FROM THE ARCHIVES: AUTOMATION OF FRAUD – THE VOXIS PLATFORM

August 2015

RSA® FirstWatch

TABLE OF CONTENTS

EXECUTIVE SUMMARY

RSA® FirstWatch monitors and analyzes the malicious activity of online cybercrime infrastructures on an ongoing basis.

During the recent months we identified a growing demand for tools to automate fraud related operations among the cybercriminals in their online communities and blackmarkets.

Voxis is a fraudulent platform used by criminals to monetize stolen credit card credentials and increase their illicit revenues by automating fake transactions through multiple payment gateways.

This report will cover the technical aspects and the modus operandi of the platform. It will also reveal some interesting facts concerning the inner workings of the platform and suggest mitigation steps for merchants and retailers.

A MULTI GATEWAY FAKE MERCHANT STORE

The Voxis Platform, described in the fraud related online communities as ”the most advanced CVV/fullz cash-out software,” is a fraud toolkit which is designed to send batches of stolen credit card charges to multiple payment gateways, automating their returns. As a result, Voxis claims to provide better cash-out success rates than most traditional fraud options.

The Voxis Platform was developed by the “Voxis Team”, allegedly a cybercrime group specializing in money laundering and led by a fraudster who is believed to use the alias *Bl4ckS14y3r*.

THE VOXIS PLATFORM IS LEAKED

The Voxis platform was first reported on in late October 2014 by *IntelCrawler*, a cyber-threat intelligence firm from Los Angeles. It was later covered by other security related websites and mainstream media. A few days after *IntelCrawler’*s publication, an unlicensed “starter” version of the Voxis platform was leaked in the fraud related online communities.

For this report, RSA® FirstWatch researchers have analyzed the leaked version in order to better understand the makings of what appears to be a sophisticated platform.

PAYMENT GATEWAYS

To understand how Voxis works, one needs to understand how payment gateways work. When a legitimate user pays for a service offered by an online merchant, the approval for the transaction goes through several service providers. First in line is the merchant's online store, where the buyer fills a shopping cart until ready to check out. When a checkout is requested, all the information about the transaction is then forwarded by the merchant's web server to the payment gateway, which in turn will send it for approval by the payment processor and from there to the acquiring bank.

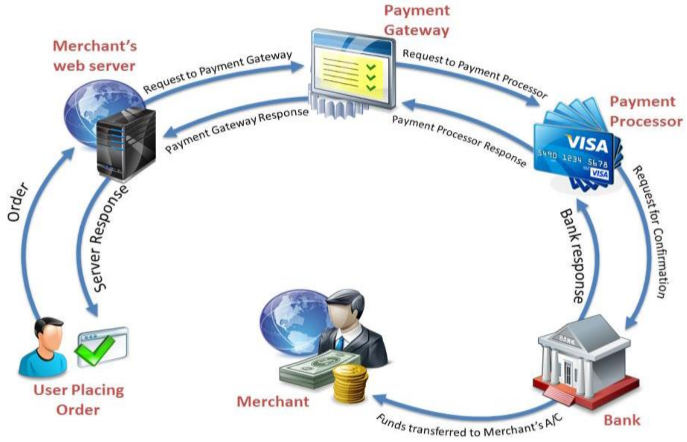


FIGURE 1:

The payment cycle

When all parties approve the transaction, an approval will be sent back to the merchant's web server and the money will be transferred by the acquiring bank to the merchant's account.

The way Voxis works is by "pretending" to be a merchant's web server, so it will be able to communicate with up to 32 different payment gateways, making false transactions using stolen credit card credentials – all without human interaction.

Generally speaking, in order to obtain an account on a payment gateway, a merchant needs to prove that it is legitimate. Most payment gateways do this by requiring various documents used to identify the merchant's identity and business before granting a merchant account. To overcome this obstacle, fraudsters use money mules[1] and stolen identities that give them the necessary resources to open an account with limited risk of being discovered. They then build fake web sites and supply fraudulent or stolen documents to get the approved merchant accounts.

