

# Bring back the Honeypots

{marco|haroon|azhar}@thinkst.com

# Introduction



A super quick introduction..

I'm haroon, this is marco (azhar didn't come out) - we work at a company called Thinkst.

..which is a small applied research company with designs on changing the world.

Whats possibly worth mentioning is that we have been doing breaking (of software & into networks) for a loooooong time.

So to steal a phrase.. we are truly talking about defence guided by offence

# The Plan



Really quickly the plan for today then:

- 1) Some brief history on honeypots (we promise not to do the same old stuff);
  - 2) Why they failed;
  - 3) What currently stops honeypot adoption.
- (in doing this - we will hit and share some solutions we have built)



# Some History





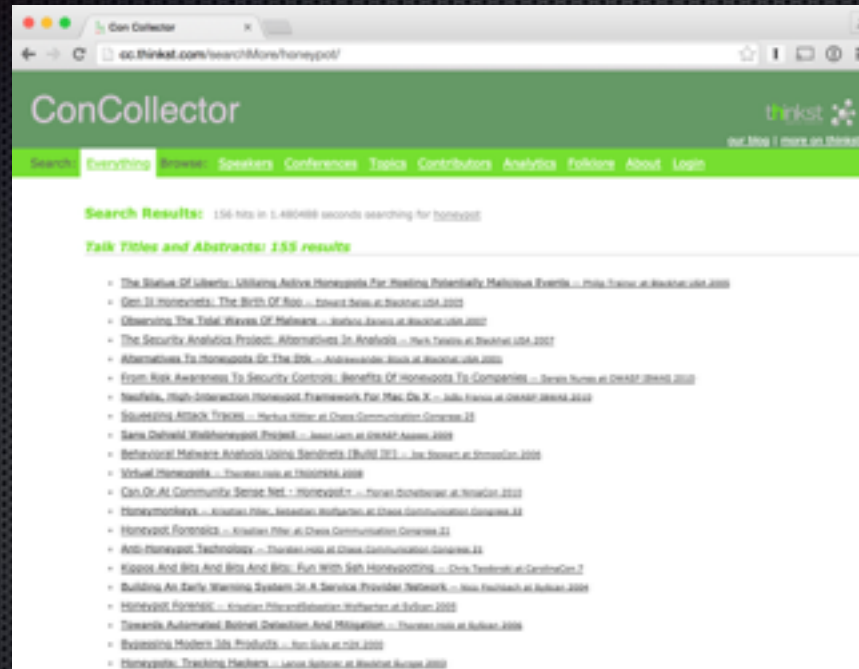
# Some (distant) History

- This isn't new?
- JP 3-13.4
- 1989 | 1991
- The Honeynet Project



The quick obvious thing is that this topic isn't new

# This isn't new



<http://cc.thinkst.com/searchMore/honeypot/>



Our Conference Collector has 155 talks on “honeypots”

Deception in battle is as old as it gets.. and possibly dates back as far as we go

<Abel> What't that in your hand?

<Cain> Nothing.. what hand?



Probably since just after the garden!



# Some (distant) History

- This isn't new?
- JP 3-13.4
- 1989 | 1991
- The Honeynet Project



In the military space, if you are talking “Deception Operations”, you will bump into Chief of staff Joint Publication 3-13.4

# Some (distant) History

- ✦ This isn't new?
- ✦ JP 3-13.4
- ✦ 1989 | 1991
- ✦ The Honeynet Project

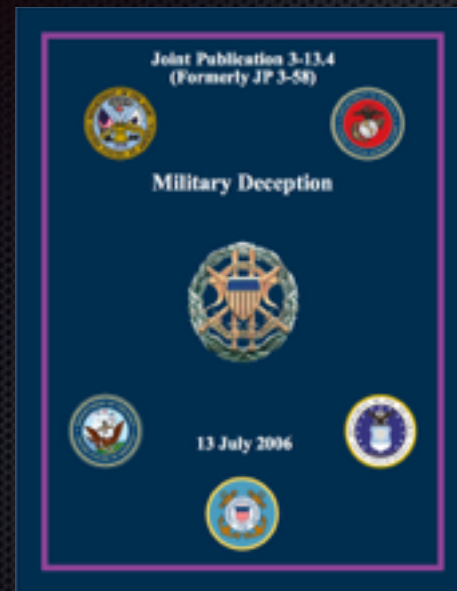


[http://jfsc.ndu.edu/Portals/72/Documents/JC2IOS/Additional\\_Reading/1C3-JP\\_3-13-4\\_MILDEC.pdf](http://jfsc.ndu.edu/Portals/72/Documents/JC2IOS/Additional_Reading/1C3-JP_3-13-4_MILDEC.pdf)



Whats worth noting more than anything is that deception in warfare is well studied, documented and understood..

# Some (distant) History



## 1. Policy

The Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 3211.01C, *Joint Policy for Military Deception*, provides joint policy guidance for military deception (MILDEC). Refer to that document for information concerning responsibilities relating to MILDEC and for specific guidance and restrictions relating to MILDECs planned and conducted in support of joint operations.

## 2. Definition


MILDEC is defined as those actions executed to deliberately mislead adversary decision makers as to friendly military capabilities, intentions, and operations, thereby causing the adversary to take specific actions (or inactions) that will contribute to the accomplishment of the friendly mission.

[http://jfsc.ndu.edu/Portals/72/Documents/JC2IOS/Additional\\_Reading/1C3-JP\\_3-13-4\\_MILDEC.pdf](http://jfsc.ndu.edu/Portals/72/Documents/JC2IOS/Additional_Reading/1C3-JP_3-13-4_MILDEC.pdf)



- JP 3-13.4 : prepared under the direction of the Chairman of the Joint Chiefs of Staff, provides joint doctrine for the planning and execution of military deception
- MILDEC defined as those actions executed to deliberately mislead adversary decision makers as to friendly military capabilities, intentions and operations thereby causing the adversary to take specific actions (or inactions) that will contribute to the accomplishment of the friendly mission.





**Deception for the Cyber Defender:  
To Err is Human; to Deceive, Divine**

Tom Cross, Drawbridge Networks  
Dave Raymond, West Point  
Greg Conti, West Point

[http://www.rumint.org/gregconti/publications/201501\\_ShmoosCon\\_Deception\\_Final.pptx](http://www.rumint.org/gregconti/publications/201501_ShmoosCon_Deception_Final.pptx)

If you like: tom cross, et al did a nice ppt on this

[http://www.rumint.org/gregconti/publications/201501\\_ShmoosCon\\_Deception\\_Final.pptx](http://www.rumint.org/gregconti/publications/201501_ShmoosCon_Deception_Final.pptx)

# Some (distant) History

- This isn't new?
- JP 3-13.4
- 1989 | 1991
- The HoneyNet Project



Then, in 89 and 91, 2 seminal stories brought honeypots into popular culture:

Bill Cheswicks: An Evening with Berferd (In Which a Cracker is Lured, Endured, and Studied)

And

The Cuckoos Egg (by cliff stoll)

**An Evening with Berferd  
In Which a Cracker is Lured, Endured, and Studied**

**Bill Cheswick  
AT&T Bell Laboratories**

**Abstract**

On 7 January 1990 a cracker, believing he had discovered the famous sendmail (BERKEG) hole in our Internet gateway machine, attempted to obtain a copy of our password file. I sent him one. For several months we led this cracker on a merry chase in order to trace his location and learn his techniques. This paper is a chronicle of the cracker's "successes" and disappointments, the bait and traps used to lure and detect him, and the almost "kill" we finally made his activities. We concluded that our cracker had a lot of time and persistence, and a good list of security holes to use once he obtained a login on a machine. With these holes he could often subvert the user and his accounts in short order, and then use. Our cracker was interested in military targets and new machines to help locate his connections.

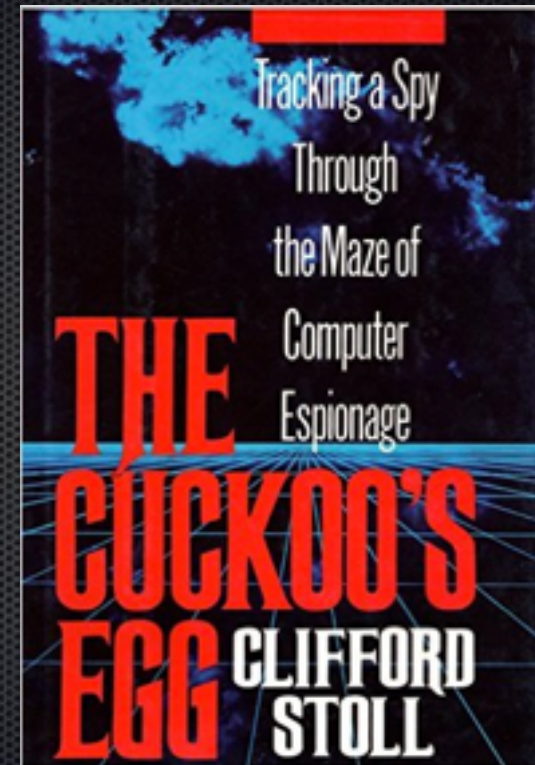
**1. Introduction**

Our secure Internet gateway was finally in place by the spring of 1989/90. With the castle gate in place, I wondered how often the lock was tried. I knew there were hackers out there. Who were they? Where did they attack from and how often? What security holes did they try? They weren't doing any damage to A&S, mostly talking with the door. The ultimate they would be to have a cracker into a situation where we log his sessions, learn a thing or two, and warn his subsequent targets.

The owner of an average workstation on the Internet has five tools for answering these questions. Commercial systems detect and report some probes, but ignore many others. Our gateway was producing 10 megabytes of detailed logs each day for the standard services. How often were people trying to use the services we did not support?

We added a few false services, and I wrote a script to scan the logs daily. This list of services and other facts has grown—we now check the following:

- **FTP:** The scanner produces a report of all login names that were attempted. It also reports the use of a little to possible probe of an old FTP bug, all attempts to obtain FTP's /etc/passwd and /etc/group files, and a list of all files found in the gate directory. People who obtain the password file are often looking for account names to try, and password entries to crack. Sometimes system administrators get their real password file in the FTP directory. We have a bogus file whose passwords, when cracked, are why are you wasting your time.
- **Schwerlog:** All login attempts are logged and reviewed daily. It is easy to spot when someone is trying many accounts, or hammering on a particular account. Since there are no authorized accounts for Internet users on our gateway other than guest, it is easy to pick out probes.
- **Guest/visitor accounts:** A public computer account is the first thing a cracker looks for. These accounts provide friendly, easy access to nearly every file in the machine, including the password file. The cracker can also get a list of hosts known by this machine from the /etc/hosts.equiv file and various personal /etc/hosts files. The login script for these accounts look something like this:



<http://csrc.nist.gov/publications/secpubs/berferd.pdf>







“I was wondering how much effort this was worth.....  
It was fun to lead this guy on, but whats the goal?”

“Though the Jail was an interesting and educational exercise, it was not worth the effort. It is too hard to get it right, and never quite secure.”



Cheswick worked with Wietse Venema , Shimomura, etc..

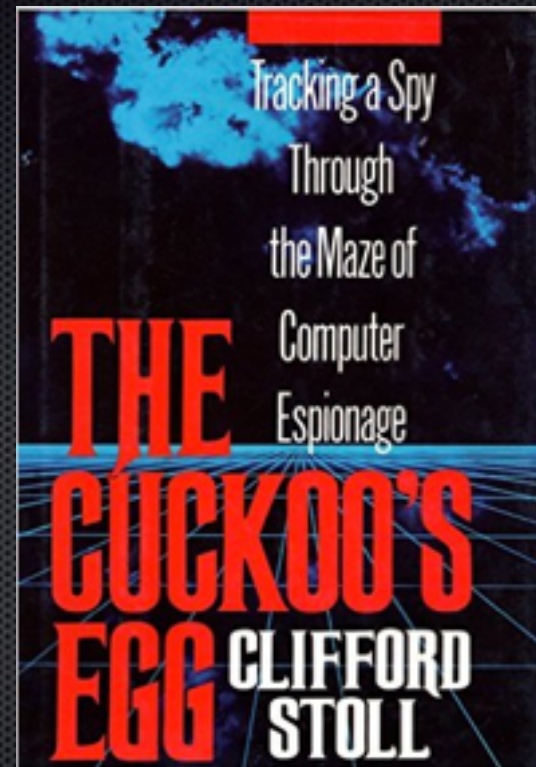
And it was fascinating, but in the end gets to the question:

“was this worth it?”

He wrote that largely, it wasn't.

Although personally, i think a bunch of owned Stanford boxes were discovered to be owned only known only through this..

I felt it was already useful then, and is even more useful now..

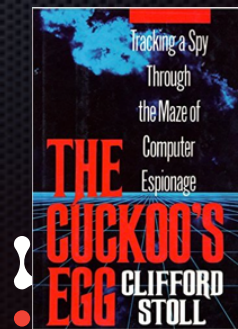


The Cuckoos Egg (by Cliff Stoll)

This was interesting for a whole bunch of other reasons.. (written in .. 89)

It was the intro for many people in info-sec, and the book has held up really well.. (so its worth a re-read if you have not lately)

HOW DO YOU SPREAD THE WORD WHEN A COMPUTER HAS A SECURITY HOLE? SOME SAY nothing, fearing that telling people how to mix explosives will encourage them to make bombs. In this book I've explicitly described some of these security problems, realizing that people in black hats are already aware of them.

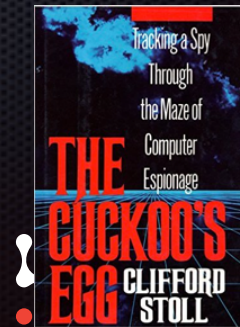


You will find it covers some very familiar themes:  
Disclosure debates; (page 2)



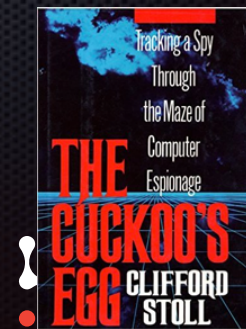
Trapdoor functions are mathematical ratchets: you can turn them forwards, but not backwards. They quickly translate text into ciphers. To make these locks pickproof, it's got to be impossible to reverse the algorithm.

