

LENAML

A "VZnet Netzwerke" Crawler

"LenaML" is basically a (Web)Crawler. The primely focus of LenaML is to logon at either "studiVZ", "meinVZ" or "schuelerVZ" and crawl through each user profile. Next it extracts all desired information off these user profiles and store them into a MySQL-Database. The name "LenaML" has been chosen by myself to honor the winner of the german casting show USFO (Unser Star Für Oslo), Lena Meyer-Landrut.

Florian Strankowski, Leuphana University of Lueneburg

21.03.2010 - updated 27.04.2010

Disclaimer: This proof of concept code is for educational purpose only. Please do not use it against any system without prior permission. You are responsible for yourself for what you do with this code.

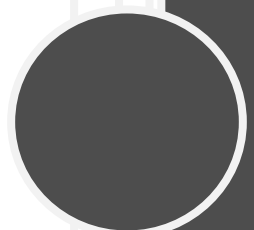


TABLE OF CONTENTS

1. PREAMBLE
2. SOURCECODE-ANALYSIS (VZ)
 - 2.1 LOGIN
 - 2.2 LOGOUT
 - 2.3 CRAWLING
3. PROGRAMMING
 - 3.1 BYPASS CHECKS
 - 3.2 THE PROGRAM
4. MANUAL
5. SOME STATISTICS
6. CONCLUSION

1. PREAMBLE

The purpose of this POC-Code is to show up what is possible nowadays. Due to the reason that it is not ment to be used against any of the VZ-Network (VZ) portals, it is clear that this code is for educational purposes only. If you still want to use this code against those portals you are responsible for what you do.

The social networking portals "studiVZ", "meinVZ" and "schuelerVZ" with currently over 16 million users are germanys biggest and most often used social networking websites. In fact, there is a reason to protect the quite sensitive information which is beeing published by the users on those social networking sites.

In 2006 and October 2009 the VZ has been targeted by a small amount of people who successfully tried to crawl the VZ-Networks. The fact that the VZ-Websites are beeing checked by the german TÜV proofs that these websites are secure and the people behind do their best to ensure that the sensitive data is beeing protected in the most efficent way - or probably not?

This paper describes my way of analysing the structures and methods behind VZ-Networks to prevent ripping off userdata and ofcourse howto bypass these checks.

2. SOURCECODE-ANALYSIS (VZ)

To successfully rip off all userprofiles using a crawler, we need to know what we need todo so. The first step is the analysis of the login-procedure. Then, one need to look for the logout-procedure. Crawling through the pages is the last step which needs to be taken care of.

2.1 Login

To capture the login-procedure i'd like to suggest the Firefox-Addon "Live-HTTP-Headers" which is available for free. Using this addon one is able to capture the headers sent while logging into any of the VZ-Portals.

<https://secure.schuelervz.net/Login>

POST /Login HTTP/1.1

Host: secure.schuelervz.net

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; de; rv:1.9.2.2)

Gecko/20100316 Firefox/3.6.2

Accept: (...)

Accept-Language: (...)

Accept-Encoding: gzip,deflate

Accept-Charset: (...)

Keep-Alive: 115

Connection: keep-alive

Referer: http://www.schuelervz.net/

Cookie: (...)

Content-Type: application/x-www-form-urlencoded

Content-Length: 307

email=fs%40coresec.de&**password**=mypassword&**login**=Einloggen&**js Enabled**=true&**formkey**=286c0d554cb0cbef9ddc2894617de33036f97d705546006b8f39dbe51d7fd621579aaac5b9741b08b09d6f0eedc71d0550ec7c51a85143992d076e92eef531f0a9cde3682afb14e21f50b952896ffa04dfbca56eeb351613ae7b1201eb0c317c&**iv**=753415e16d79aca410864d7551c4a4ab

Everything which is of importance has been highlighted. Next step is to bring all these information to a level we can work with.