However, paying for the payment gateway services is a different story. These services can cost several hundred dollars per month, not including the additional charges for each transaction. Managing accounts on several payment gateways can easily push the cost to thousands of dollars per month for fraudsters. This would imply that in order to take full advantage of the Voxis platform capabilities, fraudsters would need to go through the effort of establishing the merchant accounts and paying for the services before being able to move on to the actual cash-out process.

TECHNICAL ANALYSIS

On our initial assessment, the PHP-based source code of the Voxis platform gives the appearance of having been built by an experienced team of web developers (or possibly one individual). This leads our researchers to believe that Voxis was probably not this team's first project in the fraud business.

The first portion of the Voxis code we analyzed was a licensing module, hidden deep inside one of the project's folders. This licensing manager was based on a PHP class that our team was able to trace back to a PHP developer out of India who posted it online for free use. It is not believed that this individual is part of the core team.

From the analysis, it appears that the Voxis dashboard was built on top of a common web template. A simple web search shows that it is a popular web template, being sold for under $30 USD by different online marketplaces.

This is not the first time the researchers observed fraudsters using such multipurpose administration dashboard themes. In fact, it is becoming very popular among the "low-cost" projects, because it allows developers to spend less time on the general design of their platform and instead focus on core functionalities.

The Voxis dashboard (Figure 2) is divided into several sections, ordered in a very intuitive structure that basically guides the fraudster step-by-step to their illegitimate cash-out goal. After a successful login, the main screen of the system provides the fraudster with general statistics on the attack:

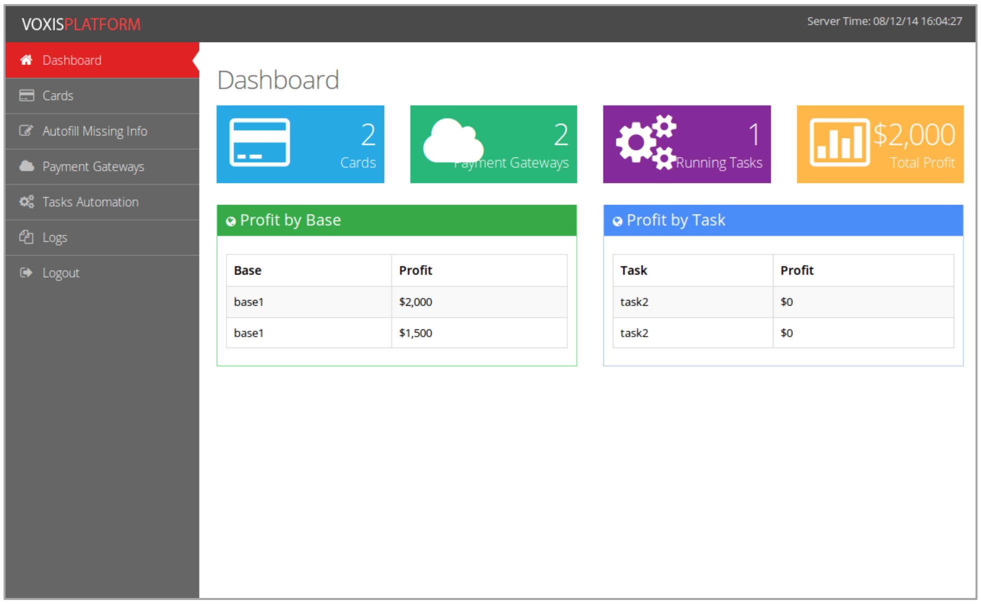
- The amount of stolen credit cards in the system, divided into "bases" (an underground term for a list of stolen credit cards stolen from the same location/merchant or by the same group)
- The number of payment gateways used for the cash-out purpose
- The number of tasks that are currently running in the system
- The total profit of the entire operation

The fraudster can also see the exact profit for each base/task in the attack and make decisions based on the most profitable ones.