Our trapdoors were built upon the Data Encryption Standard (DES), created by IBM and the National Security Agency. We'd heard rumors that the electronic spooks of NSA weakened the DES. They hobbled it just enough to be crackable by NSA, but kept it strong enough to resist the efforts of ordinary mortals. The grapevine said that this way NSA could crack the code and read messages, but nobody else could.



You will find it covers some very familiar themes:  
NSA weakened crypto; (page 33)

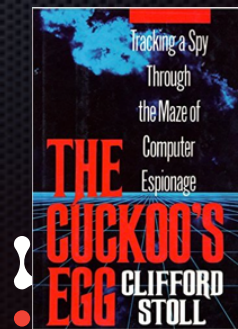
The same thing must be happening all over. The hacker didn't succeed through sophistication. Rather he poked at obvious places, trying to enter through unlocked doors. Persistence, not wizardry, let him through.



You will find it covers some very familiar themes:

Attackers get in due to persistence not wizardry; (84)

How many other security holes were lurking in my system?  
The NCSC might know, but they weren't saying.  
NSA's motto, "Never Say Anything," seemed to come into play. Yet by keeping silent about these computer security problems, they hurt us all. I could see that the hackers had long ago discovered and exploited these holes. Why wasn't someone telling the good guys?  
"It's not our bailiwick," Bob Morris said. "We collect this information so as to better design future computers."  
Somewhere, somehow, something was wrong here. The guys in black hats knew the combinations to our vaults. But the white hats were silent.

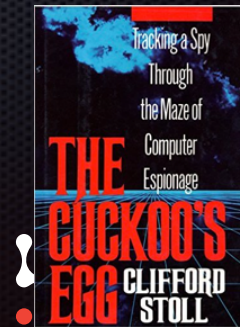


You will find it covers some very familiar themes:

NSA knowing weaknesses and not reporting them; (197)



Yow! Martha had come up with the obvious solution to our problem. Give the guy what he's looking for. Create some files of phony information, laced with bogus secret documents. Leave 'em lying around my computer. The hacker stumbles on them, and then spends a couple hours lapping it up, copying it all.  
Elegant.

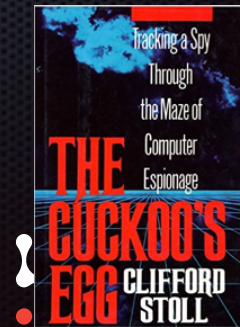


You will find it covers some very familiar themes:

Create fake docs to keep hacker busy; (164)

How do we find 500 pages of fake secrets?; (164)

Yow! Martha had come up with the obvious solution to our problem. Give the guy what he's looking for. Create some files of phony information, laced with bogus secret documents. Leave 'em lying around my computer. The hacker stumbles on them, and then spends a couple hours lapping it up, copying it all.  
Elegant.

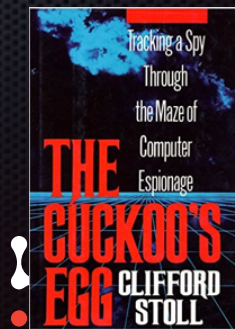


You will find it covers some very familiar themes:

Create fake docs to keep hacker busy; (164)

How do we find 500 pages of fake secrets?; (164)

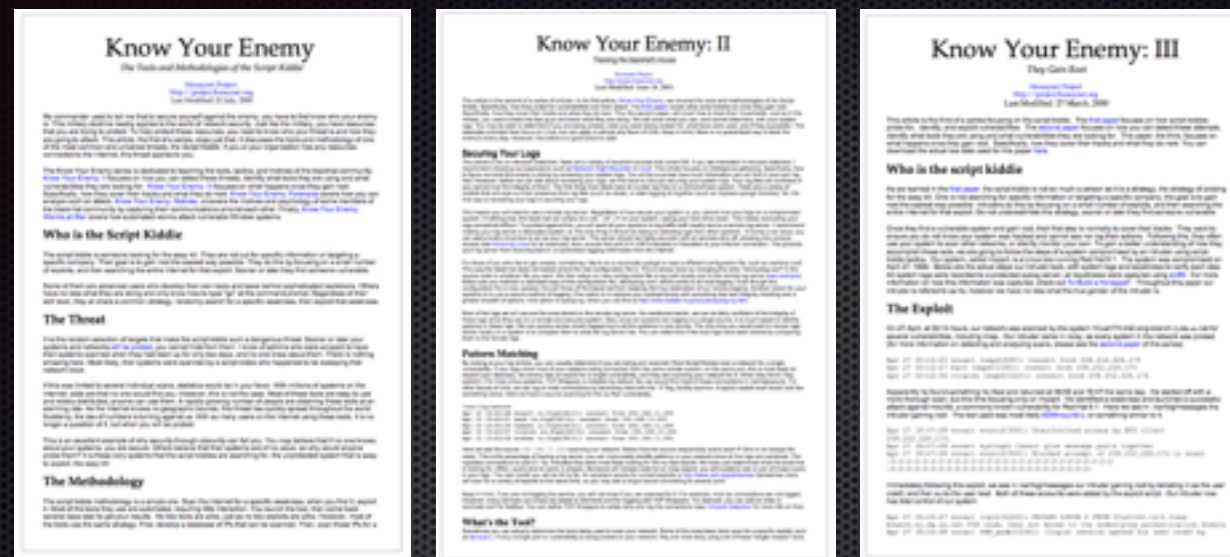
Just to make certain, though, I attached an alarm to those SDI network files. If anyone looked at them—or just caused the computer to try to look at them—I'd find out about it. Right away.



You will find it covers some very familiar themes:

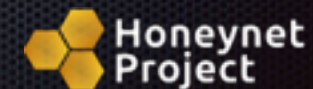
Added an alarm to the sdi files; (166)





“Dedicated to learning the black-hat community’s tools, tactics, and motives and then sharing any lessons learned”

<https://www.honeynet.org/>



Then we had the start of honeypots as we know them..

Fred Cohen introduced everyone to the dtk (the deception toolkit) in 1997

And then we saw the rise of the honey net project..

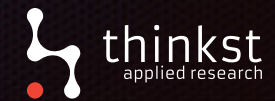
Began in 1999 as a small mailing list of a group of people.

Dubbed itself as the Honeynet Project in June 2000

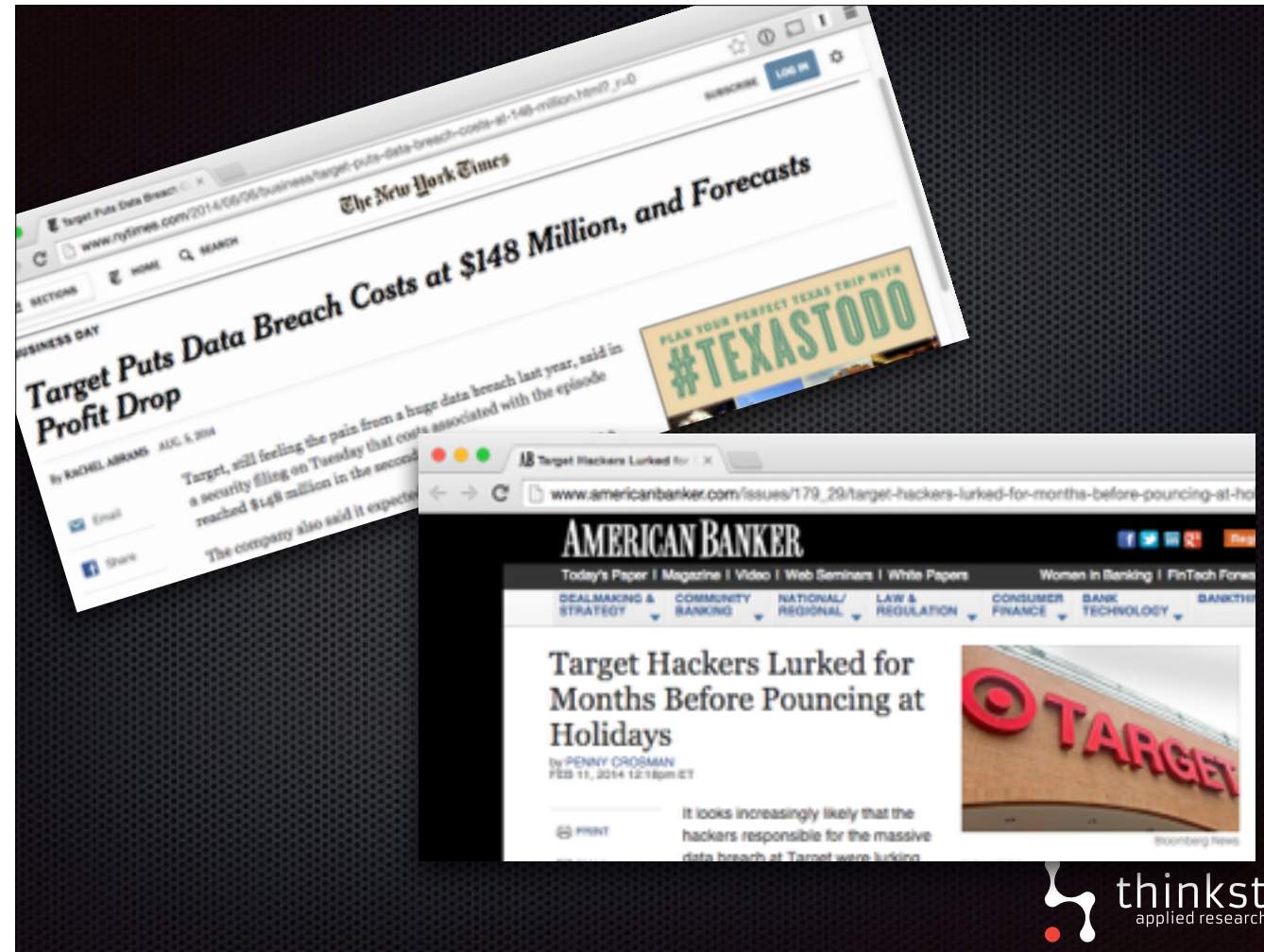
“Dedicated to learning the black-hat community’s tools, tactics, and motives and then sharing any lessons learned..”

This was awesome, and i personally loved the “know your enemy” series back then..

(More recent) History



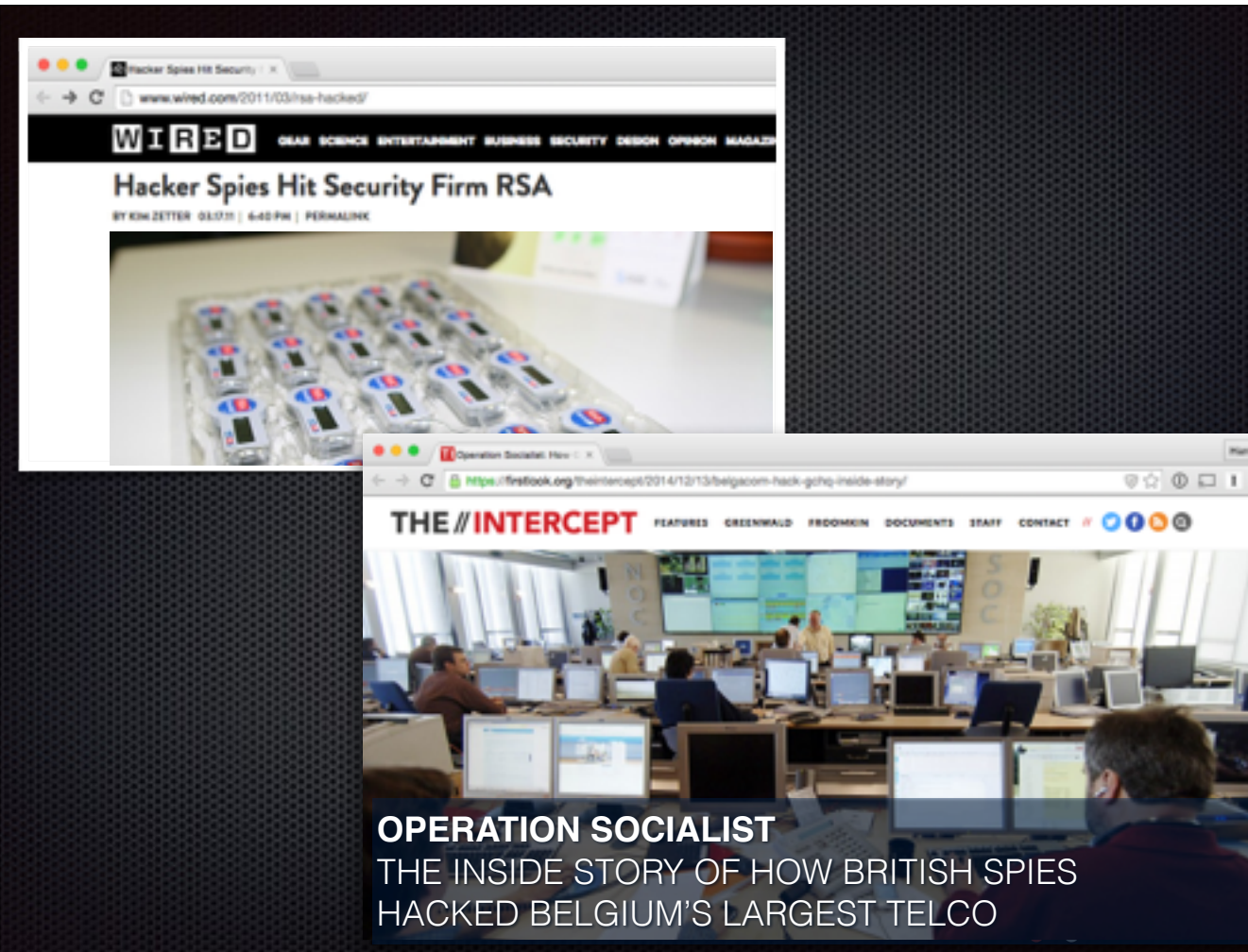
A quick segue then, to more recent history..