<https://secure.schuelervz.net/Login> : Platform specific login-URL.

email : It is obvious that this is the email-address we're using to login.

password: As obvious as the email.

login : Needs to be set to "Einloggen" if we're going to login.

jsEnabled : Tells whether we're going to use JS - ofcourse we need this.

formkey: A randomly generated, valid key which is required to logon.

iv : Also a randomly generated, valid key which is required to logon.

Looks like most needed variables are static - obviously "formkey" and "iv" are not. The question is howto get those random variables to login.

To get these variables we're going one step back and visit the Website <http://www.schuelervz.net> - without logging in. Ok - here we got something in the sourcecode :

```
</script>
<input type="hidden" name="formkey"
value="41844e15b96636e8f0027a924f2bfe9456f5c76423d1fc2c1f4c651
c16fb311166819c3aa03fc92fdf1756bc62de459b962839059b42908068d77
e5c3f4c703f6969f7d2d250ace9e0b0671ef213f3c422af2ea79c368c77636
f2fb25dd62922" />
<input type="hidden" name="iv"
value="c8ad214fd1c5cbfb87effc64374ee43d" />
```

And also some lines below we'll find exactly the same again :

```
</fieldset>
<input type="hidden" name="formkey"
value="f675c8d19404159ddfcd434a7ee2cfc2fe91b65c87d6bd99b8a92e3
c1f7f6292d1696gc3c436a3e8fd64c7b00e6797673a0120e797a27ed2c101e
4e193021aab8600e4227c6fbbbeb7bf36fd27496f2202cff51fe4866b5bfff8
1779de9584d7653ac7d1edc3b87636277448b3dd83768" />
<input type="hidden" name="iv"
value="6a45073dca0ed98fa5a4920330bb6d30" />
```

The question which one is the right one can be answered by simply test which "formkey" and "iv" work. I wont show this now but it is the first one, with the </script> tag before the <input ..> field.

Finally we got everything together to login. The next step is to find out howto logout.

2.2 Logout

Ok, next step is to find out the logout-procedure. Same as for the login applies here - using "Live-HTTP-Headers" will provide us with the necessary information.

When logging out, the following URL will be used to clear the users session-data :

```
http://www.schuelervz.net/Logout/15b16a7d12d71ce092528c705a734419/tid/127
```

Again, most of the URL is static, the only alphanumeric string which is being used to logout is the one between Logout/(...)/tid. We'll find this key after logging into our account right in the sourcecode of our landingpage, the users home.

```
<li><a href="/Logout/cf04398b52ea826862e2b610ad27eedb/tid/127" class="logout" rel="nofollow" title="Raus hier">Raus hier</a></li>
```

Again we got everything to reconstruct the logout - fine.

2.3 Crawling

Due to the fact that the developers quite often change things on the website, i had to find a good way to easily see howto crawl through the user profiles. Furthermore, the unique User-IDs have been changed from static to variable IDs, atleast this is what we see when we access a userprofile. Everytime you logon, with a different account, someones user-id looks different to previous requests using other accounts. This makes it quite hard to collect data without having multiple duplicate entrys using the user-id as primary key in a database (primary keys may only occur once in a db).

To solve this problem i decided to construct a database using a unique-key which covers 3 columns at a time:

- The URL of the profile image
- The Name
- The School-ID

The picture-URL might change once a user uploads a new profile-image but the chance of filtering out double entries is quite successful. The Name might be changed every 3 months, so this is an even more effective way. The School-ID most likely never changes. Using an INDEX over those three columns one can build a nearly 100 percent clean database without double entries. The reason for this fact is that the combination of three columns for an INDEX using UNIQUE identifiers takes care of all three columns - only if all columns are exactly the same two entries become marked as a double entry .

Furthermore i am using groups and friend-pages to collect user-profiles. The reason for this is that most users join a group of their interest and those groups consist of a lot of users (mostly 50.000+ users). The sourcecode for our crawler is nearly the same for both examples. Here i'll just show parts of the sourcecode fetching users using groups. Ofcourse, using the friends-pages its possible to collect much more profiles - because not everyone joins a group. But i wont show this in here right now - maybe in a later version.

