

Password reuse, credential stuffing and another billion records in Have I been pwned

05 MAY 2017

The short version: I'm loading over 1 *billion* breached accounts into HIBP. These are from 2 different "combo lists", collections of email addresses and passwords from all sorts of different locations. I've verified their accuracy (including my own record in one of them) and many hundreds of millions of the email addresses are not already in HIBP. Because of the nature of the data coming from different places, if you're in there then treat it as a reminder that your data is out there circulating around and that you need to [go and get yourself a password manager](#) and create strong, *unique* passwords. And before you ask for your password from the data, [read about all the reasons I don't make passwords available via HIBP](#).

Read on for full details...

There's a huge amount of hacked data floating around the web. Of course, you know that by now if you've been reading here or watching what I've been doing with [Have I been pwned](#) (HIBP) and up until writing this blog post, there were 2.7 billion examples of that on the site. There's a lot more there now, but we'll get back to that in a moment.

So there's a lot of stuff getting hacked and a lot of credentials floating around the place, but then what? I mean what do evil-minded people do with all those email addresses and passwords? Among other things, they attempt to break into accounts on totally unrelated websites. Here's a great example: someone grabs the 164 million record LinkedIn data dump that turned up last year and cracks the hashes. They're SHA1 without a salt so the protection on the passwords [is pretty useless](#). In no time at all you've got tens of millions of email address and plain text password pairs. And this is where the real problems begin.

As fallible humans, we reuse passwords. We've all done it at one time or another and whilst I hope that by virtue of you being here reading security stuff you've [got yourself a good password manager](#), we've all got skeletons in our closets (more on mine soon). Most people are just out there YOLO'ing away with the same password or three across all their things. We know that because again, we've all done it and hackers know that because that's their job! As such, they're going to try and break into as many other accounts as they can using the credentials from a data breach. Which brings us to [credential stuffing](#):

“

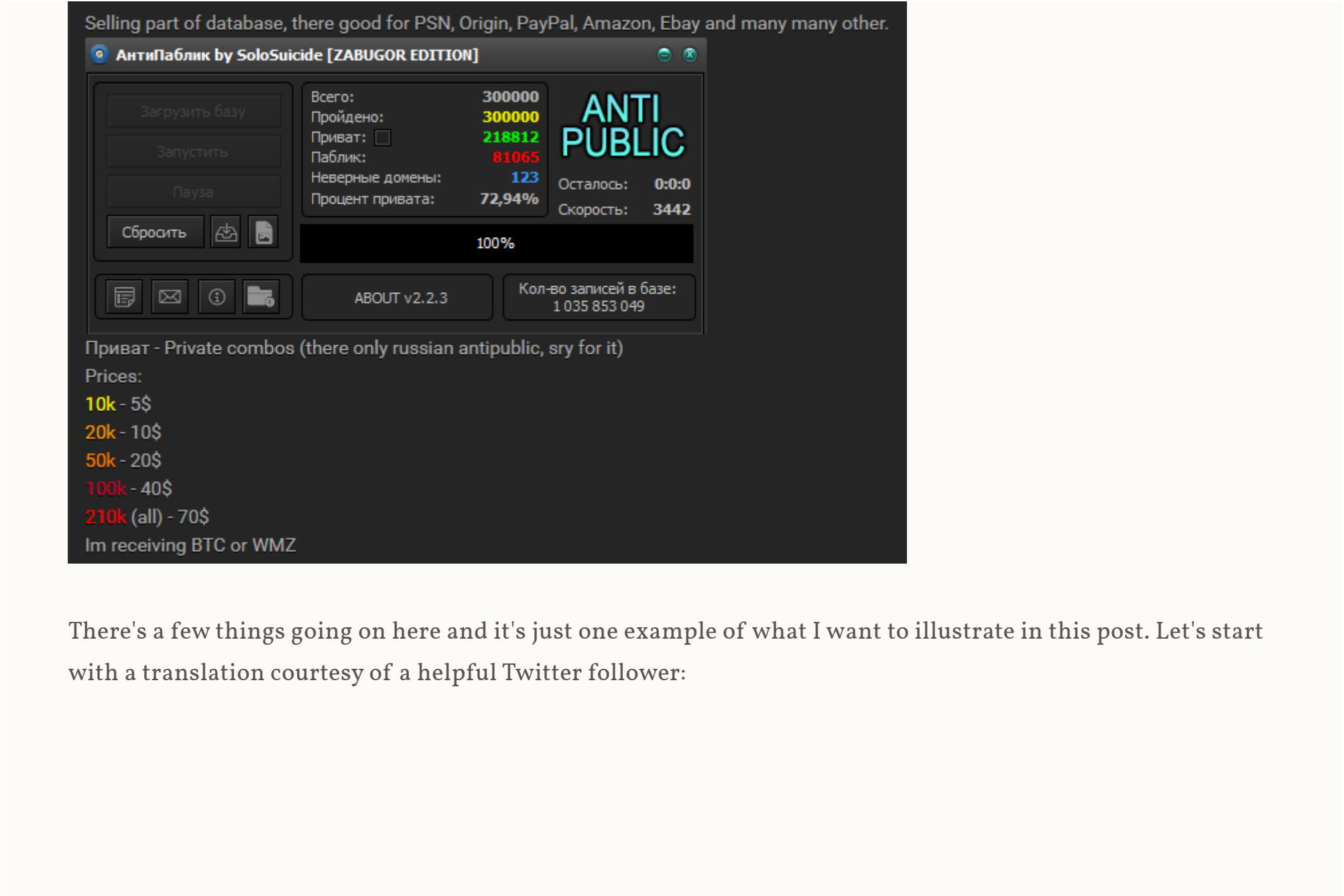
Credential stuffing is the automated injection of breached username/password pairs in order to fraudulently gain access to user accounts. This is a subset of the brute force attack category: large numbers of spilled credentials are automatically entered into websites until they are potentially matched to an existing account, which the attacker can then hijack for their own purposes.