We see orgs almost daily getting compromised, and we see massive breaches, but we almost always see that these guys were owned for months or years without knowing it..

Target had a massive security team and their attackers lurked for months before grabbing their loot..





The same holds for most of the recent breaches..

In the case of Belgacom, they were owned by the NSA for years and only found out when the reporters showed up to do a story on it..

## NSA let Snowden crawl all over its intranet unchecked

Snowden grabbed 1.7 m docs using open source tool and scripts, officials say



And the much feared NSA were the same.. Snowden bounced around from Share to Share, and they didn't know they had a problem till he was gone.. OPM is the same.. owned for ages..

# Wat ?



this makes little sense..

- the tech for detection has been around..
- the need is clearly around..
- these orgs all spent millions collectively..

So why have honeypots not made it through to the big time?

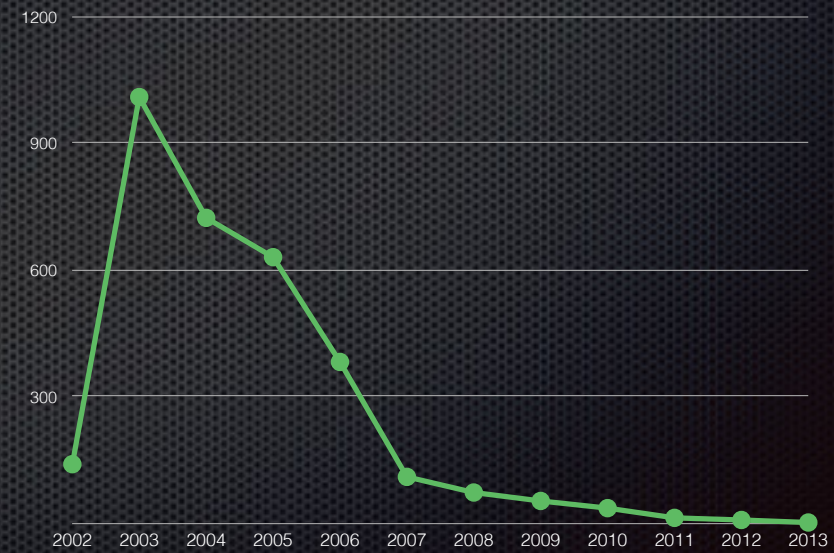


# Why honeypots stumbled



We have to ask.. Why did honeypots stumble / not make it as must have tools?  
To be fair (and to avoid begging the question).. we have to first ask:  
Did they actually stumble?

	Jan-Mar	Apr-Jun	Jul-Sep	Oct-Dec
2013	1	—	—	—
2012	4	2	1	—
2011	8	2	1	1
2010	16	9	8	2
2009	24	19	9	—
2008	15	18	9	30
2007	33	25	28	23
2006	97	135	78	71
2005	213	199	107	110
2004	245	215	116	146
2003	227	379	264	138
2002	—	—	—	139



<http://seclists.org/honeypots/>



We are quite sure they did...

This for example is traffic to the honeypot mailing list over the years..

Peaking in 2003.. and then into the ground..

We believe they are just not living up to their potential and tried to figure out why this is..

# Why honeypots stumbled

- Other technology sells better
- Defence is too busy fighting fires
- Sold badly (Research | HoneyNet Alliance)
- Cliff Stoll
- Bad Rap



One of the things that make honeypots a hard sell, is that they don't "sell" well..

An IDS demos really well.. (drop one in front of a firewall, it lights up and a scared client whips out his cheque book)

AV demos well.. (see infections, see "cleaned", produce cheque book)

but

A good honeypot shouldn't.. It should be silent and should only be screaming occasionally.. so the thing that needs to be silent, ends up being hard to sell because its so silent.

(its also not a preventative control, which people dislike, even though many people are now warming to the fact that complete prevention is impossible)

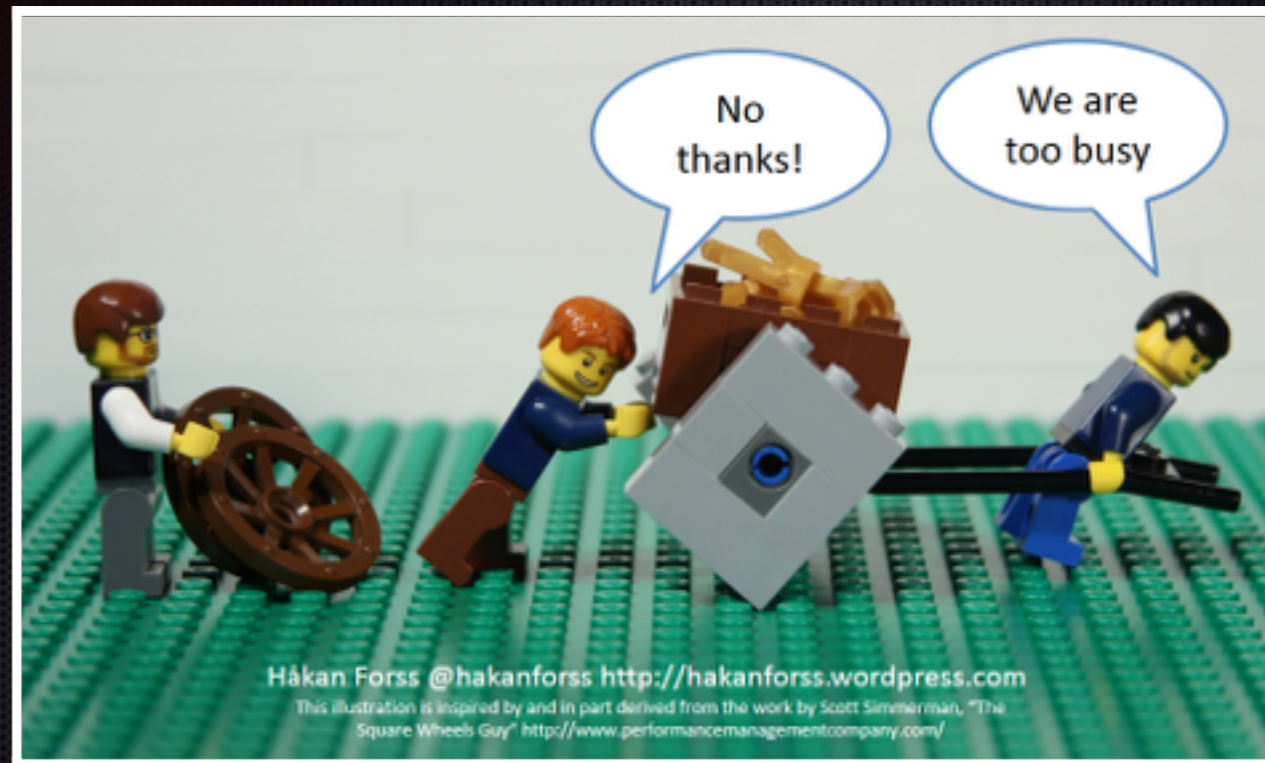


# Why honeypots stumbled

- Other technology sells better
- Defence is too busy fighting fires
- Sold badly (Research | HoneyNet Alliance)
- Cliff Stoll
- Bad Rap.



Honeypots also seemed to have fallen into this: “we are busy now, we will get to it later” niche for security people.



Considering the value we think it brings to organizations, we think this is a little wrong minded..  
In any normal business, theres never going to be a better time.. Especially if aimed right..

# Why honeypots stumbled

- Other technology sells better
- Defence is too busy fighting fires
- Sold badly (Research | HoneyNet Alliance)
- Cliff Stoll
- Bad Rap.



We also think that through nobody in particular's fault, honeypots ended up being pigeonholed as tools of academic research more than tools used in operation.



# Know Your Enemy: Sebek

*A kernel based data capture tool*

The Honeynet Project

<http://www.honeynet.org>

Last Modified: 17 November 2003

## Introduction:

Back at Berkley in the 1980's, Cliff Stoll quickly ran into a problem that still vexes honeypot researchers today: How do you observe an intruder without them noticing? In Cliff's situation he was able to directly monitor serial lines to capture the keystrokes of the intruder<sup>1</sup>. Much has



A quick look at the project Honeynet papers will find that they almost always made it an intellectual pursuit..

“how do we study these attackers?”

You will find honeypots heavily over represented in academic papers and conferences

This isn't horrible by itself, but we believe that it pushed honeypots into a different category of tools.. more suitable to academics and people with “extra time” on their hands.

# Why honeypots stumbled

- Other technology sells better
- Defence is too busy fighting fires
- Sold badly (Research | HoneyNet Alliance)
- Cliff Stoll
- Bad Rap.



Even Cliff Stoll's well documented work from the Cuckoo's Egg, seemed to reinforce this idea of “studying the attacker”.

Back then, it made more sense (because very little was known about attackers) but to a modern network defender with long checklists of things to do, this seems like a luxury..



<https://www.youtube.com/watch?v=pGwqNjLcjuM>



And then, in 2002 this was highlighted when Gobbles took to the stage to talk about “wolves amongst us”





<https://www.youtube.com/watch?v=pGwqNjLcjuM>



They called out the honey net project as being useless, and interestingly, if you see RFP's defense, his characterisation of what HPA was..  
"is just to catch the stupid kids.."

And this allows Gobbles/UT to frame it as "well how many 0days have you guys caught?"  
which made the whole pursuit seem useless..  
All of this came together to make honeypots the red haired stepchild of infosec..

# Why we need them back

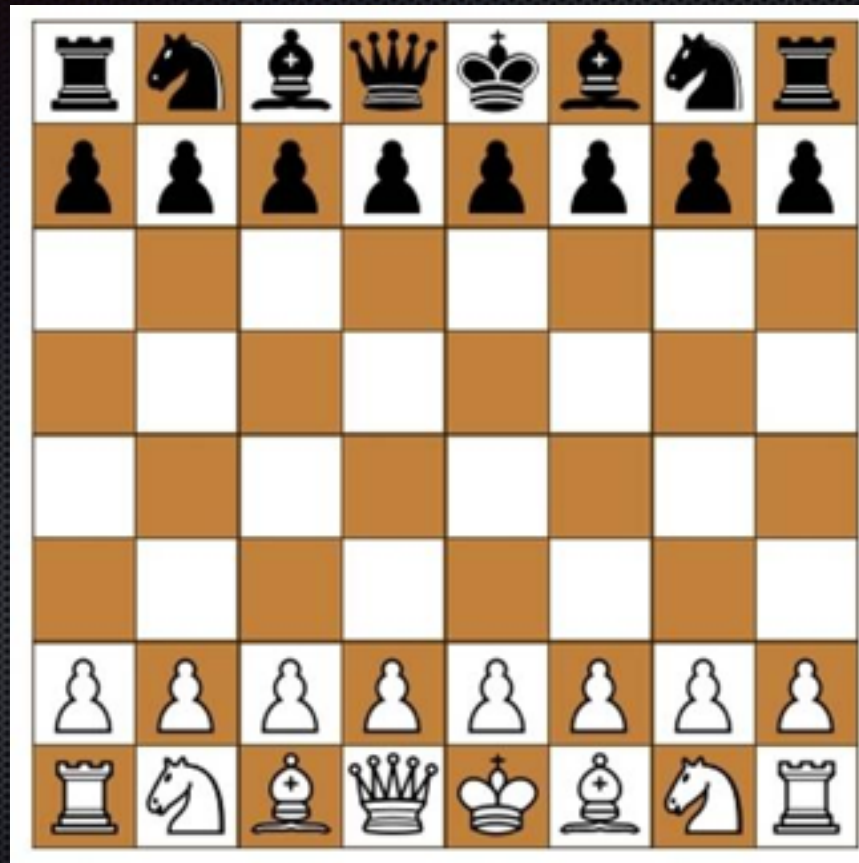


But we believe that we desperately need them back...