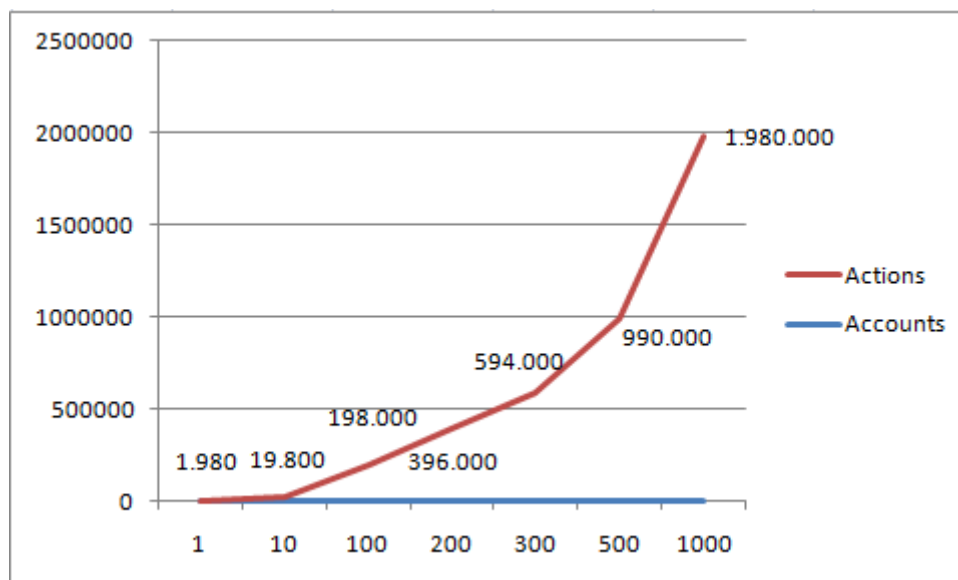
3. PROGRAMMING

Now it is time to put the information we collected together and build a working crawler. One more thing needs to be taken care of right now - the crawler-prevention which may stop us from using a crawler.

3.1 BYPASS CHECKS

The developers of VZnet ofcourse have taken care of bots and crawlers **but** the implementation of their so called "anti-crawler-system" does not prevent one from using a crawler at all. The only thing which prevents us from crawl thousands of user profiles is the fact that VZnet implemented a "click-check". This click-check counts the actions a user performs per day. The limit has been set to round about 1980 actions (clicks, page-impressions, whatever) per day. Why it is 1980 - i really dont know. Maybe because Michael Brehm, founder of studivZ, is 30 years old (2010-1980). This limit has been found out by simply sending requests as long as possible, waiting for the ban-hammer.

So what do we do to bypass this check? Correct, we simply create 1000 accounts which give us the ability to perform 1.98 million actions ($1000 * 1980$) per day. The average performance my crawler can reach per instance is now 6.7 profiles per second or at this stage 9 actions per second. This results in ~ 578.880 profiles ($6.7 * 60 * 60 * 24$) per day. So we dont even need 1000 accounts, we just need 392! ($578.880 / (1980 / (9 / 6.7))$). For each new instance we need accordingly the same amount more (see below).



But an advice might be to not spam the target system with traffic from a single ip. Just distribute the crawler over several hosts to not reveal the crawler.

The registration-process takes some time, because at this point VZ uses reCaptcha as anti-spam breaker. But within 2 hours one can easily create

160 accounts - 1 account per 45 seconds is easily possible. Using a premade script which automatically fills all necessary fields and let the user just type in the captcha can boost up the time to 20 seconds or less per account, including the email invitations, which result in 360 accounts per hour - all we need.

3.2 THE PROGRAM (USING GROUPS)

Basically the program is split up into 3 files. The first one, "profilebox.php" is used to rip all information off the user profile. It consists of several regular expressions to handle the sourcecode of the profile pages. Further more it consists of several MySQL-Commands to write the data into a MySQL-Database.

The main file is called "studivz.php" or "schuelervz.php", depends on the network we are crawling. This file consists of all necessary functions to perform all necessary steps.