[1] "Money mules" are people who are used to transport and launder stolen money or some kind of merchandise. Fraudsters may recruit money mules to use stolen credit card and other sensitive information

FIGURE 2:

The Voxis dashboard



Profit by Base

Base	Profit
base1	$2,000
base1	$1,500

Profit by Task

Task	Profit
task2	$0
task2	$0

The remainder of the items in the menu is used to configure the operation. The "Cards" menu is where the fraudster first needs to upload stolen credentials into the system. Just like the rest of the system, the cards screen is very intuitive. It allows the fraudster to upload the lists in several different formats, based on the information the fraudster has in a credit card lists. This information mostly depends on the origin of the stolen credit cards batch – typically either obtained by infecting PoS systems with credit card "grabbing" malware or by purchasing the lists in an underground credit card blackmarket.

Payment gateways sometimes require additional information from a client to approve a transaction. When a batch of credit card numbers are loaded into the system, the next step for the fraudster is to attempt to fill in the missing information on each one. The Voxis platform assists in this by querying different web based search engines using predefined APIs[2].

At the end of the information gathering process, the fraudster will need to obtain the following information for each one of his stolen credit cards for a successful cash-out:

```
firstName|lastName|number|expiryMonth|expiryYear|startMonth|startYear|cvv|issueNumber|type|billingAddress1|billingAddress2|billingCity|billingPostcode|billingState|billingCountry|billingPhone|shippingAddress1|shippingAddress2|shippingCity|shippingPostcode|shippingState|shippingCountry|shippingPhone|company|email
```

The next screen of the Voxis platform is the "Payment Gateways" page, where the fraudster can manage all of the available gateways. The attack theoretically supports up to 32 different payment gateways systems, however, there appears to be some major bugs with some of the payment gateways are present (which will be explained later on in the report).

The Payment Gateways page comes with a "wizard" feature to help the fraudster determine the exact information needed for each of the payment gateways. The fraudster would add the API credentials for each payment gateway and from there the fraudster he can continue with the fraud scheme.

The last setup stage occurs on the "Tasks Automation" page, where the fraudster can define all of the needed properties for his automated tasks:

- The amount of money to transact from each credit card

[2] Application Programming Interface

- The limit to transact from each card in a 24 hour period
- The currency to use for the transactions
- Whether the transaction will take place during daytime or nighttime (daytime: 08:00-21:00, nighttime: 21:01-07:59)
- Which payment gateway to use for the cash-out

The fraudster can start or pause each one of the tasks, edit their parameters or limit them to a specific time period.

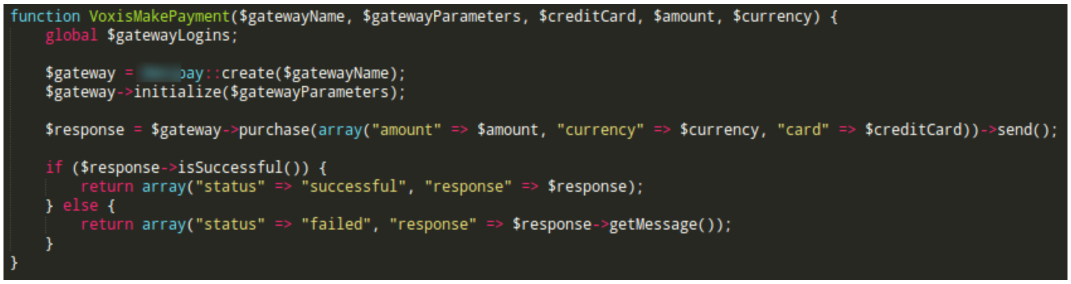
The last component that allows the entire automation to take place is the *crontab.php* file, which is responsible for the cash-out. The fraudster first needs to configure a scheduled job on Cron (a time-based job scheduler in Linux based operating system) on his server to execute *crontab.php* every several minutes. Once *crontab.php* is executed, it pulls all the tasks that need to take place at this specific time, the credit card lists and the payment gateways. With all the relevant information in hand, the crontab.php page just calls the function *VoxisMakePayment()* to complete the rest of the cash-out – and this is where our analysis got interesting.