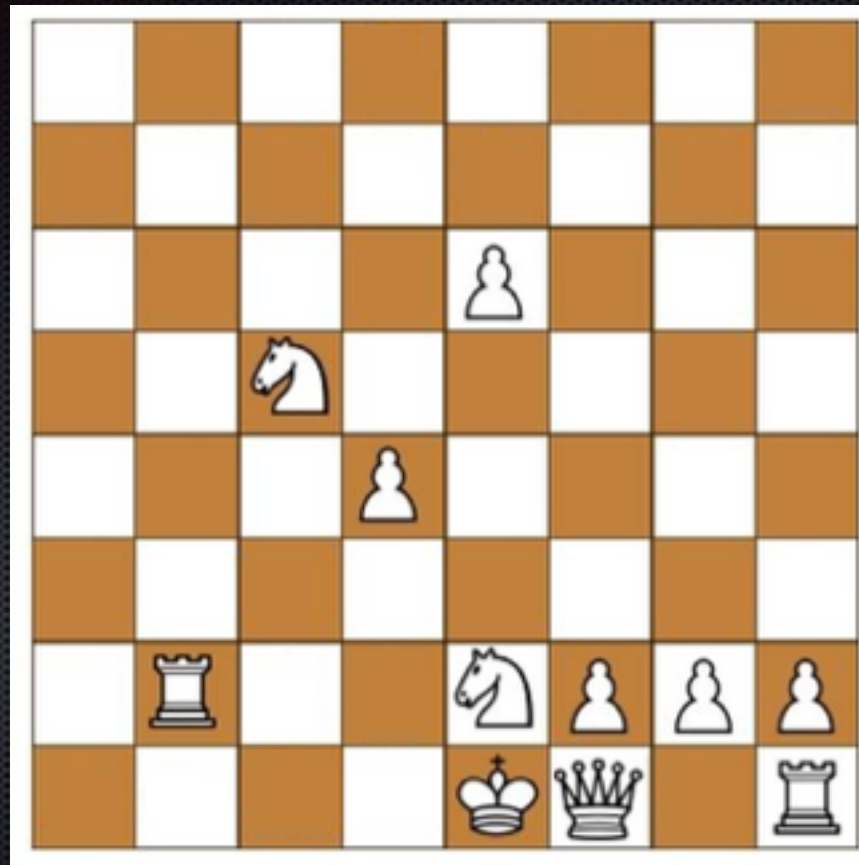


Because we know that you cant avoid being owned - but you can avoid only finding out about it from Brian Krebs

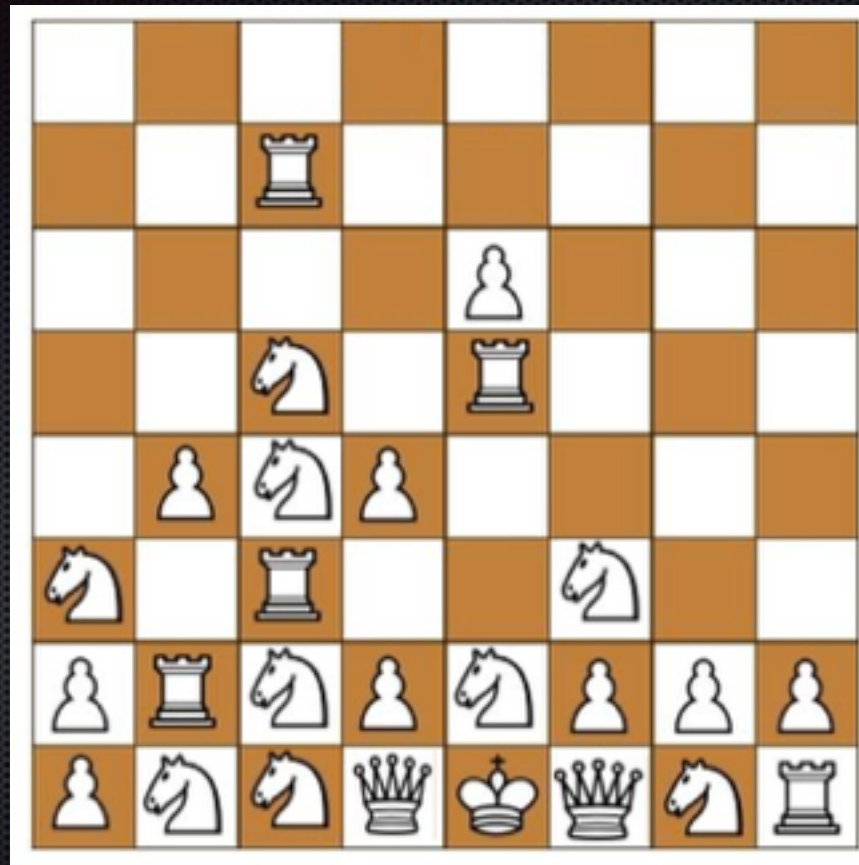




More importantly.. we feel that you are you planning a strategy and trying to play a game of chess



But its a game where your opponent sees the whole board, and you only see your half  
this is a recipe for losing.. and we think honeypots change that game



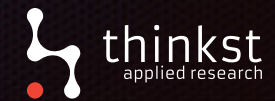
To be clear.. the advantage to be had here, is not that you will suddenly get 100% visibility into your opponents actions and not that you will suddenly see 100% of the board,  
but.. you take away his advantage.. he sees the board, but cant trust what he sees..



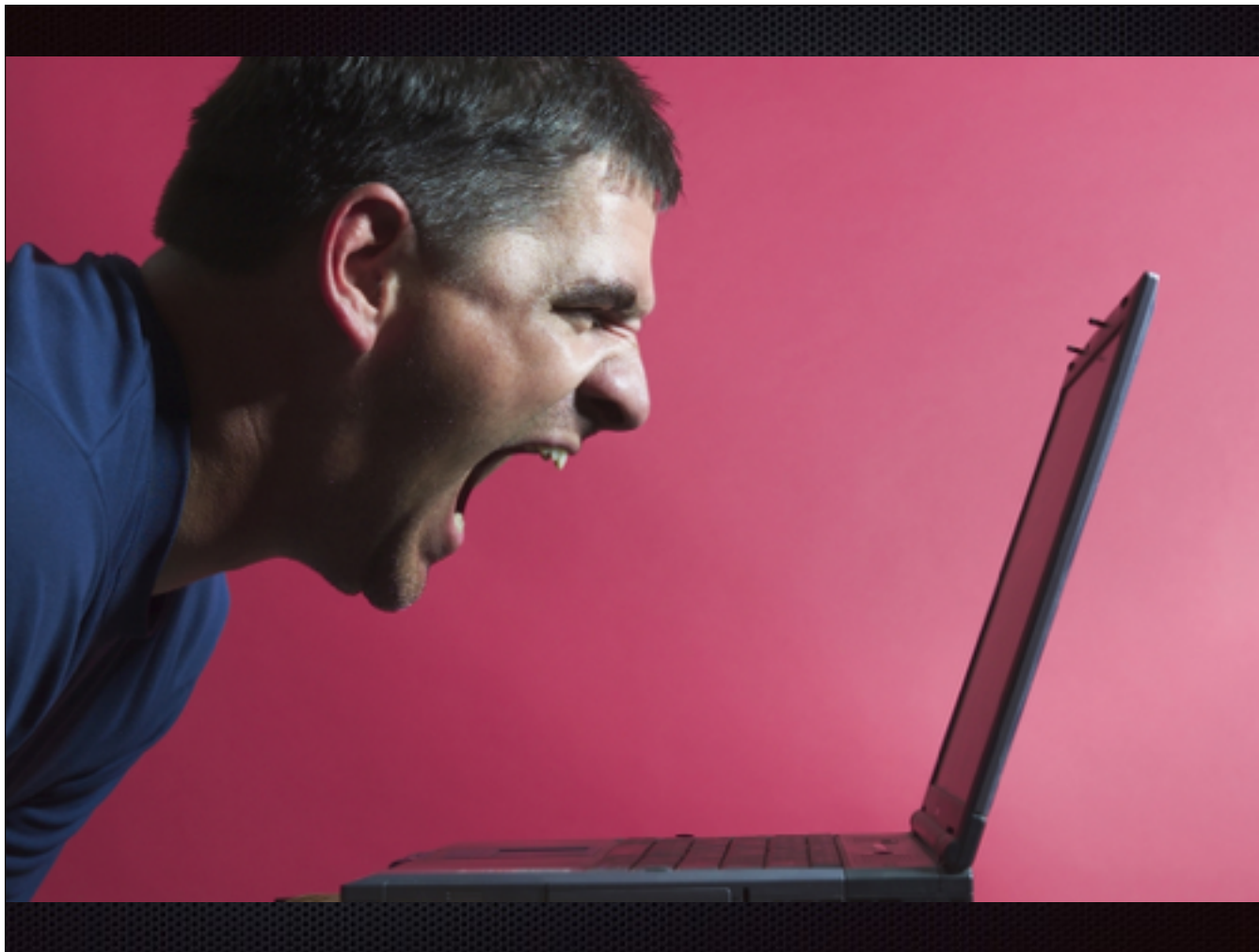


invariably.. what honeypots can allow us to do, is mess with the famed attacker / defender asymmetry  
The common refrain is: “defenders need to defend all the time and attackers needs to win once.. “  
This isn't so.. If an attacker loses he loses TTPs. He loses his ability to operate quietly..

So why are they still rare?



So why aren't we deploying Honeypots all over?



Here are some common arguments against them...



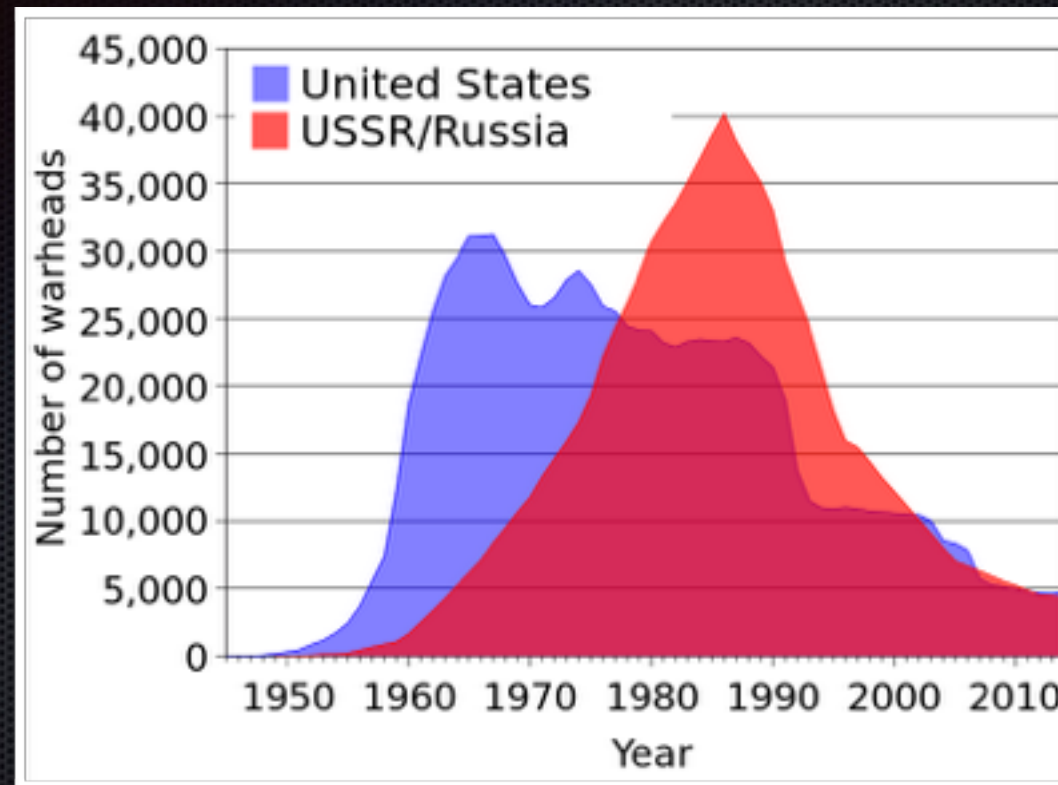
# Arguments (against)

- This is just going to start an arms race
- This introduces new risk to the org.
- They are painful to deploy
- One more device spitting logs that will be ignored
- Will the attacker bump into it?
- I can fingerprint it / discover it / This is useless!
- It is a nice to have, after everything else is done
- This is hobbyist fun, not an enterprise solution
- Not another one!



The first thing you usually hear goes along the lines of:

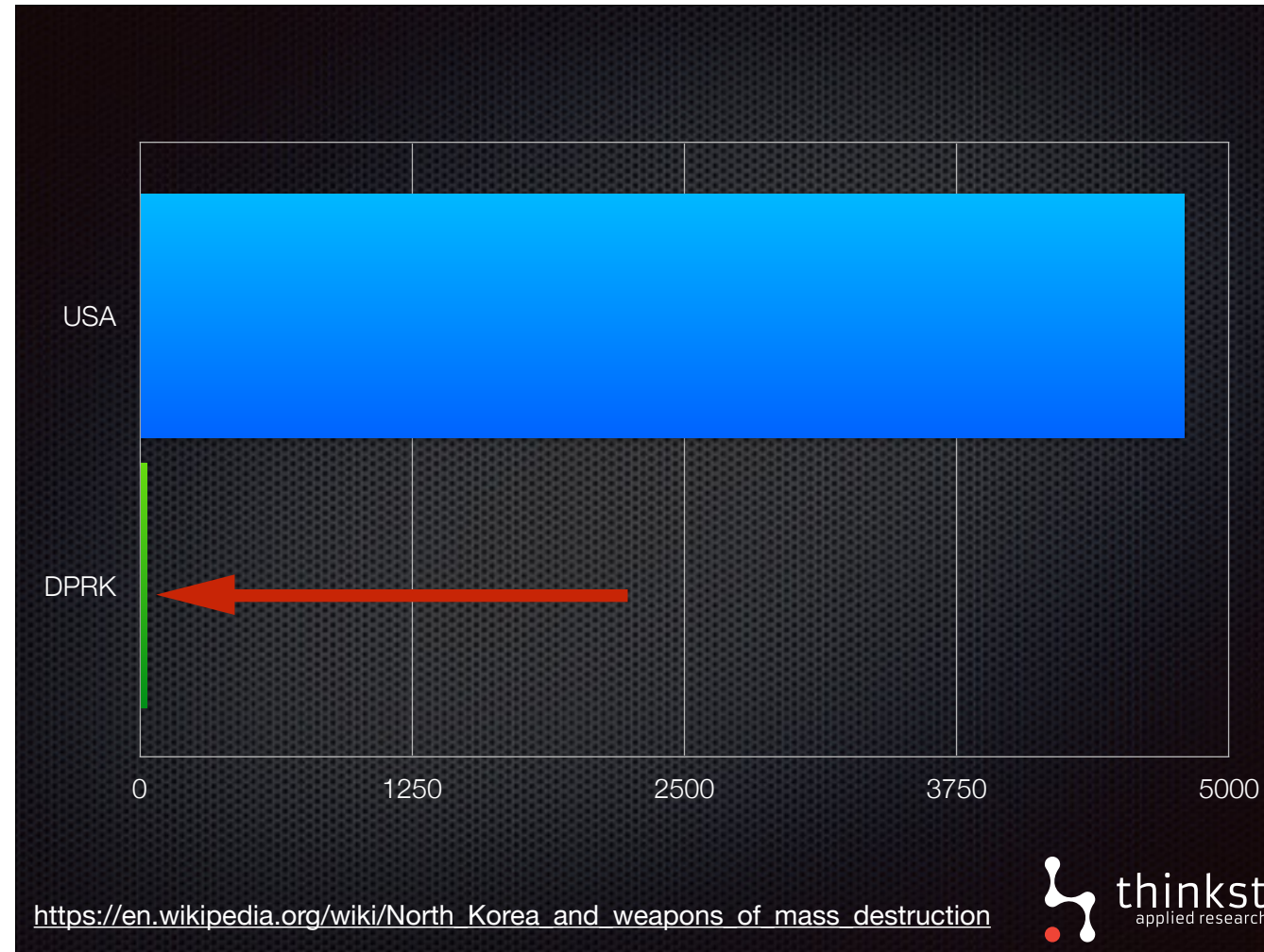
“You are just starting an arms race” - You will detect them, they will detect you, etc..