The last file is "accounts.txt" which is beeing used to store all logins in. Each email-address has its own line.

Basically it uses the following functions :

DoLogin()

This function is used to login to any of the VZ-Websites.

GroupPagesTotal(\$groupId)

Gets the count of available pages in a group and calculates the estimate profiles resulting of these pages.

GroupPagesWalk(\$groupId, \$count)

Crawls through each of the pages in a group.

GroupPagesMemberIds(\$groupPagesMarkerIsSource)

Uses regular expressions to get all profiles off a single group page.

MemberPagesWalk(\$id)

Gets all information off a single member profile page.

GroupPagesMemberIdsFlush()

Flushes all member-IDs collected in one group page.

LogoutByForce()

Cleans up any data, removes cookies and logs the current user out.

GetFormkeyiv()

Returns the formkey and iv required as mentioned in 2.1.

GetEmail()

Gets the next email available in the accounts.txt.

As previously mentioned regular expressions can be used to extract every single user entry. For example :

```
$pattern = "/ <a href=\"\"\/Search\/SearchSuper\/platform\/[0-9]\/lookingFor\/[0-9]+\/doSearch\/[0-9]\/rmC\/[0-9]\">(.*?)<\/a>,?/";
```

```
preg_match_all($pattern, $regex, $searchfor, PREG_SET_ORDER);
```

This regular expression returns a String containing what the user is looking for (dates and stuff like that). I've also written another crawler which also uses the friend-pages to collect new profiles. It works like this one with some adjustments in the functions.

4. MANUAL

To successfully run the program following steps need to be performed :

1. Create a file called "accounts.txt" with all login-emails used to login a specific VZ-Website
2. Change the MySQL-User and password in profilebox.php

3. Change the groupid which you want to fetch all profiles off in the mainfile
4. Edit the default password (used over all your crawler-accounts)
5. Create a SQL-Table using the following structure:

StudiVZ :

```
CREATE TABLE `members` (
  `id` varchar(50) character set latin1 collate latin1_general_ci NOT NULL,
  `image` varchar(255) NOT NULL,
  `name` varchar(255) NOT NULL,
  `verzeichnis` varchar(10) NOT NULL,
  `memberjoin` varchar(10) default NULL,
  `memberupdate` varchar(10) default NULL,
  `uniid` int(10) default NULL,
  `uniname` varchar(255) default NULL,
  `regionid` int(10) default NULL,
  `regionname` varchar(255) default NULL,
  `regionexact` varchar(255) default NULL,
  `status` varchar(255) default NULL,
  `gender` varchar(50) default NULL,
  `birthdate` varchar(10) default NULL,
  UNIQUE KEY `image` (`image`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8
```

SchuelerVZ:

```
CREATE TABLE `membersfull` (
  `id` varchar(50) character set latin1 collate latin1_general_ci NOT NULL,
  `image` varchar(255) NOT NULL default "",
  `name` varchar(255) default "",
  `memberjoin` varchar(10) default "",
  `memberupdate` varchar(10) default "",
  `schoolid` int(10) default NULL,
  `schoolname` varchar(255) default "",
  `cityid` int(10) default NULL,
  `cityname` varchar(255) default "",
  `status` varchar(255) default "",
  `jahrgangsstufe` int(2) default NULL,
  `jahrgangsstufeadd` varchar(2) default "",
  `gender` varchar(50) default "",
  `birthdate` varchar(10) default "",
```

```

`iam` varchar(50) default "",
`loveclass` varchar(50) default "",
`hateclass` varchar(50) default "",
`nebenjob` varchar(100) default "",
`whatido` varchar(200) default "",
`msnliveid` varchar(50) default "",
`icq` int(12) default NULL,
`aim` varchar(50) default "",
`skype` varchar(50) default "",
`kontakcity` varchar(300) default "",
`homepage` varchar(200) default "",
`searchfor` varchar(200) default "",
`bezstatus` varchar(50) default "",
`politics` varchar(20) default "",
`hobbys` varchar(250) default "",
`clubs` varchar(250) default "",
`lovemymusic` varchar(250) default "",
`lovelybooks` varchar(250) default "",
`lovemymovies` varchar(250) default "",
`talk` varchar(400) default "",
`whatilike` varchar(400) default "",
`whatidislike` varchar(400) default "",
`aboutmyself` varchar(400) default "",
UNIQUE KEY `image` (`image`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1

