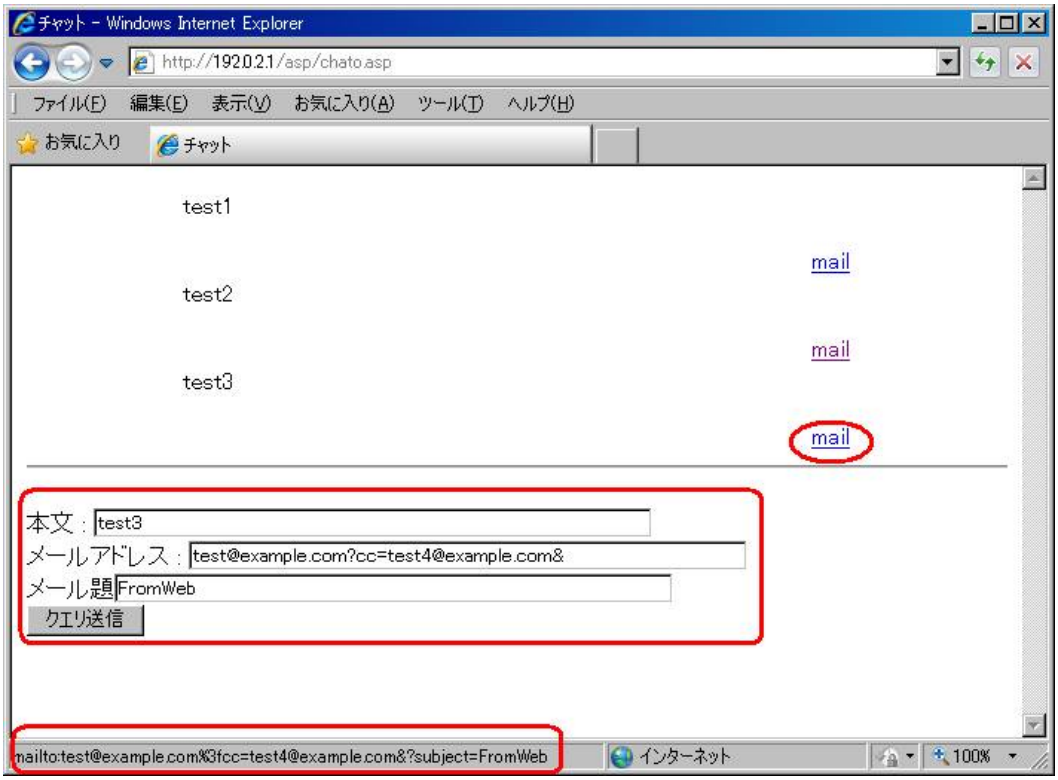

図 2.3-1 : 「メールアドレス」に

「test@example.com?cc=test4@example.com&」というデータを与えた

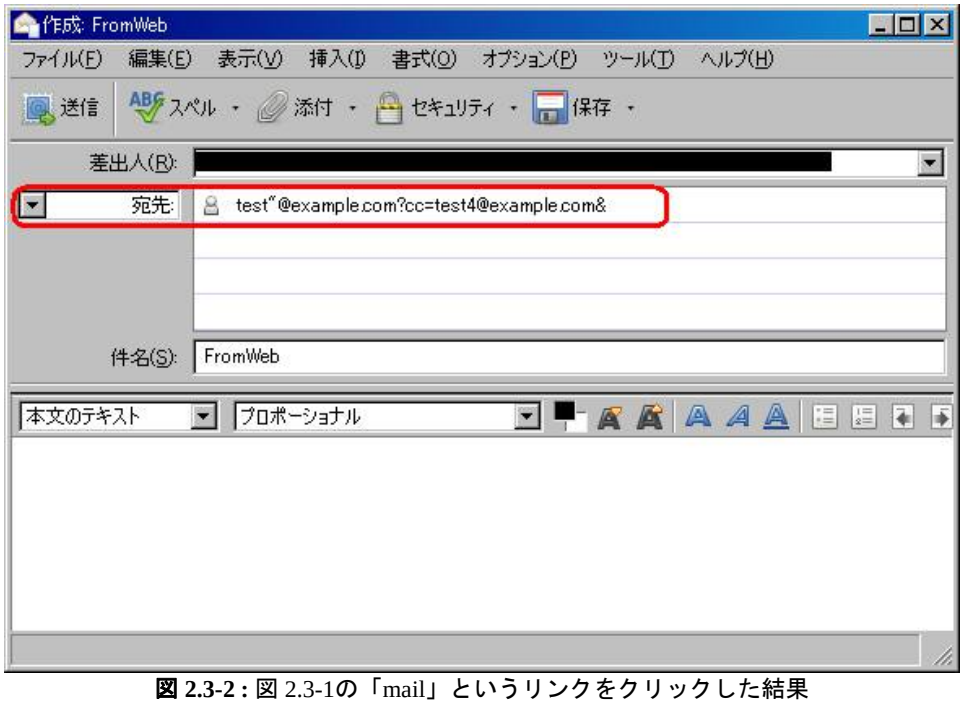

図 2.3-2 : 図 2.3-1の「mail」というリンクをクリックした結果

メールアドレスに入力したデータは「宛先」の中に閉じていることが確認できる

2.4. 想定して脆弱なWebページに不正行為を行う 2(mailtoスキームのクエリ文字列)

次は、mailto スキームに与えられるクエリ文字列の値が汚染データの場合を考えてみる。
クエリ文字列として「&」のエスケープ処理を怠っていると、Webブラウザからメーラー(MUA)に渡される際に任意のクエリ文字列が指定可能となり、このWebサイトを構築したWebアプリケーション・プログラマが想定していないようなデータが渡されることになる(図 2.4-2)。