[https://en.wikipedia.org/wiki/Nuclear\\_arms\\_race](https://en.wikipedia.org/wiki/Nuclear_arms_race)



This is a diagram of the number of nuclear warheads held by USA and USSR.  
We see around 22k and 35k nukes at their peak..  
This is an arms race



This is the graph of warheads between USA & DPRK.. (North Korea has < 30)

This is not an arms race..

Right now, when you honestly eval the state of security on most of our networks, its not an arms race...

Defenders are just plain outgunned.

If it were an arms race.. it would actually be an improvement.



# Arguments (against)

- ✦ This is just going to start an arms race
- ✦ This introduces new risk to the org.
- ✦ They are painful to deploy
- ✦ One more device spitting logs that will be ignored
- ✦ Will the attacker bump into it?
- ✦ I can fingerprint it / discover it / This is useless!
- ✦ It is a nice to have, after everything else is done
- ✦ This is hobbyist fun, not an enterprise solution
- ✦ Not another one!



The other thing that comes up when people are considering it, is:

“Will this honeypot introduce risk to my org”

There are 2 ways i would like to answer it:

# Introduce Risk ?

- Run python on hardened server
- Support only minimal protocols
- “one alert”



1) Technically, We can do a bunch of things to limit the exposure.. so its not bare metal C/C++, we can support a minimal subset of protocols.. and we always subscribe to the “one alert” philosophy.. meaning.. if you have your honeypot, and it can squeeze off its “one alert” saying its being owned, even if its compromised, its done its job.. You are better off than if you didn't have it..

also



but the second way i want to handle this question:  
if we asking about increased risk...



# Introduce Risk ?

## Come on!



I want to scream: “Oh Come on!”

I’ve heard lots of ppl say this initially.. its like a gut reaction..

but we both know full well that they have a subnet with some NT 4 boxes or a range of machines vulnerable to ms08XX running around their networks.

You notice this silly “boardroom utopia vs reality” problem that really hurts us in infosec.. we all find faults for a living.. and so when a new solution pops up.. we spring into action..

We talk about our networks as if they are pristine, and so we slam solutions that would help tremendously (but are not perfect) because we could shoot down the salesmen in the boardroom.

I have seen the same with orgs avoiding EMET, because it can be bypassed.. Sure it can.. but your users are getting owned daily by things that wont bypass EMET..

# Arguments (against)

- This is just going to start an arms race
- This introduces new risk to the org.
- They are painful to deploy
- One more device spitting logs that will be ignored
- Will the attacker bump into it?
- I can fingerprint it / discover it / This is useless!
- It is a nice to have, after everything else is done
- This is hobbyist fun, not an enterprise solution
- Not another one!



Another argument that pops up is that they are more work..  
“who needs more work?”



# Thinkst Canary & OpenCanary

<https://canary.tools/>



And this is something we actively have been working hard to fix..

Our (commercial) Canary product - takes 3 minutes from unboxing, to fully up and running..

and we have much of that goodness in our new openCanary that we are releasing today..

its not crippleware - actually today has more modules in the free OpenCanary than in our commercial Canary



# OpenCanary

- Mixed( Low + High + ?) interaction honeypot
- Python
- Produces high quality signals
- It's a sensor
- Trivial to deploy and update



So lets talk OpenCanary.

People usually like to talk about low, medium or high interaction honeypots, but OpenCanary, like Canary, is a mixed interaction honeypot. It allows you to run some services in low interaction mode (just a fake banner) and it allows for high interaction services like SSH or File Sharing.

Its written in Python, runs everywhere & has reduced exposure to memory corruption exploits.

It's also quick to extend

# Sensor?

- Configure it to
  - mail you on every event (noisy, e.g. a brute-force)
  - output JSON to a TCP connection
  - write to Syslog
  - or any other Python logging option
- opencanary-correlator
  - Example consumer of events that coalesce into incidents.



For the most part, we want our honeypots to act as sensors.

It's a sensor producing signals. You can follow every signal, but often the signals are duplicates or could be related across multiple devices.

The correlator coalesces events into incidents and sends one notification.



# Demo 1 – Installation and MySQL service

We do a quick install of OpenCanary, to show how easy it is to setup and make it useful.

- `pip install opencanary`
- `cd demo1_install/`
- `opencanaryd --start`
- `nmap canary`
- `nmap -sV canary`
- `mysqlclient -u root -h canary -p`
- switch to Gmail to show alerts coming through.



# Quick code peak: SIP

```
15     SIP requests sent its way.
16 """
17
18 class SIPServer(Base):
19     def handle_request(self, request, addr):
20         try:
21             logdata={'HEADERS': request.headers.data}
22             self.transport.getPeer = lambda: IPv4Address('UDP',#
23             self.factory.log(logdata=logdata, transport=self.tr#
24         except Exception as e:
25             self.factory.log(logdata={'ERROR': e}, transport=se#
26
27 class CanarySIP(CanaryService):
28     NAME = 'SIP'
29
```

26,0-1 58%

To show how easy it is to write modules, lets take a look at a SIP module.

This slides shows a SIP reporting module in two classes and 18 lines (minus imports)

Recall: Our aim is not adherence to full protocol spec

We want the service to produce a useful signal.

In most cases that happens really early in the protocol.

# Modules

- ✦ TCP
  - ✦ FTP
  - ✦ SSH
  - ✦ Telnet
  - ✦ SMB
  - ✦ MSSQL
- ✦ MySQL
  - ✦ RDP
  - ✦ Git
  - ✦ VNC
  - ✦ HTTP
  - ✦ HTTP Proxy
- ✦ UDP
  - ✦ SIP
  - ✦ SNMP
  - ✦ TFTP
  - ✦ NTP
  - ✦ Portscanning

Currently, OpenCanary ships with the following modules:  
(RDP and Samba use external code)

# Sample triggers

- HTTP/ SSH / FTP / Telnet / VNC / MySQL / MS-SQL / RDP
  - Login attempts
- NTP
  - 'monlist' command
- SIP
  - Any SIP request
- Samba
  - File read



Each of these is a strong signal *inside* a network. Exposed to the world, you're going to see loads of attempts you can't do anything about, but inside a network, you would want to know if someone was requesting these services from a host they shouldn't..



## Demo 3: All modules.



A quick demo to show:  
Show samba file read,  
Show RDP access,  
Show VNC access.

# Integrated projects

- RDP integration – RDPy
- SSH integration – Kippo
- SMB integration – Samba



Quick shoutout slide for projects we've used

# It's Open Source

<http://opencanary.org>



So if the original question was: “Is it tough to deploy?”

We are saying with Canary its literally less than 3 minutes, and with OpenCanary, its as simple as `pip install`

As of today, you can grab OpenCanary at: <http://opencanary.org>



# Arguments (against)

- This is just going to start an arms race
- This introduces new risk to the org.
- They are painful to deploy
- One more device spitting logs that will be ignored
- Will the attacker bump into it?
- I can fingerprint it / discover it / This is useless!
- It is a nice to have, after everything else is done
- This is hobbyist fun, not an enterprise solution
- Not another one!



Of course a fair argument is that you are creating one more device that will create alerts that will be ignored..

# Sensors are noisy

Because if we want a sensor, Sensors can be noisy..

OpenCanary is able to log to a file, or send a mail on every event, or do whatever python loggers can. But that's pretty noisy.

Can we reduce the noise?



# opencanary-correlator

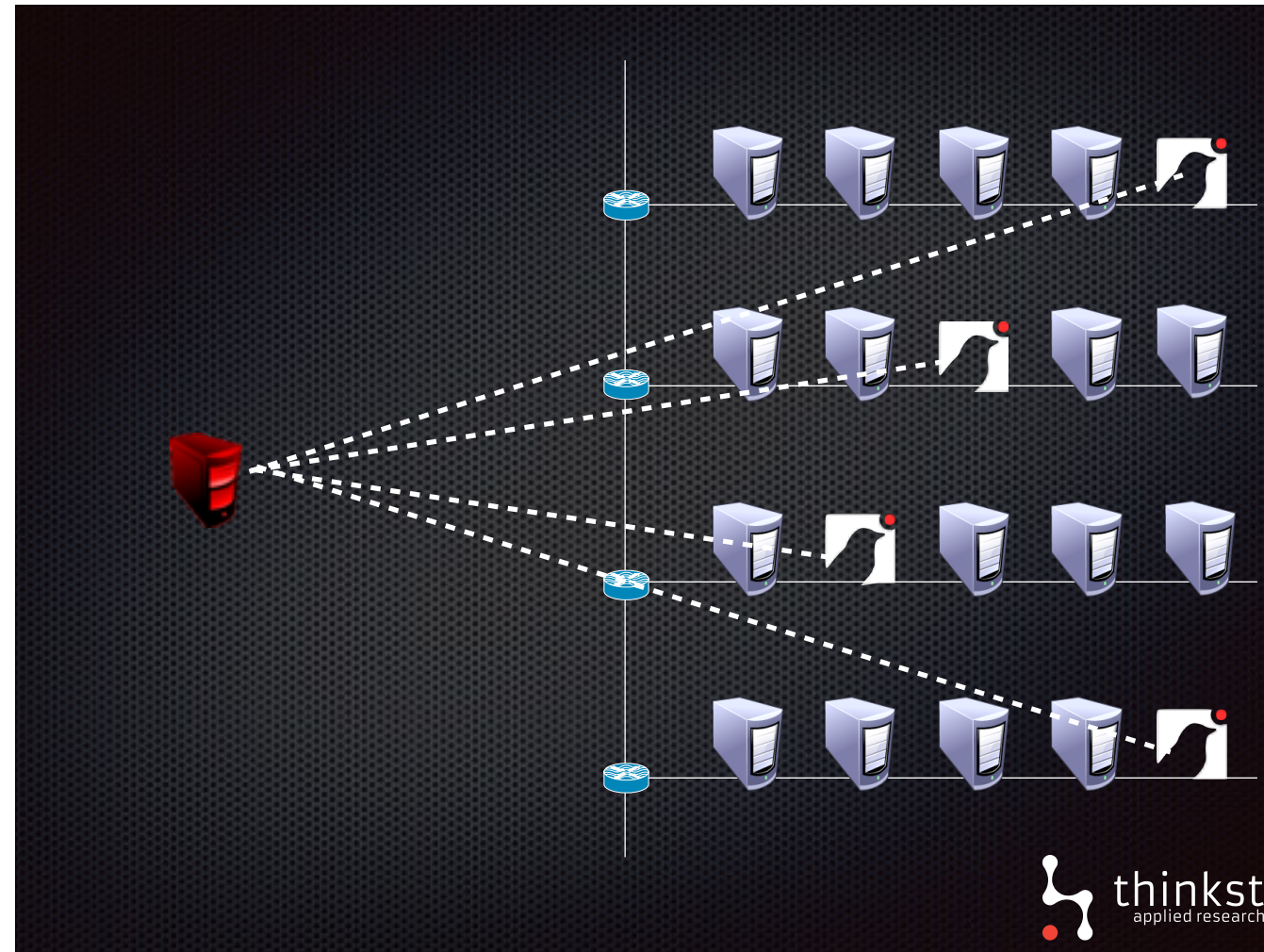
- Correlation engine for multiple opencanaries.
- It processes incoming events, distills incidents.
- Sends one notification per incident. (email or SMS)
- A central correlator has some advantages.



We make use of the Correlator.

So multiple OpenCanary Sensors can send their events to a correlator, which then processes and distills incidents.