BORROWED SOURCE CODE

During the analysis, RSA researchers attempted to determine whether the Voxis platform can fulfill its promises for a profitable cash-out. After diving into the code responsible for making the transactions inside *VoxisMakePayment()*, a vast PHP library that provides a consistent API to all the different payment gateways supported by the Voxis platform was revealed.

From the software developing side, having such a library makes life easier for the fraudster, who won’t have to apply each one of the different gateway-specific APIs directly in his code. It was clear that many work hours were invested in developing such a well-structured PHP library. But who did this job? Was it the Voxis Team?

Further examination of the PHP library code revealed that it was fully borrowed, almost as is, from a public web-based repository hosting service project called ****pay, managed by several individuals. In fact, it appeared that the Voxis Team decided to skip some of the payment gateways supported by ****pay, probably due to low cash-out reasons.



```
function VoxisMakePayment($gatewayName, $gatewayParameters, $creditCard, $amount, $currency) {
    global $gatewayLogins;

    $gateway = [redacted]ay::create($gatewayName);
    $gateway->initialize($gatewayParameters);

    $response = $gateway->purchase(array("amount" => $amount, "currency" => $currency, "card" => $creditCard))->send();

    if ($response->isSuccessful()) {
        return array("status" => "successful", "response" => $response);
    } else {
        return array("status" => "failed", "response" => $response->getMessage());
    }
}
```

FIGURE 3:

Screenshot of Voxis code using ****pay’s API

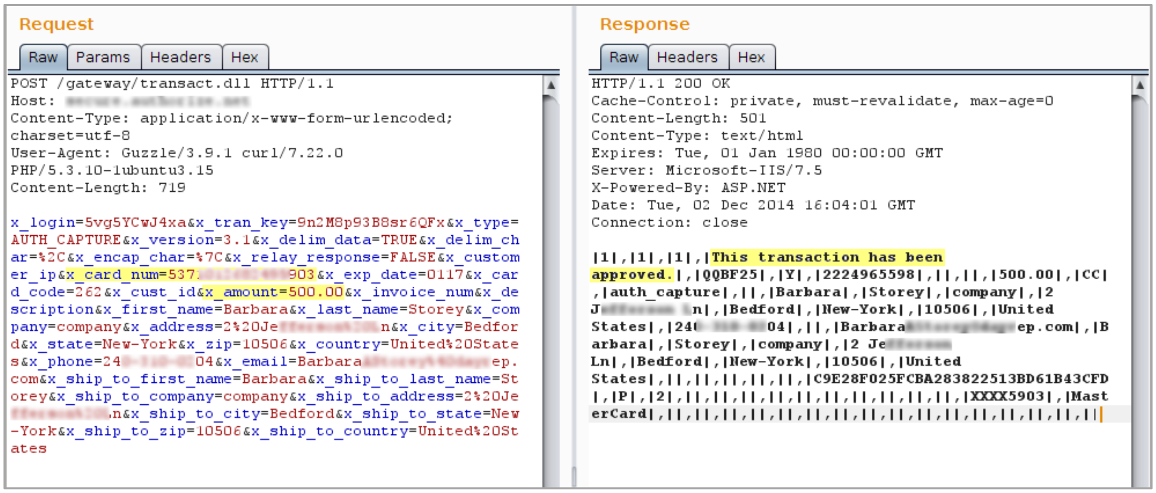
Basically, the Voxis platform can be described as the GUI interface and tasks scheduler for the ****pay module. The rationale of building a project based on several free to very cheaply priced modules is that it leads to larger profits for the developers, compared to the resources that were invested while developing the project itself.

POOR CODE IMPLEMENTATION

In order to analyzing the Voxis platform, several test accounts on selected payment gateways were opened, the system filled in the required information (all the details were made up for testing purposes), created several tasks and started using the system. The first test appeared to yield accurate results: the Voxis platform successfully communicated with the payment gateway over SSL and forwarded the information that was inserted into the attack.

In fact, it was all up to ****pay to do the job. As long as the Voxis Team implemented the ****pay library well enough in their code, the transaction should be sent to the payment gateways and get approved immediately. But this was always not the case.