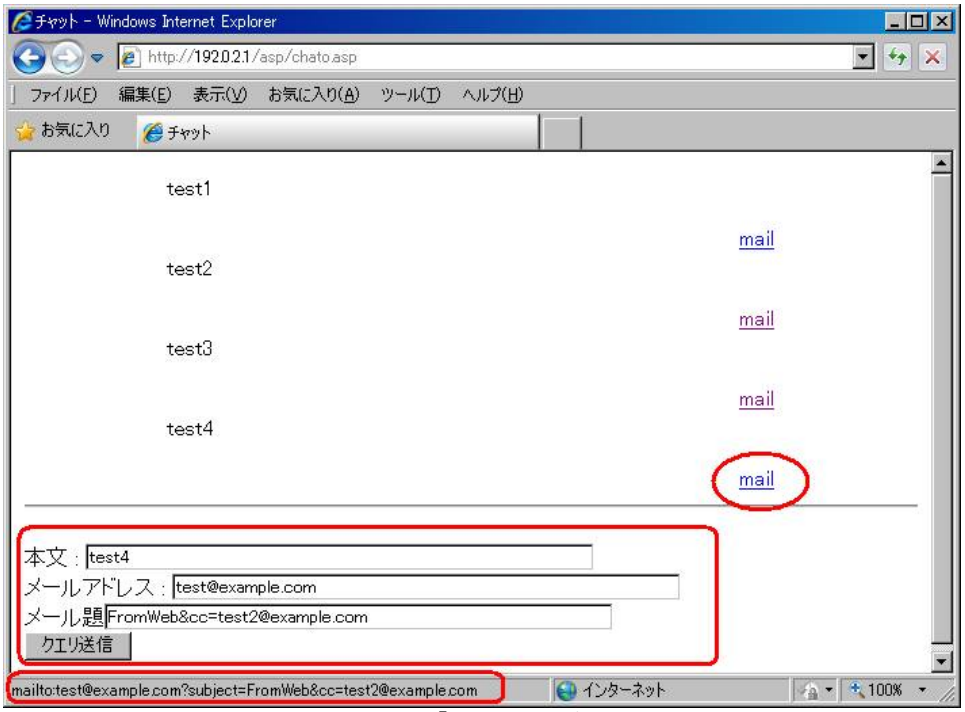

図 2.4-1 : 「メールの題名」に

「FromWeb&cc=test2@example.com」というデータを与えた

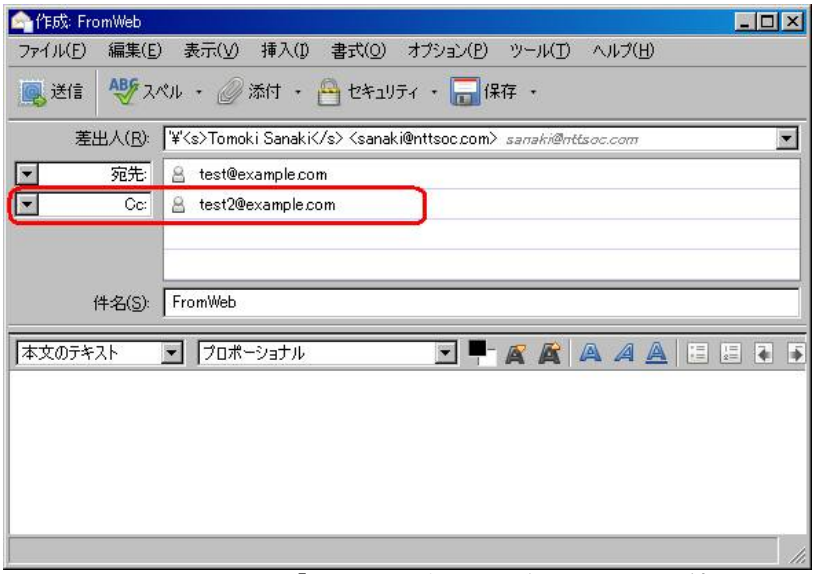

図 2.4-2 : 図 2.4-1の「mail」というリンクをクリックした結果

2.5. 2.4のエスケープ処理を実装する

「2.4 想定して脆弱なWebページに不正行為を行う 2(mailtoスキームのクエリ文字列)」での問題点は、クエリ文字列として「&」(場合によっては「=」も)を与えることができ、かつmailtoスキームでエスケープされていなかったことである。
ということで、「&」などをURLエンコードする処理を追加した結果が、図 2.5-1～図 2.5-2である。