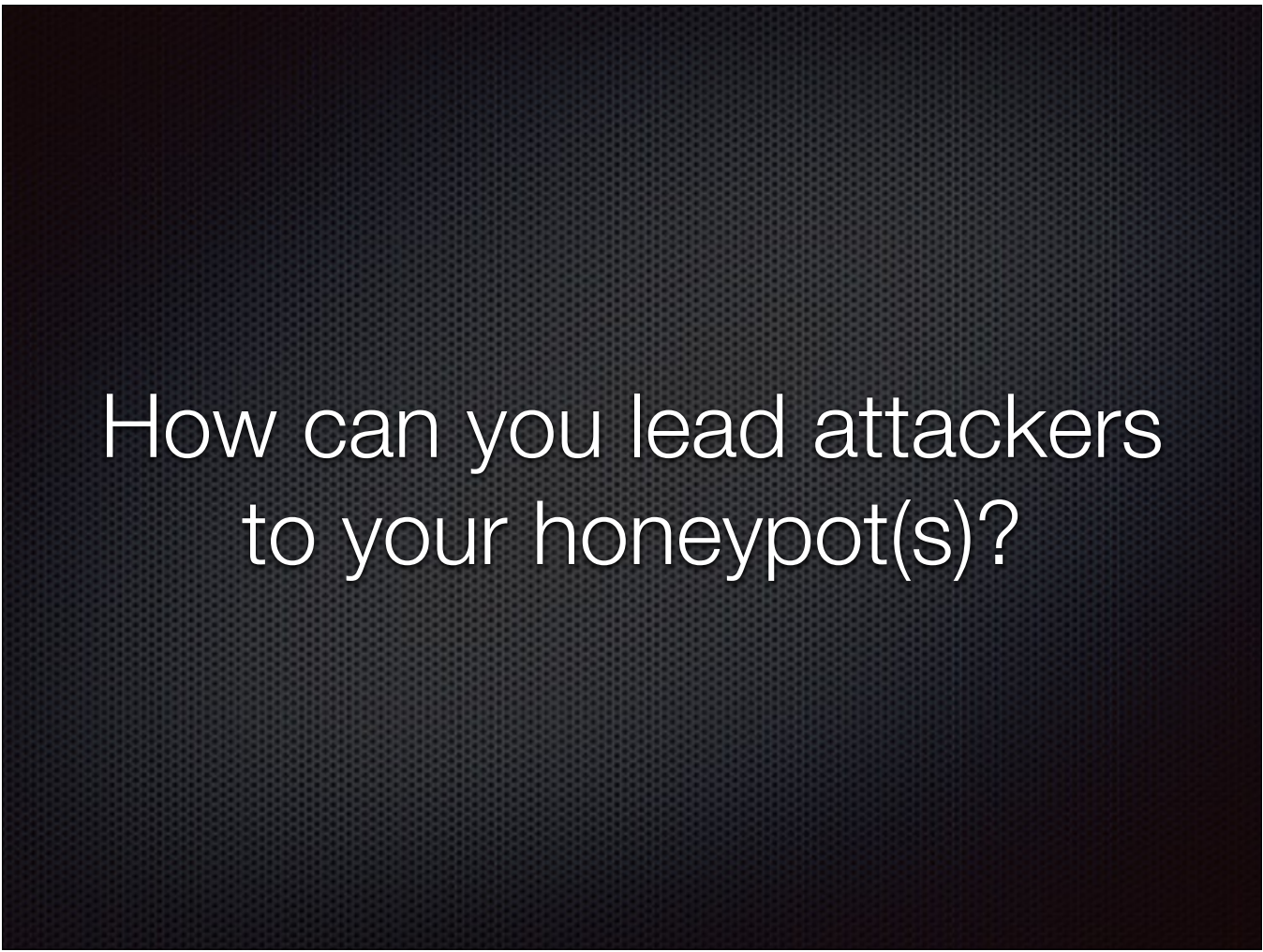
This allows us to detect events like slow scans across multiple canaries..  
Keep in mind though, that we are still not talking about hundreds or even tens of events per day. Largely, we are still looking for our canaries to be “one alert” devices. I.e. when you get an alert from a canary, you have a problem worth chasing.

# Arguments (against)

- ✦ This is just going to start an arms race
- ✦ This introduces new risk to the org.
- ✦ They are painful to deploy
- ✦ One more device spitting logs that will be ignored
- ✦ Will the attacker bump into it?
- ✦ I can fingerprint it / discover it / This is useless!
- ✦ It is a nice to have, after everything else is done
- ✦ This is hobbyist fun, not an enterprise solution
- ✦ Not another one!



The next question then is, How do we make sure that the attacker will bump into our canary?



How can you lead attackers  
to your honeypot(s)?

this can be rephrased simply as: “How can you lead attackers to your honeypot(s)?”

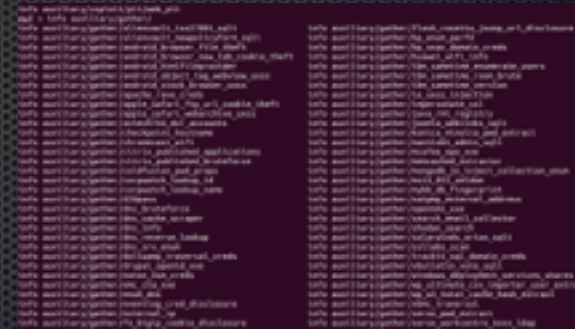
Now we’re getting into heavy deception.

If your honeypots are more attractive or more visible, then they would trip up eager attackers.



# Seeking targets

- ✖ Attackers choose subsets of the full network as targets.
- ✖ Recon techniques are well studied.



Its worth noting that during a real attack:

- Not every host is attacked.
- we use both Active and passive recon.

(Pic is of metasploit modules for info gathering. Still many more scanners.)

# Honeypot discoverability

- Two factors contribute:
  - References to the honeypot
  - Honeypot density
- Doesn't appear to be much prior public work in this space.
- Passive only recon might miss your honeypots if they operate in isolation.

There doesn't appear to be much documented work in the space of honeypot discoverability.

We figure that there are 2 big contributing factors:

- 1) The number of references to the honeypot strewn around;
- 2) The number of honeypots that will be used.



# Discoverability: References

- A reference is some artefact that implies the presence of the honeypot host.
  - host file entries, directory entries (DNS, ARP, MDNS), NFS exports, web.config, shell histories, Pastebin
  - A host that emits traffic is much more likely to be detected by passive techniques. (NetBIOS name announcements, SSDP queries, CDP packets)
  - Misdirected packets
- The more references the better the chance the honeypot is noticed,

It's going to be a balancing act. Too many references and things start to look strange (but that's a potential for misdirection of actual services).

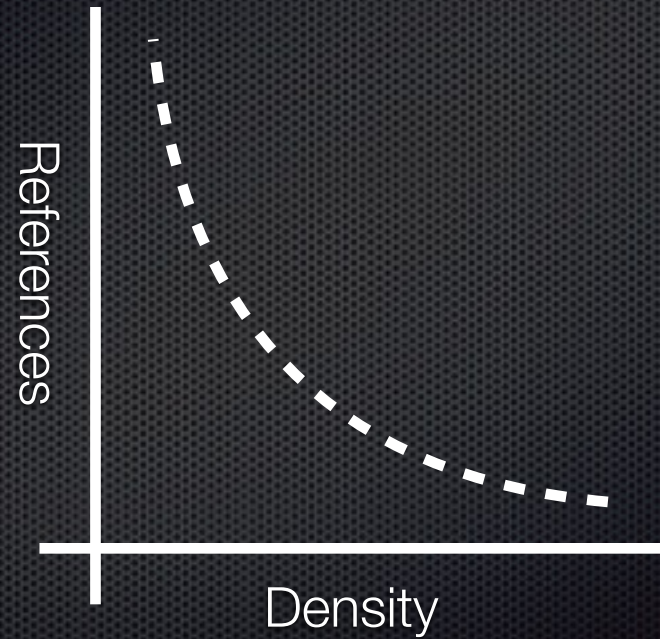


# Discoverability: Density

- In a large network with low numbers of honeypots, the probability that a honeypot will get hit early is low.
- What percent of your hosts should be honeypots?
  - (And density abstracts implementation details like segments)

These are open questions. Density and references are also orthogonal and it seems that they

# Efficient honeypot deployments?



We made up this graph, but it seems about right to us..

That there is an efficiency trade-off between the number of hosts deployed, and the number of references to hosts..

# OpenCanary: Discoverability

- DHCP hostname writes into DNS
- Active Directory listing
- NetBIOS ads
- ARP
- Scope for more
  
- Deployment dependent



Left natively, OpenCanary will do some stuff that helps make it discoverable.  
These are standard network artefacts of a host being up (and possibly some chatty protocols)



We think it can do more..



We think we can help these along..

# Hadededa



- Our attempt at increasing honeypot references to increase discoverability
- Talks an array of noisy network protocols:
  - MDNS, Dropbox LanSync, CDP
- Write into directories:
  - Dynamic updates for DNS
- Injects TCP fragments to each hosts on the local network



A Hadededa. is a noisy South African bird..

It's also now our (experimental) tool that can be used to increase references to our Honeypots.

Essentially Hadededa will inject TCP fragments to hosts on the local network, with fake chatty traffic referencing our honeypots.



# TCP fragments

- Basic idea is on Ethernet, you'll sometimes get packets intended for other hosts.
- If you're an attacker passively sniffing traffic, these packets can provide a clue about what is happening on other hosts.
- So we generate fake TCP/IP fragments between a honeypot and another host.
- Then send that fragment to every MAC listed. Passive listeners see a packet addressed to them at Layer-2 but who's IP addresses belong to other hosts. Works for Ethernet broadcast too.



On Ethernet, you will due to congestion or disconnects, often see interesting traffic on the segment.

Attackers sniffing on a local segment will use the traffic to locate other hosts of interest, So we generate fake TCP/IP fragments between a honeypot and another host. We then send that fragment to every MAC listed. Passive listeners see a packet addressed to them at Layer-2 but who's IP addresses belong to other hosts. Works for Ethernet broadcast too.



# Discoverability drawbacks

- Patterns get built into detection tools.
- At which point, flip the detection tools by making your live servers look like honeypots.
- Subterfuge!

# Arguments (against)

- This is just going to start an arms race
- This introduces new risk to the org.
- They are painful to deploy
- One more device spitting logs that will be ignored
- Will the attacker bump into it?
- I can fingerprint it / discover it / This is useless!
- It is a nice to have, after everything else is done
- This is hobbyist fun, not an enterprise solution
- Not another one!



Another argument that then comes up, is the usual:

“I can fingerprint it / discover it / This is useless!”



We touched briefly on spotting honeypots with todd beardsleys excellent talk on spotting Kippo, and i know even this year, theres a talk here on spotting them but thats not really the point,



# Know Your Enemy: Sebek

A kernel based data capture tool

The Honeywet Project  
<http://www.honeywet.org>  
Last Modified: 17 November 2003

## Introduction:

Back at Berkeley in the 1980's, CDF first quickly ran into a problem that still vexes honeypot researchers today: How do you observe an intruder without them noticing? In CDF's situation he was able to directly monitor serial lines to capture the keystrokes of the intruder. Much has changed since then, however our fundamental challenge has remained the same, how can we capture the activities of an intruder without them knowing? It is critical that a honeypot designed for the purposes of observing human intruders looks and behaves as any normal system, thus we must ensure that the intruder cannot detect that he or she is being observed. The traditional approach is to capture these activities by recording the associated network data. This is a desirable approach because it can be done in such a way as to be invisible to the intruder.

To nobody's surprise, Backfire has not aged well. It is increasingly common for even the most basic intruders to utilize encryption to protect their communications channels. If encryption services are not available on the victim host, it is not uncommon for them to install their own trusted services such as SSH, an encrypted SUD client, or SSL. Without the key, network based data capture tools are unable to view the channel.

To observe intruders using session encryption, researchers needed to find a way to break the session encryption. For many organizations this has proven extremely difficult. In an attempt to circumvent session encryption rather than break it, the Honeywet Project began experimenting with using kernel based modules for the purposes of capturing the data of interest from within the honeypot's kernel. These experiments led to the development of a tool called Sebek. This tool is a piece of code that lives entirely in kernel space and records either some or all data accessed by users on the system. It provides capabilities to: record keystrokes of a session that is using encryption, recover files copied with SCP, capture passwords used to log in to remote system, recover passwords used to enable Burpype protected binaries and accomplish many other forensic related tasks. This paper discusses Sebek version 2 which will be referred to as Sebek. What follows is a detailed discussion of Sebek, how it works and its value. We will examine the architecture and key components. From there, we will go down into the implementation issues and technical details of operation. Finally, we will show a usage example demonstrating the use of the Sebek, including its new web interface.

The Sebek development system starts on a Linux system. Sebek is ported to other Operating Systems such as Win32, Solaris, and OpenBSD after it working on Linux. As a result, this paper, and its examples, will be based on the Linux version of Sebek, however many of the concepts discussed here apply to the other ports as well.

<sup>1</sup>The Cuckoo's Egg, CDF Book, Pluribus Books/HarperCollins, 1990. New York, NY.

## The Architecture:

Sebek has two components: a client and server. The client captures data off of a honeypot and exports it to the network where it is collected by the server (refer Figure 1). The server collects the data from one of two possible sources: the first is a live packet capture from the network, the second is a packet capture archive stored as a tcpdump formatted file. Once the data is collected it is either uploaded into a relational database or the keystroke logs are immediately extracted. The communications used by Sebek are UDP based and as such are connectionless and unicast.



Figure 1  
Typical Sebek deployment. The client module is installed on the honeypot (in blue). The attacker's activity captured by the honeypot is dumped to the wire (hidden to the attacker) and passively collected by the Honeywet Gateway.


The client resides entirely in kernel space on the honeypot and, in the case of the Linux version, is implemented as a Loadable Kernel Module (LKM). The client can record all data that a user accesses via the `read()` system call. This data is then reported to the server over the net in a manner that is difficult to detect from the honeypot running Sebek. The server then gathers the data from all of the honeypots sending data. Because there is a standard platform independent log format the server can collect from any honeypot independent of Operating System type. Let us now take a closer look at how the client actually captures the data.

## Client Data Capture:

Data capture is accomplished with the use of a kernel module. With this module we gain access to the kernel space of the honeypot. Using this access, we then capture all `read()` activity (data). Sebek does this by replacing the stock `read()` function in the System Call Table with a new one. The new function simply calls the old function, copies the contents into a packet buffer, adds a header, and sends the packet to the server. The act of replacing the stock function involves changing one function pointer in the System Call Table.

For example:

Sebek which was released in 03 was a covert way to get keystrokes from a honeypot machine



## Detecting Sebek

- Overview
- HoneyPot Technology
  - NoSEB-Erk
  - Introduction
  - Detection
    - Avoid Logging
    - Kernels
    - Other Techniques
- Detecting Other HoneyPot Architectures
- Conclusion

Several ways to detect Sebek come to mind:

- Latency
- Network traffic counters
- Modification of sys-call table
- Finding hidden module
- Other craft in memory

Thorsten Holz – Laboratory for Dependable Distributed Systems

21st Chaos Communication Congress - slide #16

In 2004 there was then an awesome talk on detecting Sebek.  
And its pretty simple to figure the ways that Sebek can be detected  
Its in the ways that the machine with Sebek is different to one without it..