In a different payment gateway, after calling *crontab.php* to make the transaction, the predefined credit card was **not** sent to the payment gateway server. Further analysis showed that a bug appears to be present that caused an incorrect implementation of the ****pay library in the Voxis side for certain offsite payment gateways. This bug redirected the user to another website to complete the transaction and require a slightly different procedure to be taken.



```
POST /gateway/transact.dll HTTP/1.1
Host: [redacted]
Content-Type: application/x-www-form-urlencoded;
charset=utf-8
User-Agent: Guzzle/3.9.1 curl/7.22.0
PHP/5.3.10-1ubuntu3.15
Content-Length: 719

x_login=5vg5YCwJ4xa&x_tran_key=9n2M8p93B8sr6QFx&x_type=
AUTH_CAPTURE&x_version=3.1&x_delim_data=TRUE&x_delim_ch
ar=%2C&x_encap_char=%7C&x_relay_response=FALSE&x_custom
er_ip&x_card_num=537[redacted]903&x_exp_date=0117&x_car
d_code=262&x_cust_id&x_amount=500.00&x_invoice_num&x_de
scription&x_first_name=Barbara&x_last_name=Storey&x_com
pany=company&x_address=2%20Je[redacted]n&x_city=Bedfor
d&x_state=New-York&x_zip=10506&x_country=United%20State
s&x_phone=24[redacted]04&x_email=Barbara[redacted]ep.
com&x_ship_to_first_name=Barbara&x_ship_to_last_name=St
orey&x_ship_to_company=company&x_ship_to_address=2%20Je
[redacted]n&x_ship_to_city=Bedford&x_ship_to_state=New
-York&x_ship_to_zip=10506&x_ship_to_country=United%20St
ates
```

```
HTTP/1.1 200 OK
Cache-Control: private, must-revalidate, max-age=0
Content-Length: 501
Content-Type: text/html
Expires: Tue, 01 Jan 1980 00:00:00 GMT
Server: Microsoft-IIS/7.5
X-Powered-By: ASP.NET
Date: Tue, 02 Dec 2014 16:04:01 GMT
Connection: close

|1|,|1|,|1|,|This transaction has been
approved.|,|QQBF25|,|Y|,|2224965598|,||,||,|500.00|,|CC|
,|auth_capture|,||,|Barbara|,|Storey|,|company|,|2
J[redacted]n|,|Bedford|,|New-York|,|10506|,|United
States|,|24[redacted]04|,||,|Barbara[redacted]ep.com|,|B
arbara|,|Storey|,|company|,|2 Je[redacted]
Ln|,|Bedford|,|New-York|,|10506|,|United
States|,||,||,||,||,||,|C9E28F025FCBA283822513BD61B43CFD
|,|P|,|2|,||,||,||,||,||,||,||,||,||,||,|XXXX5903|,|Mast
erCard|,||,||,||,||,||,||,||,||,||,||,||,||,||,||,||,||
```

FIGURE 4:

A successful transaction made through one of the payment gateways. On the left: the request sent by the Voxis platform. On the right: the response of the gateway where it accepted the transaction

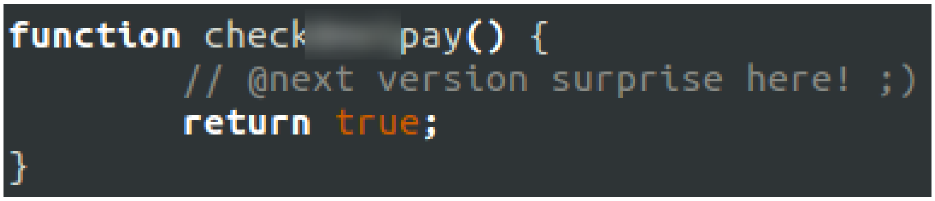
VOXIS PLATFORM VS. FRAUD SCREENING SOLUTIONS

The Voxis Team appears to possess the ability to avoid fraud detection solutions implemented by payment gateways to block false transactions. They claim to achieve this by mimicking legitimate human interaction with the different payment gateways, causing it to look like real people are sending the credit card information to the different gateways.