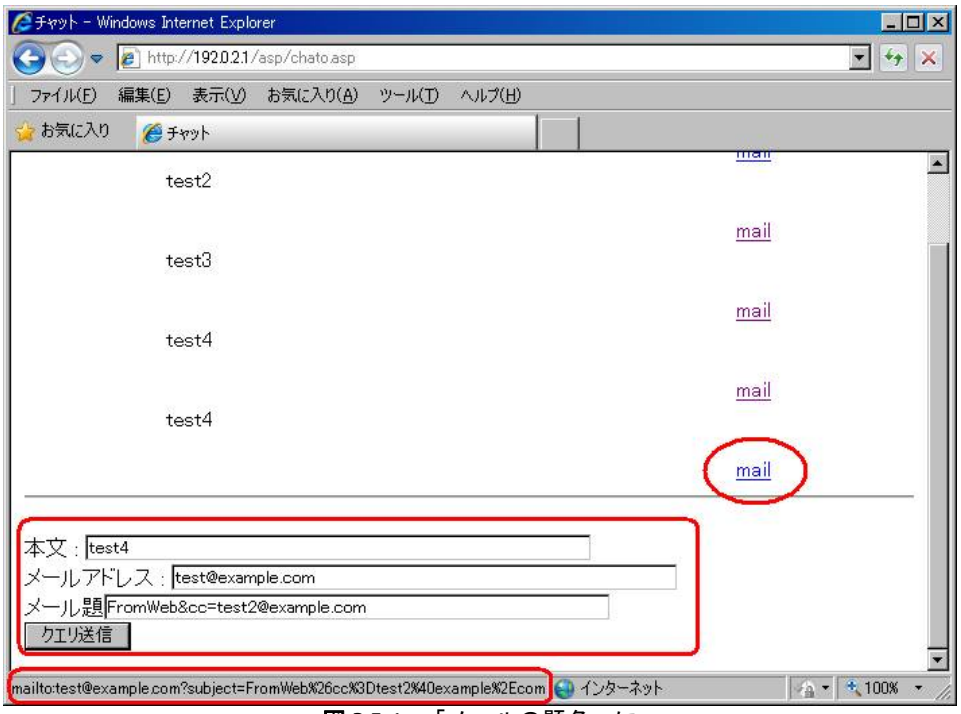

図 2.5-1 : 「メールの題名」に

「FromWeb&cc=test2@example.com」というデータを与えた

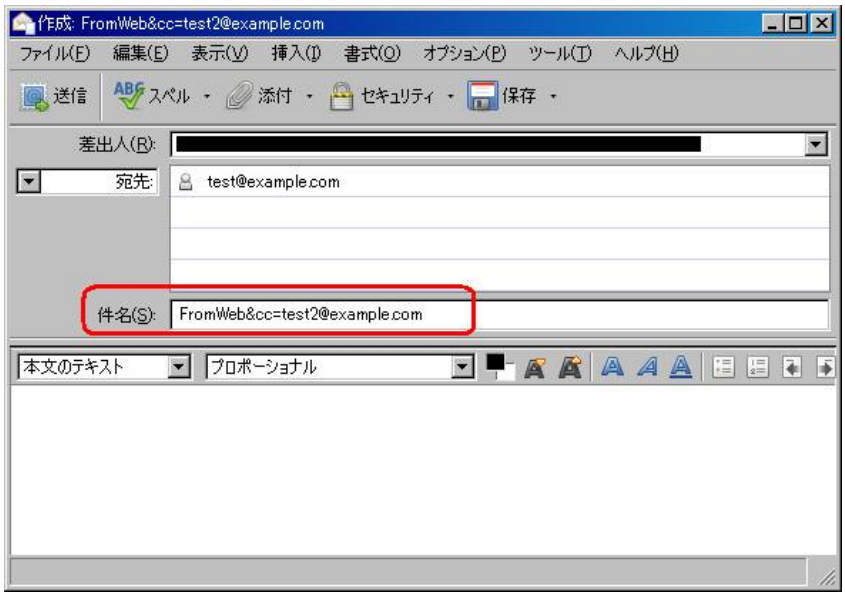

図 2.5-2 : 図 2.5-1の「mail」というリンクをクリックした結果

メールの題名として与えたデータがメーラーの「件名」の中で閉じていることが確認できる

2.6. とにかくURLエンコード処理を行う

「2.3 2.2 のエスケープ処理を実装する」、「2.5 2.4 のエスケープ処理を実装する」で、個別に検討したが、とにかくmailtoスキームのURIに使う際にはURLエンコードするという実装ではどうか試してみた結果が図 2.6-1～図 2.6-3である。
全体的に URL エンコードを実施しても、特に問題がなく正常に動作することが確認できる。