## “Red Pill” by Joanna Rutkowska

● Overview

HoneyPot Technology

NoSEB-EaK

Detecting Other HoneyPot  
Architectures

● UML-based HoneyPots

● VMWare-based HoneyPots

● Others

● Further things

Conclusion

```
int swallow_redpill () {
    unsigned char m[2+4],
        rpill[] = "\x0f\x01\x0d\x00\x00\x00\x00\xc3";
    *((unsigned*)&rpill[3]) = (unsigned)m;
    ((void(*)())&rpill)();
    return (m[5]>0xd0) ? 1 : 0;
}
```

- Get contents of the interrupt descriptor table register (IDTR)
- SIDT instruction (encoded as 0F010D[addr])
- Can be used in user-mode, but returns sensitive register
- On VMWare, relocated address of IDT is e.g. at 0xffXXXXXX

Thorsten Holz – Laboratory for Dependable Distributed Systems

21st Chaos Communication Congress - slide #55



And one of the ways it could be detected is with Joanna Rutkowskas red pill.. And this is all great and useful research

And it makes it clear that Sebek can be defeated and detected in a lab..

But so what?



# These are the wrong arguments



This is almost perfectly the wrong argument..

We do want to increase the attacker cost

We do want the attacker to have work harder..

We do want that shot that the attacker stumbles and we are waiting for that one alert to get you into the game..

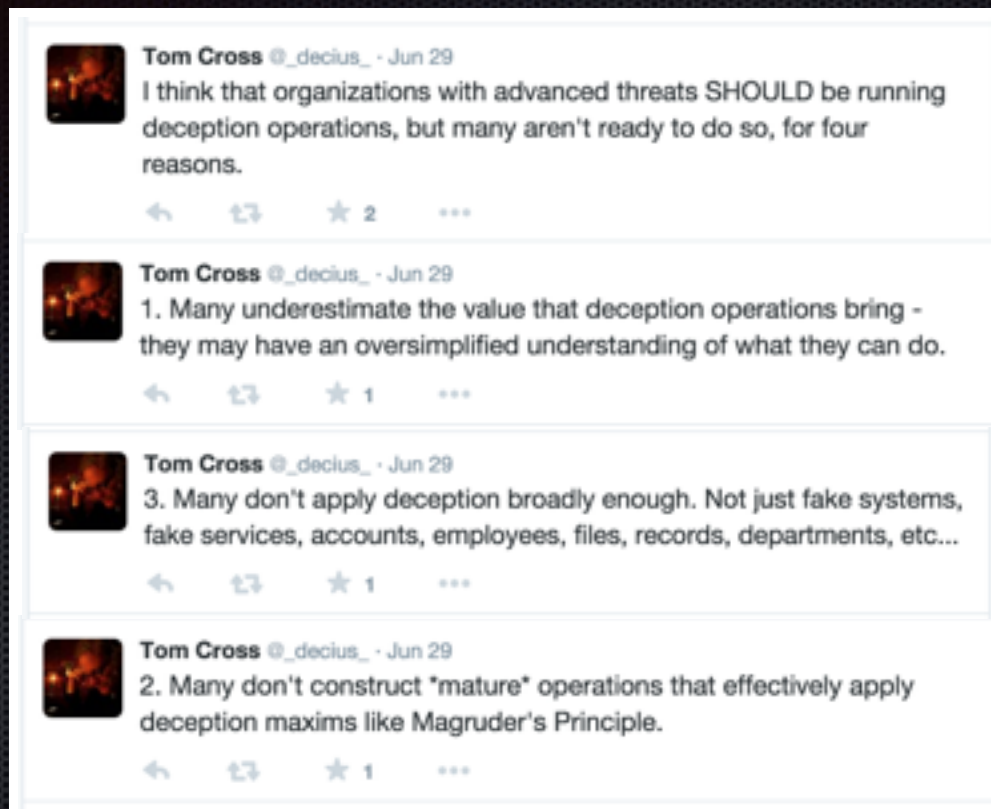
The people who took Target and RSA and OPM were not running Sebek red pill checks. Even if Sebek was spottable in a lab, it would have helped those guys. Just because something is beatable in the lab, doesn't mean it won't solve a bunch of your problems..

# Arguments (against)

- ✦ This is just going to start an arms race
- ✦ This introduces new risk to the org.
- ✦ They are painful to deploy
- ✦ One more device spitting logs that will be ignored
- ✦ Will the attacker bump into it?
- ✦ I can fingerprint it / discover it / This is useless!
- ✦ It is a nice to have, after everything else is done
- ✦ This is hobbyist fun, not an enterprise solution
- ✦ Not another one!



We touched on this argument before.. when we spoke about being too busy to install it..

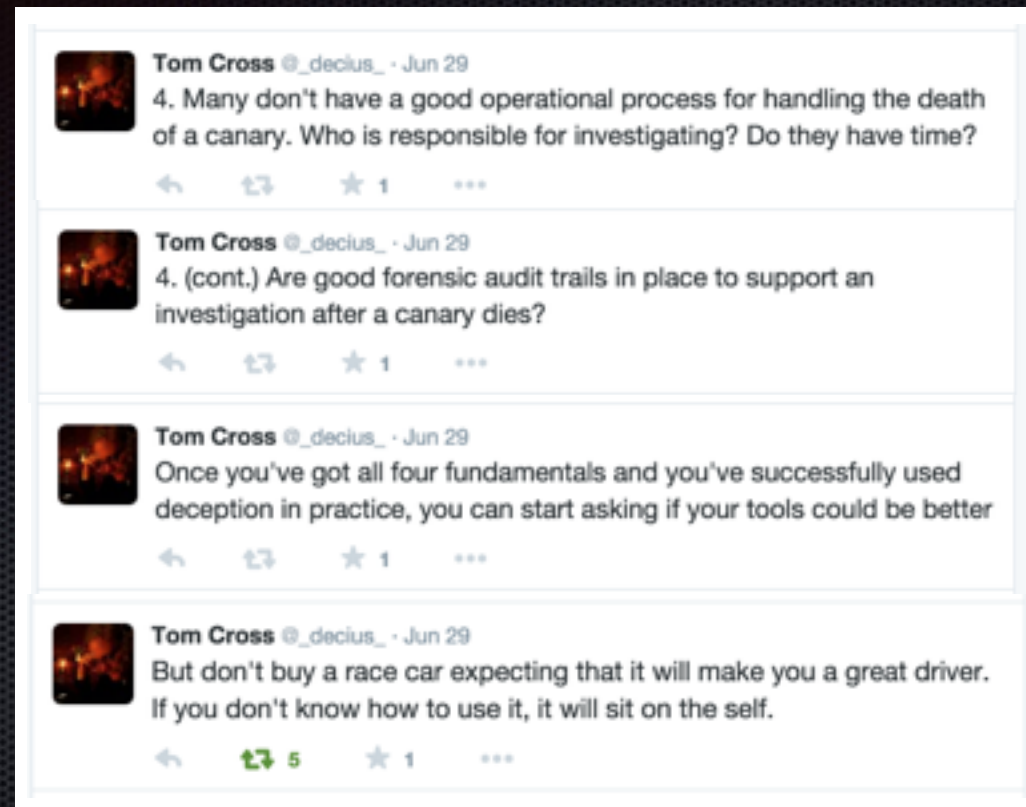


[https://twitter.com/\\_decius\\_/status/615519239651336193](https://twitter.com/_decius_/status/615519239651336193)



The uber smart Tom Cross recently had a tweet storm talking about how companies shouldn't stray into deception till they have the basics covered.





[https://twitter.com/\\_decius\\_/status/615519239651336193](https://twitter.com/_decius_/status/615519239651336193)



He (understandably) riled against people buying a new thing and expecting magic.. He strongly advises understanding deception first..



[https://twitter.com/\\_decius\\_/status/615519239651336193](https://twitter.com/_decius_/status/615519239651336193)



And i would agree that that approach would be bad.. if it was massively expensive or a massive waste of time..  
But.. for example with our Canary product, we are looking at a deployment in 5 minutes.. its what canary would take..  
its what open canary would take..

Now if it only takes 5 minutes to get setup.. that tradeoff doesn't look so bad.. (In fact we have customer testimonials calling it the best security money they ever spent!)

Even quicker than that.. is to actually use honey tokens / canary tokens

# Canary Tokens

- Spitzner (2003), Spafford & Kim (1994)
- Not a new Concept.
- City of “Agloe” ?

<http://www.symantec.com/connect/articles/honeytokens-other-honeypot>

<http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2114&context=cstech>



So lets talk Canary Tokens.. Essentially we are talking about an old concept. Stafford and Kim spoke about inserting special data/documents that could be alerted on, and Lance Spitzner called this honeytokens. It dates way back to old map makers who used to create fake towns to detect competitors stealing their maps. Like the town of Agloe. The town was invented as a cartographical ruse in the 1930s, but it somehow ended up becoming real. Agloe's story might be the strangest in the already strange history of copyright traps in maps.



# Canary tokens are simple

- Unique tags that can be embedded in a wide number of places.
  - Documents, emails, databases, file watchers, process watchers, LinkedIn, Bitcoin, Imgur
  - (And we've barely scratched the surface.)
- When that tag is triggered, you get an alert.



Which leads us then to our next tool release, Canary Tokens.

These are really simple, unique tags that can be embedded in a wide number of places.

## Demo 4: Basic Canarytoken

You can hit canary tokens right now:

- Browse [canarytokens.org](https://canarytokens.org)
- Generate a token for “[cdemo@thinkst.com](mailto:cdemo@thinkst.com)”, memo is “Web bug”
- Browse the URL
- and you should get your alert.

It is worth noting, that this is just a building block.. and we can now use this in a bunch of place.. with a bunch of tools.

- Web bugs? What's the deal?



Web is just one channel to  
detect token usage

Let's look at other channels supported by canary tokens



## Demo 5: DNS channel

- Browse [canarytokens.org](https://canarytokens.org)
- Generate a token for “[cdemo@thinkst.com](mailto:cdemo@thinkst.com)”, memo is “DNS bug”
- Lookup the hostname
- Show Alert

In this case, we get notified when someone does a lookup or tries to resolve our token.

# Fundamental channels

- ✦ Output
  - ✦ Email
  - ✦ SMS
- ✦ Input
  - ✦ HTTP
  - ✦ DNS
  - ✦ SMTP
  - ✦ Imgur
  - ✦ LinkedIn
  - ✦ BTC

We currently have our Canary Tokens working as follows:

We generate alerts over email or SMS and then have a bunch of different input channels.

# Tokens can also be tied to recurring queries, which are potential flags

Examples:

- Poll Imgur pic for view count changes.
- Poll LinkedIn for profile view changes.
- Poll Bitcoin address for change in balance.



So we can setup the canary token server to poll imgur for watch counts. So we would create a reference to an imgur image, and if the count on it increases, we know we have a problem..

its worth remembering what we want here: a simple way to get a heads up that someone got owned



# Imgur

- Popular image host.
- Provides view counts on images.
- Idea: post a non-public image, embed a link in a mail with a tasty Subject.
- If view count increments, someone found the link.

# Releasing Canarytokens

- ✦ <http://canarytokens.org>
- ✦ Available under BSD license
- ✦ <https://github.com/thinkst/canarytokens>
- ✦ <https://github.com/thinkst/canarytokens-docker>
- ✦ follow @thinkstcanary for new channels



Idea is to make deployment as simple as possible.

You can use the site to get your tokens or you can deploy your own token server.

Encourage installation of private canarytokens service.

so lets talk about how you use em..



# Applied Tokenage



The use cases above were simple to understand..

Hide the token somewhere, and hope someone clicks the link or hope they lookup your DNS name.. but.. theres actually a bunch of places we can use tokens for some adhoc canary'ing



# How could tokens help spot us on a network?



So, How could tokens help spot us on a network?

caveat: already have an awesome, tuned, monitored SIEM..

(probably SIEM holes/blindspots too)

What we are looking for here, is quick, dirty but effective..

# Actions we care about

- Failed logins to databases
- Database queries on specific honey tables
- Reads on honey files
- File opens
- Directory browsing
- Process execution



If you were trying to spot bad-actors on your network:

You could deploy an agent-based solution. That's intensive (though likely worthwhile).

If you can't or won't, there's a lightweight solution aided by canaries.

examples...



# Canaryfy – Linux file reads

- Tiny daemon which waits for inotify events, and then fires a canarytoken.
- When fired, the alert encodes the filename in the query.
- `./canaryfy <new_psname> <dns_canarytoken>  
<path1> <path2> ... <pathN>`



So now we see a bunch of small tools we released that can then make use of the canary tokens.

The first one being: Canaryfy - which is a tiny Linux util that waits for inotify events, and then fires a canarytoken.

Essentially it watches on linux for a file being accessed, and notifies you via email or sms when it is touched.



## Demo 6: Canaryfy



- `cd demo4_canaryfy/`
- show contents of `/src/repos`
- Get a new DNS token from [canarytokens.org](https://canarytokens.org)
- Run `./canaryfy '[scsi_eh_33]' <token> /srv/repo/test-av3.git/README.txt`
- `cat /srv/repo/test-av3.git/README.txt`
- Alert comes through

Monitor access to a Git repo, without relying on git hooks

Or, use git hooks.

# OS X – File read

- No inotify so Canarfy doesn't work
- Kevent means a rewrite
- Quick and cheap is Dtrace



Without iNotify on OS X, we make use of Dtrace to get the same effect.

## Demo 7: OS X File reads

Our sample dtrace script simply watches for reads on a file you specify, then uses [canarytokens.org](https://canarytokens.org) to report if that file/files is/are accessed.