```

MeinVZ:

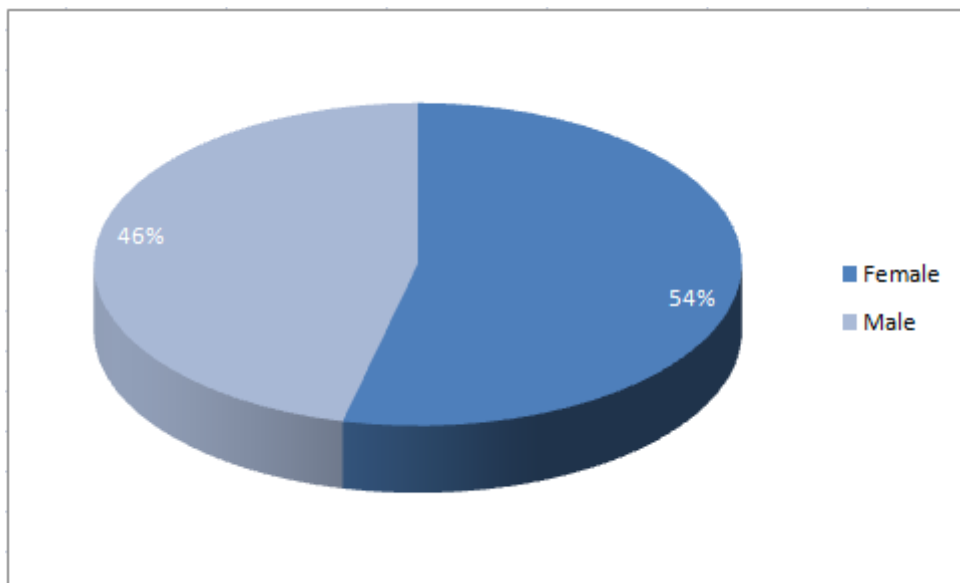
Accordingly to StudiVZ with some small adjustments (no university for example).

Further more one need to create an index over 3 columns as mentioned in 2.1.

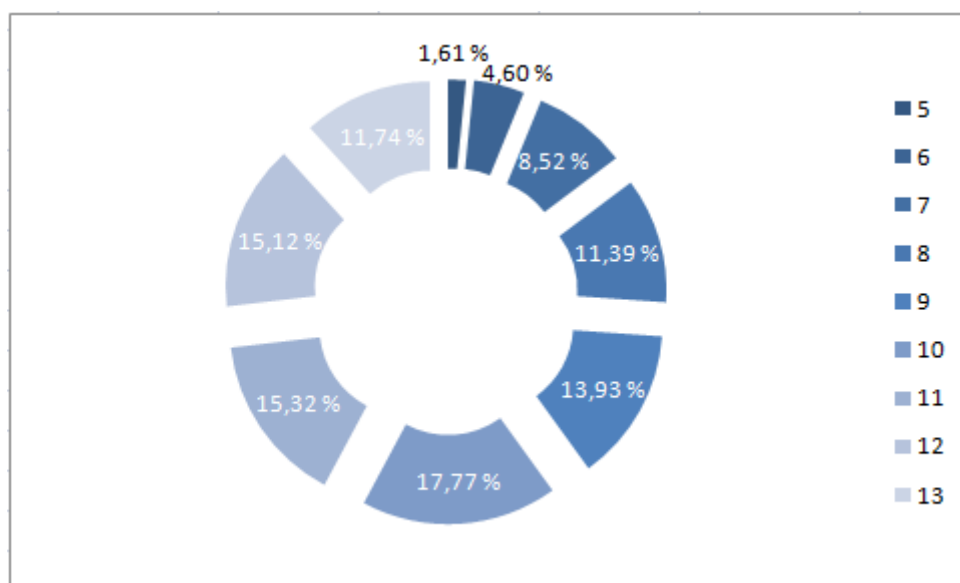
5. SOME STATISTICS

Using the fetched data its fun to generate some interesting statistical breakdowns (using 1.35 million datasets). Ofcourse those statistics base on the fact that the users have made these information visible but using such a huge amount of datasets these statistics are quite representative :