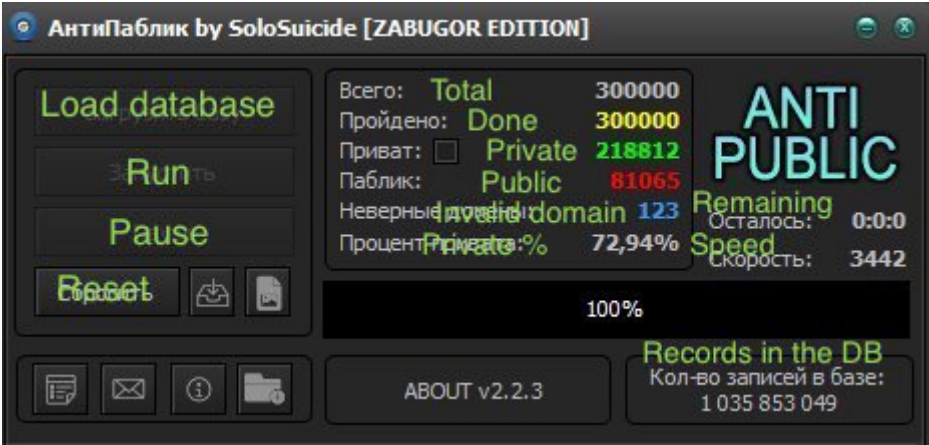
”

This is a serious threat for a number of reasons:

1. It's enormously effective due to the password reuse problem
2. It's hard for organisations to defend against because a successful "attack" is someone logging on with legitimate credentials
3. It's very easily automatable; you simply need software which will reproduce the logon process against a target website
4. There are readily available tools and credential lists that enable anyone to try their hand at credential stuffing

That last point is the impetus for this post because because we have situations like this:





This is the "Anti Public" tool and it's used for verifying the legitimacy of hacked credentials. I got some support from Twitter followers who explained the process as follows:

“

the tool itself is for sale here [redacted]

it's pretty cheap

it's mostly used in Russia, but he does sell an english version

most common use-case: someone buys a dump on x forum, uses the tool to verify which ones are legit

similar to sentryMBA and account hitman
you will often see a uniqueness score associated with the sale based on output

”

Account Hitman is actually a great example because it's in English and there's a handy tutorial showing precisely how it works:

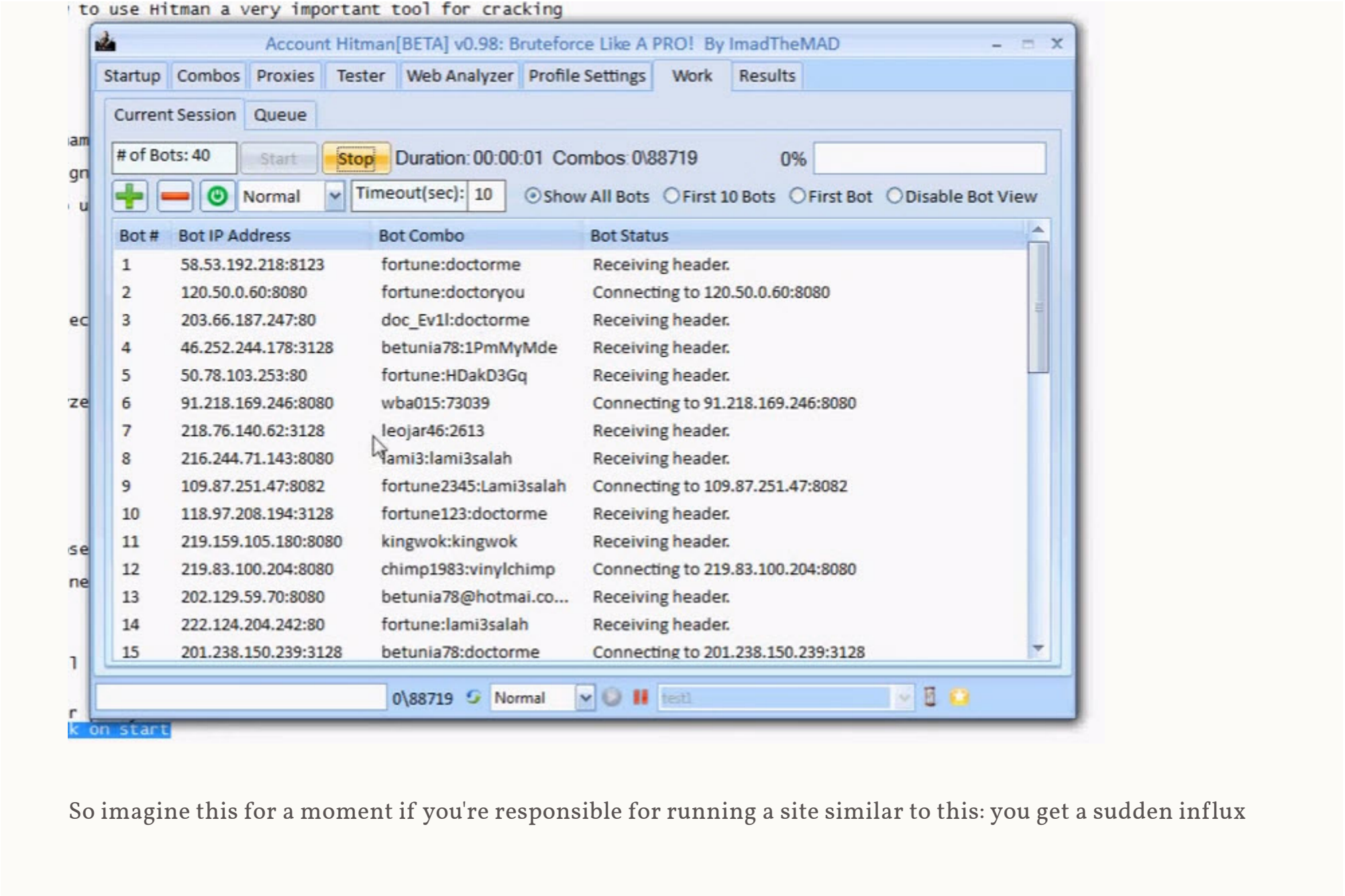


In short:

1. Capture the HTTP request during login
2. Identify the response that indicates a successful login
3. Provide a payload of credentials

4. Fire them at the target site

In this case, Account Hitman also distributes the workload out via a range of proxies which means the target website sees requests coming from a bunch of IP addresses (the term "bots" is used which isn't technically correct as it's not really a distributed workload, rather just routing of the traffic). The final seconds of the video give a good sense of what's going on:



So imagine this for a moment if you're responsible for running a site similar to this: you get a sudden influx

of login requests that match the precise pattern of a legitimate request because they've literally been cloned from one. Valid user agents, valid referrers, a whole range of IP addresses *and* bunch of them actually have legit credentials too because of the reuse problem. There are countermeasures, but you can see the challenge in defending against an attack like this and you can imagine the success rates available to those mounting them.