Voxis does achieve this partially, by enabling the fraudster to decide how much money to cashout from each card, while providing the system 24 hours to do so.

CONCLUSIONS

Judging by the spike in credit card breaches during recent years, it seems only natural that fraudsters are constantly seeking automated solutions for a quick and reliable monetization and cash-out. We believe the Voxis platform indicates the direction many other fraudsters will be going in the future. The underground blackmarkets are showing a high demand for automation of fraud and this demand is going to be fulfilled as long as there are opportunistic fraudsters.



```
function check[redacted]pay() {
        // @next version surprise here! ;)
        return true;
}
```

FIGURE 5:

A screenshot of Voxis code, showing a function that will be added to its next version

MITIGATION STEPS

As the Voxis platform and similar tools aim to automate the monetization of stolen credit cards, there are several mitigation steps that will help merchants and retailer organizations to protect their customer's data from being stolen.

Reduce the attack surface - Restrict internet access to a whitelist-based approach and block any unnecessary services that can be exposed and later on abused by attackers, such as: VNC, RDP, SSH and FTP. Change all the default passwords, choose strong and complex passwords to protect from dictionary attacks, and never allow authentication without any password at all. Apply software security patches from reliable sources on a regular basis.

Implement EMV technology, also known as "Chip and PIN". EMV won't prevent breaches, but it can lower fraudster motivation to attack an organization, reducing risk for company and customers.

Apply P2PE (Point-to-Point Encryption) - ***This is by far the most effective mitigation step***. All sensitive information is encrypted right from the entry point on the swiping device, and it renders the RAM scraping method almost useless.

Device and network monitoring solutions - **RSA® ECAT** can help in monitoring employee endpoint devices; **RSA® Security Analytics** can help monitor the corporate network; and **RSA FraudAction™** services can help enhance and enrich perimeter protection and help keep managers and security personnel up to date with the most recent and relevant threats to your organization.

Follow the PCI-DSS regulations – This does not provide full protection, but it is the required minimum for storing sensitive payment information.

Two-Factor Authentication (2FA) – Adopt this type of authentication across the entire network to lower the risk of compromise (RSA SecurID® and RSA® Adaptive Authentication may help you to achieve this goal).

AUTHOR AND CONTRIBUTORS

AUTHOR

- Lior Ben-Porat

CONTRIBUTORS

- Uri Fleyder
- Ami Kaufman
- Gabriel Glusman

Content and Liability Disclaimer

This Research Paper is for general information purposes only, and should not be used as a substitute for consultation with professional advisors. EMC has exercised reasonable care in the collecting, processing, and reporting of this information but has not independently verified, validated, or audited the data to verify the accuracy or completeness of the information. EMC shall not be responsible for any errors or omissions contained in this Research Paper and reserves the right to make changes anytime without notice. Mention of non-EMC products or services is provided for informational purposes only and constitutes neither an endorsement nor a recommendation by EMC. All EMC and third-party information provided in this Research Paper is provided on an "as is" basis.

EMC DISCLAIMS ALL WARRANTIES, EXPRESSED OR IMPLIED, WITH REGARD TO ANY INFORMATION (INCLUDING ANY SOFTWARE, PRODUCTS, OR SERVICES) PROVIDED IN THIS RESEARCH PAPER, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

In no event shall EMC be liable for any damages whatsoever, and in particular EMC shall not be liable for direct, special, indirect, consequential, or incidental damages, or damages for lost profits, loss of revenue or loss of use, cost of replacement goods, loss or damage to data arising out of the use or inability to use any EMC website, any EMC product or service. This includes damages arising from use of or in reliance on the documents or information present on this Research Paper, even if EMC has been advised of the possibility of such damages.

ABOUT RSA

RSA's Intelligence Driven Security solutions help organizations reduce the risks of operating in a digital world. Through visibility, analysis, and action, RSA solutions give customers the ability to detect, investigate and respond to advanced threats; confirm and manage identities; and ultimately, prevent IP theft, fraud and cybercrime. For more information on RSA, please visit www.rsa.com.

H13742