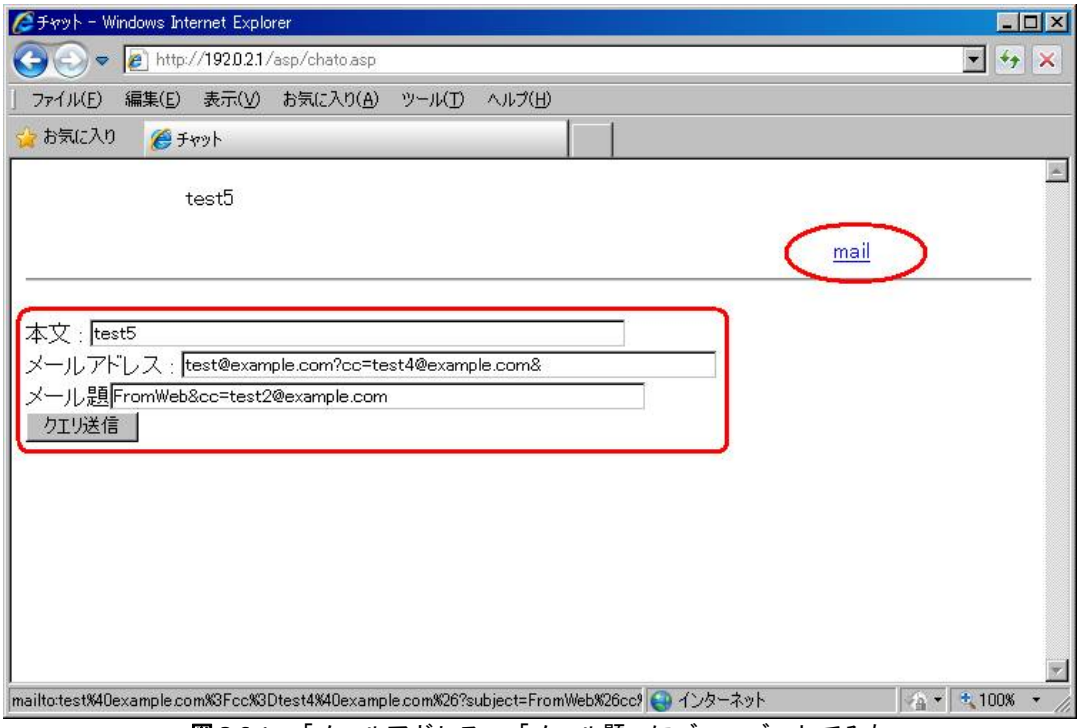

図 2.6-1 : 「メールアドレス」「メール題」にゴニョゴョしてみた

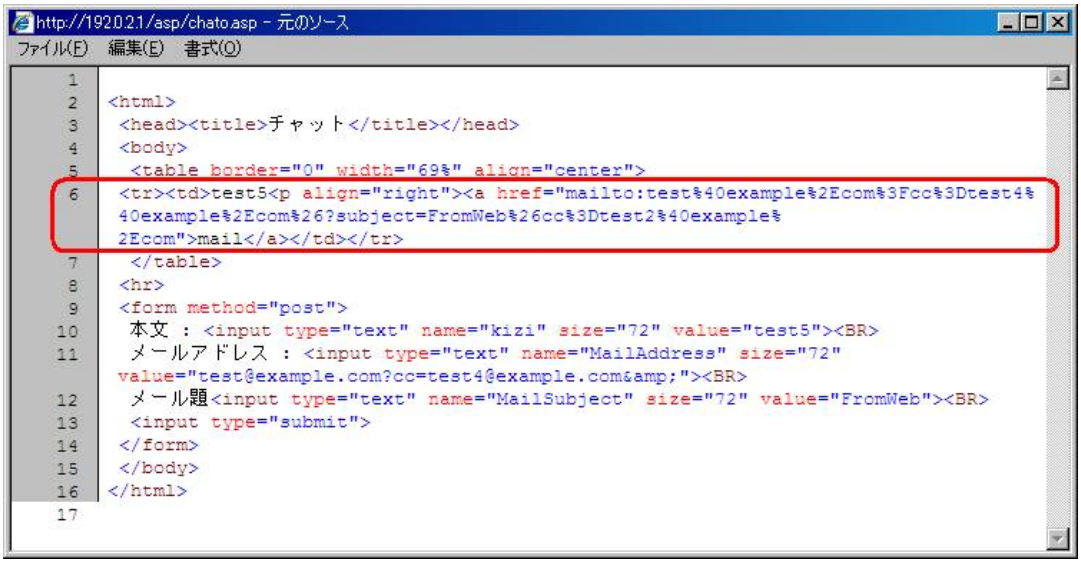

図 2.6-2 : 図 2.6-1のHTMLソース

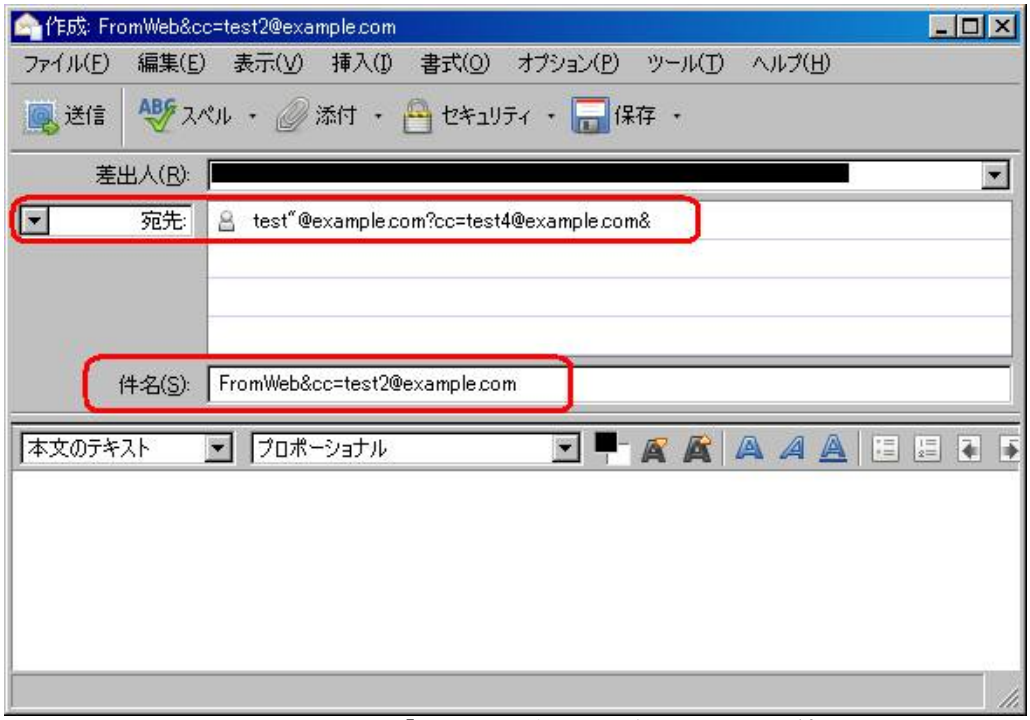

図 2.6-3 : 図 2.6-1の「mail」というリンクをクリックした結果

「メールアドレス」も「メールの題名」も与えたデータが

個々の枠内の中で閉じていることが確認できる

2.7. 対策

「2.6 とにかくURLエンコード処理を行う」の結果から、

◆ URI として出力する際に URL エンコード処理を行う

という至極一般的な結論となってしまったが、一応明示しておく。

- mailto スキームでのメールアドレス部分は URL エンコード処理を行う
  - 基本的には標準の URL エンコード処理関数を使えばよい
  - 個別に実装する場合、特に「?」を「%3F」へ URL エンコード処理を行う
  - 個別に実装する場合、特に"」を「%22」へ URL エンコード処理を行う
    - HREF 属性値を囲んでいる文字を URL エンコードする(この点は本文書の範囲外とする)
- mailto スキームでクエリ文字列の値や名前を指定する際、URL エンコード処理を行う
- メールアドレスして適切かどうか、入力値チェックを行う。(この点は本文書の範囲外とする)

2.8. まとめ(mailtoスキームの特殊性)

「入力データを mailto スキームの一部として利用する」という場面は、稀かとは思う。