(Incidentally, the user agent may well be an indicator of malicious requests. Some automated tools have static or default user agents that may be blacklisted. In the example above, the presence of *precisely* the same UA across a mass of requests is a bit of a giveaway. Then again, I've seen attacks against services I've run where the UAs have been very cleverly rotated such that it would be near impossible to reliably blacklist them.)

The prevalence of attacks like this has led to many of the web's larger players being much more proactive in looking for risks associated with credential stuffing. For example, I saw this pop up mere hours before publishing this post:

“

@troyhunt *just received this for an old spotify account*

pic.twitter.com/nry5FbPJPT

— *André Duarte (@atduarte)* May 4, 2017

”

I shared this one from Digital Ocean only a couple of days earlier:

“

Anyone else seen something similar to this from @digitalocean? The recipient is confident their password was unique pic.twitter.com/x4YrOkcYfu

— *Troy Hunt (@troyhunt)* May 2, 2017

”

And they did indeed then confirm that the email address was identified in a data breach. Thousands of companies across the globe use HIBP specifically to identify this sort of risk, some of whom happily share the fact publicly:

“

[@troyhunt](#) [@mattkocaj](#) [@alpha_zero](#) [@haveibeenpwned](#) *We did not in fact*

attempt to verify any leaked passwords. Suspicious behavior and presence of the email in @haveibeenpwned was sufficient.

— MEGA (@MEGAprivacy) April 18, 2017


”


So you can see both why lists such as this are extremely damaging and why HIBP plays an important role in *minimising* that damage.



















One of the terms you'll regularly see represented when looking at tools like Anti Public and Account Hitman is "combos" and indeed you can see a "Bot Combo" column in the image above. You'll also see combos represented on that Anti Public screen where it refers to "private combos" followed by a bunch of prices. There are just username and password pairs and per that image, you can buy them at rates based on a sliding scale; \$5 for 10k of them, \$20 for 50k or go all the way to the whole 210k and get them for the bargain price of \$70. And this brings me to the impetus for writing this post: the prevalence of combo lists on the web.

Over the last few months, I'd been sent a link to this several times:

☐

 Сохранить в Облако

 Скачать

antipublic					10 файлов
<input type="checkbox"/>		1.txt	 966 МБ	23.12.16	
<input type="checkbox"/>		10.txt	 1.94 ГБ	23.12.16	
<input type="checkbox"/>		2.txt	 1.94 ГБ	23.12.16	
<input type="checkbox"/>		3.txt	 1.96 ГБ	23.12.16	
<input type="checkbox"/>		4.txt	 342 МБ	23.12.16	
<input type="checkbox"/>		5.txt	 1.93 ГБ	23.12.16	
<input type="checkbox"/>		6.txt	 1.94 ГБ	23.12.16	
<input type="checkbox"/>		7.txt	 1.95 ГБ	23.12.16	
<input type="checkbox"/>		8.txt	 1.93 ГБ	23.12.16	
<input type="checkbox"/>		9.txt	 1.94 ГБ	23.12.16	

This is about 17GB of combos on a Russian website which appeared back in December and looks like this:

```
abou a.com:cocoaal
abou a.com:cocoa
abou .com:jallyab
abou n:breeze01
abou .com:paradise1
abou @yahoo.com:elisagbi
abou .com:dubycami
abou ahoo.com:midar84
abou @live.com:elisagbi
abou @yahoo.fr:javemymu
abou e.net:gegakyfi
abou @aol.com:elisagbi
```

Just millions and millions and millions of lines of email addresses and passwords. Some email addresses appeared multiple times with different passwords but in total, there were **457,962,538 distinct email addresses** in the trove of data. Problem is, I had no idea where they were from. Nothing in the data was pointing to a clear source; not the domains the addresses were on or the aliases or the passwords themselves. I simply couldn't tie it down and the only name on it whatsoever was the one at the top of the list of files - "antipublic". (See the connection now?)