GENDER (%)

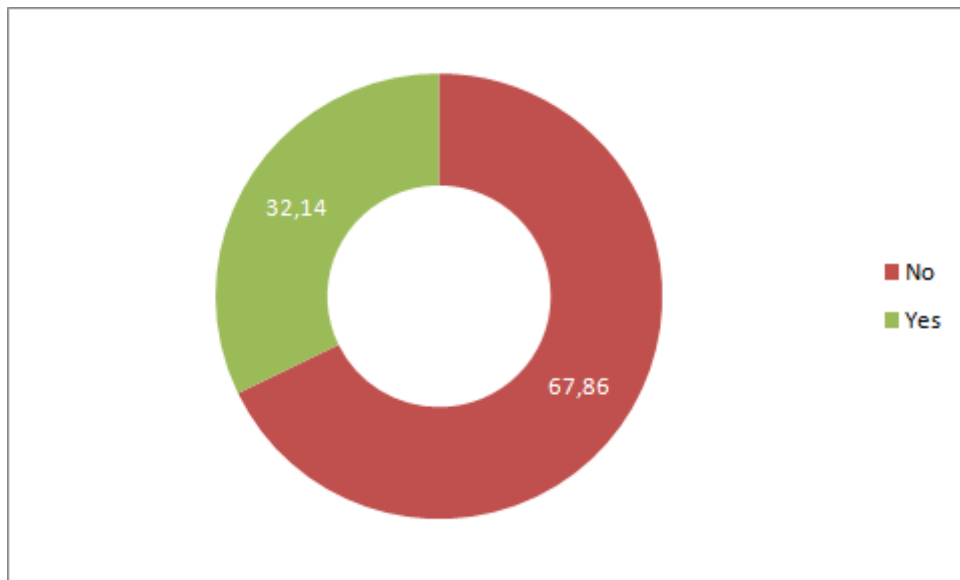


MEMBERS PER GRADE (%)



This is quite interesting. One can think that higher (13th) grade pupils are more clarified about exposing sensible data to social networks - or they're switching to StudiVZ instead.

MEMBERS SHOWING THEIR BIRTHDATE (%)



Obviously i've got several more things to show, but this is an example, i am not here to bore you with these statistics.

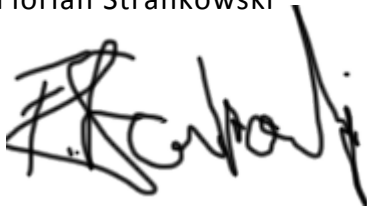
6. CONCLUSION

The VZ-Network urgently needs to work on the implementation of new way of securing users data. The challenge to prevent people ripping off hundret of thousands user profiles can be bypassed just within the time which one needs to create multiple accounts. These accounts are getting reset after 24 hours - every day. So you can reuse them without any further limitations.

One may consider that there is a so called "privacy" function where users can still choose whether they want their sensitive data beeing shared to others or not **but** it is not possible to hide ones school-id and schoolname (same applies for university-id and universityname). Even with this information its possible todo some quite bad things [..].

The german TÜV-SÜD certifies that the VZ-Networks, after a complete audit of studiVZ, meinVZ and schuelerVZ, are data safe and ensure the privacy of their members. I encourage the german TÜV-SÜD to revise their audit, because of the reasons and facts shown above.

Florian Strankowski

A handwritten signature in black ink, appearing to read 'F. Strankowski', written in a cursive style.

Student of Business Informatics
Leuphana University of Lueneburg