また、利用者の操作(mailto スキームをクリックする。メーラーを使ってメールを編集するなど)があるため、実際の不正行為に悪用される現実的可能性は低いと思われる。

HTML の A タグの HREF 属性で通常使われる「http」スキームや「https」スキームは、Web アプリケーションへのリンクに使われ、一般的な Web アプリケーションでは、任意のクエリ文字列を与えることが可能なること自体は、別段セキュリティ事項とはならない。Web アプリケーションで使わないような任意のクエリ文字列が与えられても Web アプリケーションは無視すればよいからだ。
しかしながら、mailto スキームの場合、クエリ文字列上の変数が予約されているため(例えば、”subject”はメールの題名)、「任意のクエリ文字列変数が指定できる」というのは、Web アプリケーション・プログラマが想定していないメール属性を指定可能という点で、セキュリティ上の問題になりうるだろう。

3. 検証作業者

NTT コミュニケーションズ株式会社
経営企画部マネージドセキュリティサービス推進室
セキュリティオペレーション担当
　佐名木 智貴

4. 履歴

- 2013 年 02 月 01 日 ：ver1.0 最初の公開

5. 最新版の公開URL

http://www.ntt.com/icto/security/data/soc.html

6. 参考

- RFC2368 : The mailto URL scheme
- RFC3986 : Uniform Resource Identifier (URI): Generic Syntax
- パーセントエンコーディング