I filed the data in the "here's a lot of stuff but I've got no idea what it is" folder and moved on. But people *kept*

sending it back to me and the mere prevalence with which it had obviously spread caused me to go back to it earlier this week. If this was a massive trove of credentials and lots of people knew about it, it would be reasonable to assume lots of people *in the data* were probably having a bad time of it.

I grabbed the email addresses and put a temporary copy in HIBP's relational database so I could run some queries. The first thing I found was that more than 130k of the 1.1 million subscribers I have were in the data which is obviously a pretty significant whack. I grabbed a handful of the ones who'd most recently signed up to the service (something I often do during the verification process) and fired them off an email. It was essentially "hey, I've found your data online, can you help me verify if the password on file for you is correct" and as usual, a bunch of intrigued people responded. Now I wouldn't normally email someone their password in this way, but we're talking about data which I now know is very broadly circulating plus, I didn't have anything else with which I could use to verify the data; I didn't know the site it originally came from nor did I have anything like their physical or IP address. But people were willing to help regardless and I got a steady flow of confirmations:

“

It is a variation of a password I have used, yes. Being that is a word in all lowercase, it has to be old, or from very insecure website.

”

“

I can confirm that the first two ([redacted] and [redacted]) would have been used by me at some point.

”

“

One of the two passwords is one I definitely used in the past many times; the other is very close to a similar password but off by one character (that looks a lot like the actual character). So, yes, they're certainly familiar.

”

“

Yes, that is the type of password we would have used in the past. We have

[redacted]!

”

That last redaction is a model of car because unsurprisingly, people regularly choose passwords that have an association to something in their life because hey, that makes them easier to remember!

But I was also curious as to whether people thought it made sense to load this data into HIBP, even though I had no idea where it originally came from. I had my suspicions on the answer because I'd asked a similar thing before introducing spam lists but still, I wanted confirmation. Here's what I got back:

“

I for one would definitely appreciate a heads-up from your site on a breach

”

“

I do think people should know, yes

”

I also want to point out some feedback from one of the subscribers that illustrates the severity of this problem:

“

Back in my teens I wasn't as aware of security issues so would have used that in many different places

”

Because of the prevalence with which people reuse their accounts, ***nobody who confirmed their password could actually tell me where it came from!*** Could have been anywhere. Who knows, it's "their standard password".

What I'm finding as I talk to more and more people about this whole data breach space and the way our information is circulating is that they want to know where they've been exposed. Even if it's unclear where the data originated from, simply knowing that a trace of them has appeared somewhere is important to people. I've seen first hand time and time again how people have used HIBP and seen their email address

returned one or more breached sites and had that "oh shit..." moment where the penny drops that their data has been leaked. ***That changes behaviours.*** People are more likely to practice good password hygiene and take more care of their data on the web once something like this hits them personally.

So the data had plenty of legitimate credentials in it and people wanted to know. Next, I wanted to get a sense of how unique the data was. Is this just an aggregation of individual breaches I already had in HIBP? Or was it genuinely new stuff and loading it would help a whole heap of new people have that "oh shit" moment? So I grabbed 10k random email addresses from the data and queried them against HIBP. ***75.78% of the addresses were already in HIBP.*** Now that's a lot, but each time I load a breach I'm seeing anywhere up to 60% or more of accounts already in the system anyway so based on uniqueness, this isn't entirely different to loading any other incident. And besides, in raw numbers this meant that there were about 111 million *new* email addresses which is a lot more people to influence when they do a search on the site.

So I've now loaded the data into HIBP, titled it "Anti Public Combo List" and flagged it as "unverified", a concept I introduced last year. Now of course Anti Public is the *tool* we saw earlier on rather than the data itself, but it's also the only identifier I had to refer to. The description of the breach then links through to this blog post which obviously gives all the background so at the very least, people can get a sense of the background.

Now, one more thing - there's another list. Truth be told, there are many other lists but like the one discussed

above, this is one that's been circulating quite extensively. And it's bigger. And there's also this:

```
E:\Exploit.in\51.txt:4050459:troyhunt@hotmail.com:
```

Dammit! This is indeed my email address and the password is indeed one I recall carelessly throwing at a bunch of forums back in the noughties somewhere (told you I had my own skeletons). You'll note that the file path references "Exploit.in" which is what this particular combo list is known as. Now I actually knew that particular record of mine was out there because someone had previously sent me this:

Filtered Results from VerticalScope Network (Vbulletin) (939 Websites)	
Hacked on:	2016-02-01
Encryption method for passwords: Vbulletin	
username:	troyhunt
email:	troyhunt@hotmail.com
ipaddress:	
Real_Password:	Reveal this result here
hash:	3e6fcf4bdce4913d8c119
salt:	G1z
Website:	Gtr.co.uk

This is from my blog post about [Thoughts on the LeakedSource take down](#) from earlier this year. LeakedSource was a data breach service that sold your stolen credentials to whoever would pay for them and in the least surprising news ever, they got taken down. Or raided. Or pulled the pin or who really knows (although [Brian Krebs has had a good dig into it](#)), point is they're now cactus. (Pro tip: if you're one of the folks running similar services right now, chances are you're heading towards a Krebs or three letter acronym'd department takedown.)

VerticalScope is one of those breaches that never turned up in the usual trading circles. Word at the time was that LeakedSource had paid some number of thousands of dollars for the data (remember, there was an ROI for them in paying for data because they could sell people's credentials) and that there was a proviso it was never shared beyond them. Now this is all hearsay and speculation, but the fact remains that this one (and a couple of others) remained suspiciously absent among an otherwise sea of broadly distributed credentials. Yet here was the data, in amongst hundreds of millions of other records.

So getting back to Exploit.in, it's a collection of 111 text files totalling just over 24GB. It's the same deal as Anti Public in that it's just masses of email address and password pairs. By virtue of my own unfortunate inclusion there, I also know that it absolutely, positively contains accurate credentials (I'm sure mine is not the only correct one). Furthermore, it contains data that's not in either the Anti Public list or in HIBP. ***It also has 593,427,119 unique email addresses.*** Crikey. I was conscious that there could be a great deal of crossover between the two lists so I joined them together and found that "only" 222 million of the accounts were common so in other words, 63% of the accounts in Exploit.In were not in Anti Public (I'll know how many were already in breaches in HIBP once I load the data).

So between the two lists that's a total of 1,051,389,657 accounts which means a size increase in HIBP of 39% by record count and brings the service up to 3.75 billion records in total. That's going to take a sizeable bit of cloud and speaking of which...

I'm burning through a few hundred dollars worth of Microsoft's Azure cloud bits over a couple of days to get

all of this loaded so if you find yourself in this data and would like to shout me a coffee or some beer, [I've got just the page for you](#). For the technically inclined, that cash goes into running a P2 relational database with 10 simultaneous instances of an S3 app service pushing about 2 billion transactions into Table Storage then using an A7 VM to process all the notifications. There's [a very pretty picture of how this all ties together](#) if you'd like to know more.

There are now 457,962,538 email addresses from the Anti Public Combo List [searchable within HIBP](#). The remaining 593,427,119 addresses from the Exploit.In Combo List should be searchable in about 24 hours and I'll tweet it from [the HIBP Twitter account](#) as soon as it's live. There'll also be a total of several hundred thousand emails sent to subscribers so if you've signed up in the past, there's a good chance you'll get one of those.

Edit: A lot of people are asking for copies of passwords which I can't do [for reasons I've explained in the past](#). If you're using a password manager and have strong, unique passwords across all your accounts then your risk is minimal, good on you! If you're not (and you should be regardless of this incident), go and get one now and start changing the most important ones first. My favorite password manager is [iPassword](#) and it'll take you a few bucks and a couple of hours to make a fundamental difference to your security posture.

Further to this, if you have suspicions as to where the data might have come from, leave a comment below. Many people have emailed me and there's really nothing more I can do re attribution, especially given there

are obviously many, many different sources. Leaving a comment gives visibility to others who read this and will help us collectively better understand what's going on.

SECURITY

HAVE I BEEN PWNED

Troy Hunt

Hi, I'm Troy Hunt, I write this blog, create courses for Pluralsight and am a Microsoft Regional Director and MVP who travels the world speaking at events and training technology professionals ➡

NEXT POST:

Weekly update 33 (sunrise edition)

PREV POST:

Microsoft Flow + Azure Storage + WebJobs + MailChimp + Outlook