  http://ja.wikipedia.org/wiki/%E3%83%91%E3%83%BC%E3%82%BB%E3%83%B3%E3%83%88%E3%82%A8%E3%83%B3%E3%82%B3%E3%83%BC%E3%83%87%E3%82%A3%E3%83%B3%E3%82%B0
- Security of WebAppli&Mail
  http://rocketeer.dip.jp/secProg/MailSecurity001.pdf
- セキュア Web プログラミング Tips 集
  出版社：株式会社ソフト・リサーチ・センター (ISBN978-4-88373-256-2 C3004)
- Windows 汎用 COM コンポーネント BASP21 の SMTP コマンド・インジェクション問題について
  http://www.insi.co.jp/news/070330.html
- .NET Framework 上の SMTP Command Injection について
  http://www.ntt.com/icto/security/images/sr20110110.pdf
- .NET Framework に潜む脆弱性「SMTP コマンド・インジェクション」とその対処法
  http://www.atmarkit.co.jp/fdotnet/dotnetsecurity/dotnetsecurity01/dotnetsecurity01_01.html

7. 本レポートに関する問合せ先

NTT コミュニケーションズ株式会社
経営企画部
マネージドセキュリティサービス推進室
セキュリティオペレーション担当

e-mail: scan@ntt.com

以 上