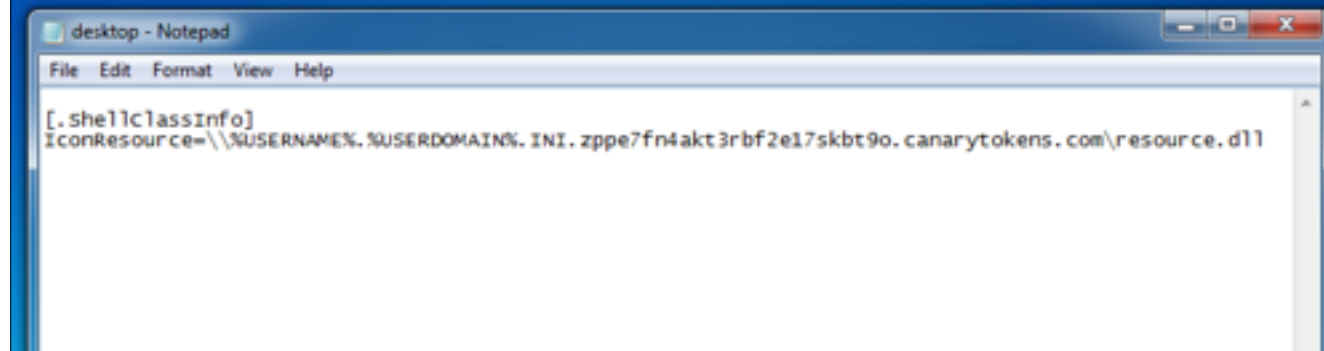


## Demo 8: OS X Process Run

We can also use dtrace to watch for commands being exec'd.

So lets send us an email/sms if someone runs id, or ifconfig, or uname -a.

# Windows – Directory browsing



```
[.ShellClassInfo]
IconResource=\\%USERNAME%.%USERDOMAIN%.INI.zppe7fn4akt3rbf2e17skbt9o.canarytokens.com/resource.dll
```

- Icons can be pulled from UNC paths
- DNS!



It turns out that windows.ini (used at least back in Windows 3.1. and still honoured) which affects how explorer.exe shows file contents allows you to set an icon-resource for a file.

This means we can get a notification on windows when someone browses a directory we care about.

Relies on FAT attributes!

## Demo 9: Share browsing alerts



Windows 7 client has V:\ mapped to a network share on Win2k12 server.  
In a sensitive directory we've placed a desktop.ini  
Client browses into target folder, triggers alert.



# File opens - Zip files

- WinZip and WinRar happily extracts the attributes needed by the desktop.ini trick.
- Re-use the same trick to determine if a Zip file has been extracted.



Same technique works to detect if someone has extracted a zip file.

# Databases

- Data leaks are what we're trying to prevent.
- Attackers in an unseen DB will map it out.
- They go for sensitive-sounding data.
- Would you exec the stored proc "get\_admin\_key()".?



Databases and canary tokens are interesting.

Some stored procedures and databases look too juicy to leave alone..

“making the theoretical  
possible”



Interestingly, for years we have been telling people to honeypot their data..

who here has? (nobody)

why? cause its not easy.. Its one of those things that consultants recommend without knowing what it would take to make happen.



# MS SQL

- Problems:
  - How to trigger on a SELECT
  - How to encode data
  - How to trigger a DNS query



Fortunately, We solved two of the three back in '07.

Triggering on a select isn't possible, but we can fake it with a view.

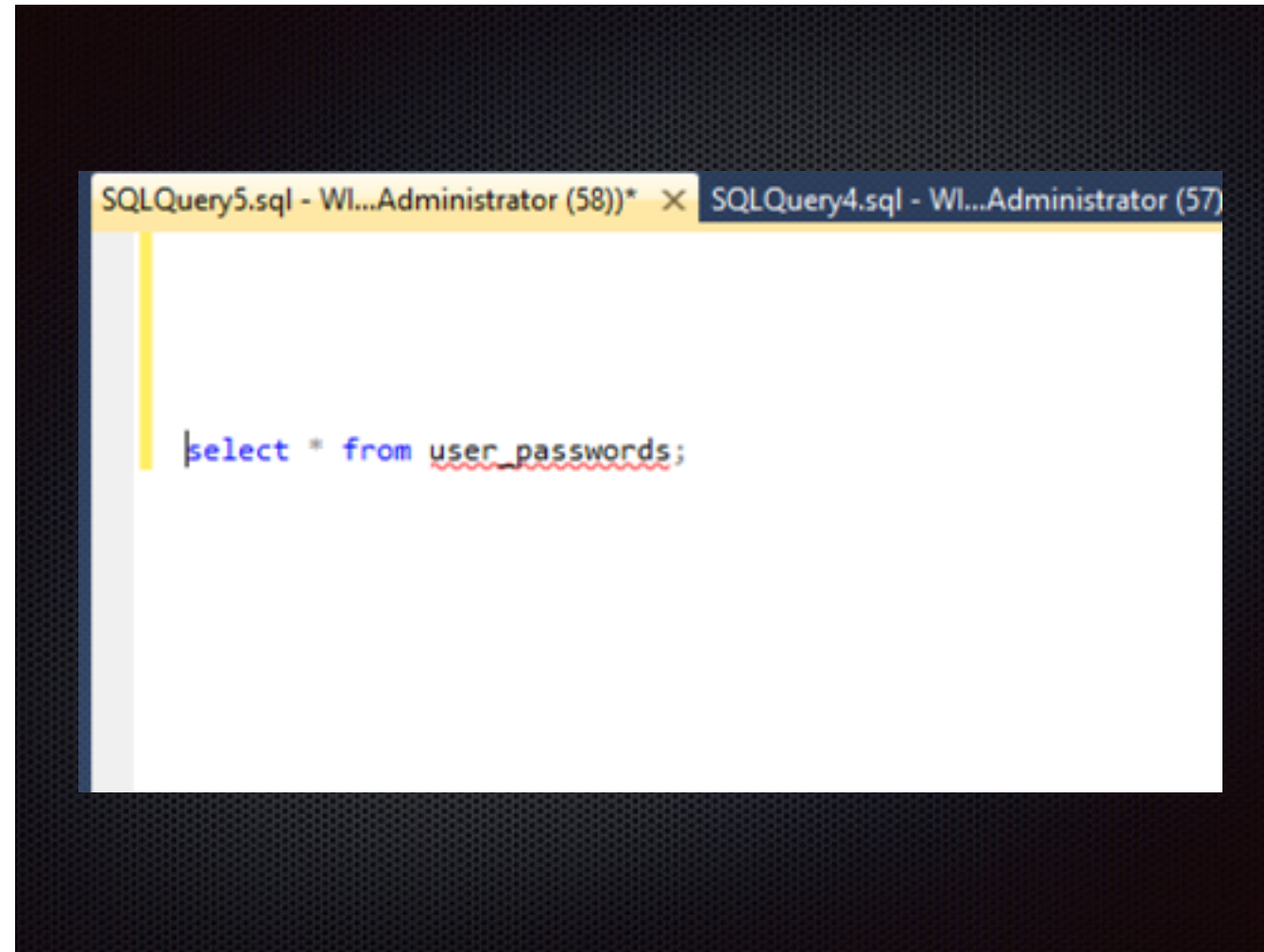
Beauty of the view is we can allow it to be select'able by anyone, but still hide its definition.

SQLQuery5.sql - WL...Administrator (58) SQLQuery4.sql - WL...Administrator (57) SQLQuery3.sql - WL...Administrator (57) SQLQuery1.sql - WIN...master (user (53))

```
CREATE function query_users(@RAND FLOAT) returns @output table (col1 varchar(max))
AS
BEGIN
    declare @username varchar(max), @base64 varchar(max), @tokendomain varchar(128), @unc varchar(128), @size int, @done int, @ran

    --setup the variables
    set @tokendomain = '4qc2plf65fthntybvja8e9bsw.canarytokens.com';
    set @size = 128;
    set @done = 0;
    set @random = cast(round(@RAND*100,0) as varchar(2));
    set @random = concat(@random, '.');
    set @username = SUSER_SNAME();

    --loop runs until the UNC path is 128 chars or less
    while @done <= 0
    begin
        --convert username into base64
        select @base64 = (SELECT
            CAST(N'' AS XML).value(
                'xs:base64Binary(xs:hexBinary(sql:column("bin")))'
                , 'VARCHAR(MAX)'
            ) Base64Encoding
```



So we can see that a Select on user\_passwords will generate an alert through [canarytokens.org](https://canarytokens.org)





**Canarytoken Mailer** noreply@canarytokens.org via mail135-4.atl141.mandrillat

8:24 PM (1 minute ago)



to me ▾

One of your canarydrops was triggered.

Channel: DNS

Time : 2015-08-05 03:23:53.595763

Memo : MSSQL view

Source IP : 74.125.187.113

SQL Server User: CORP\Administrator

Manage your settings for this Canarydrop:

<http://canarytokens.org/manage?token=4gc2plf85fthntybvja8e9bsw&auth=95cad140ebdbbeeb9a834cdc944616b>

# MySQL – Failed logins

- Different problem:
  - No internal triggers for failed logins
  - Only written to log file
- Actually, a common pattern: interesting daemon data is mixed into a bunch of less interesting logs.
- A generic solution is a log file tail'er which can trigger canarytokens.



On MySQL, we couldn't make use of a trigger. Here we wrote a generic log file tailer, that can watch for certain tokens, and then report them through [canarytokens.org](https://canarytokens.org).  
Reminder: this is completely separate from OpenCanary. There we had a fake mysql. Here, we are integrating canary tokens with an already present installation.

# Canarytokend

- Tails log files, matches regexes, triggers alerts
- Cheap way to get alerted without SIEMs



```
"No config files found. Searched ['/etc/canarytokend.conf', '/home/demo/.canarytokend.conf']"
Do you want to create a config file? (Y/n):

We're going to create a new canarytokend config file.
First up, where do you want to save the file? (/home/demo/.canarytokend.conf):

Do you want to include MySQL failed login tracking? (Y/n):
Enter the path to the MySQL error log (or enter 'help' to find it): /var/log/
mysql/error.log
Generating a new canarytoken for MySQL
Please enter an email address for receiving alerts: cdemo@thinkst.com
Please enter a short description to remind you about this token: MySQL failed
logins
Writing config file...
Done. Please run canarytokend to read the new config
```

Guided setup

```
attacker$  
attacker$ mysql -h canary -u root -p  
Enter password:  
ERROR 1045 (28000): Access denied for user 'root'@'logger-host' (using password: YES)  
attacker$ █
```



**Canarytoken Mailer** noreply@canarytokens.org via mail135-4.atl141.mandrill  
to me ▾

8:36 PM (0 minutes ago)



One of your canarydrops was triggered.

Channel: DNS

Time : 2015-08-05 03:35:56.283655

Memo : MySQL failed logins

Source IP : 204.194.237.21

Manage your settings for this Canarydrop:

<http://canarytokens.org/manage?token=1xr0vjvgsm5z6yrcbyvzmqpd6&auth=7751e5fb71e6b970a95fce147b183644>



# File opens: Word

- Standard web bug using builtin Word fields.
- Cross-platform.
- No script or prompts necessary.



Word / Office documents have a long history of allowing web-bugs

We can grab a canary token, insert it into the document or document metadata, and we have a document that will beacon home reliably when opened.

# File opens: PDF

- AdobeReader lets PDF documents requests URLs.
- Sounds great for tracking opening.
- Except it show a prompt before the request goes through.
- There is a strange behaviour though...



On AdobeReader, we found a side curiosity, that works as follows:



# File opens: PDF

- AdobeReader pre-flights some DNS.
- For the token domain **<token>.canarytokens.org**
  - Reader queries '**A canarytokens.org**'
  - If a record is returned, resolution halts.
  - Useless for tokens though.
- If the server returns NXDomain however, Reader will query:
  - **A <token>.canarytokens.org**
  - Woot!



Useless for tokens since we can just tell that something looked up our domain.



# File opens: PDF

- The user still sees a popup, but it has no effect.
- The queries are being sent while the popup is displayed.
- So with some DNS finagling we can determine if a PDF file has been opened in AdobeReader.



Useless for tokens since we can just tell that something looked up our domain.

# JS Page Copied

```
1. load in client browser;  
2. Check if URL is .mydomain.com;  
3. If not { load http://  
    canarytokens.com/traffic/tags/  
    terms/  
    y7rhuc09bh5xhwrcyb513dqzp/  
    submit.aspx }
```



With just 3 lines of Javascript on a webpage, we can now tell when someone copies our page (usually a pre-cursor to a phishing attack)

# lets get silly..

- ✦ LinkedIn
- ✦ google / bing-ads as canaries ?

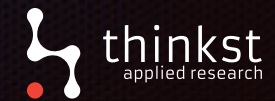


We can make use of dummy profiles on LinkedIn tied to the company. When they are browsed, [canarytokens.org](https://canarytokens.org) will notice



```
$ echo thiswontworkwillit |md5
```

```
61735ef62cf584aa378d238144444347
```



To this end, we can even make our own canary token notifier.  
We generate a random hash, and save it somewhere..

Microsoft Corporation

bing ads Home Campaigns Reports Opportunities Tools Import Campaigns

Account  
Thinkst Applied Research -  
B013FVYS

Search

All Campaigns  
Campaign #1  
Ad group #1

Shared Library  
Bulk Operations

Bing Ads will be down for maintenance between Saturday, July 18, 2015 7:00 AM (GMT-7) and Saturday, July 18, 2015 12:00 PM (GMT-7). You will not be able to sign in, but campaigns will continue to run. [Learn more](#)

All Campaigns  
Campaign: Campaign #1  
7/1/2015 - 7/17/2015

Status: **Enabled** Budget: 5.00/day Locations: All locations

Ad Groups Settings Ads **Keywords** Ad Extensions Product Targets Change History Dimensions

View: Keywords

Add keywords Edit Details Automate **NEW!** Export All keywords Columns **NEW!** Filters Keyword Text

	Keyword	Ad group	Delivery	Bid	Clicks	Imp.	CTR	Avg. CPC	Spend
	61735e652c7d584aa378c25814444347	Ad group #1	Eligible	3.15	0	0	0.00%	0.00	0.00
	Search total				0	0	0.00%	0.00	0.00
	Content total				0	0	0.00%	0.00	0.00
	Deleted items total				0	0	0.00%	0.00	0.00
	Overall total - 1 keywords				0	0	0.00%	0.00	0.00

Show rows: 50

Only clicks are MCC accredited.

© 2015 Microsoft Legal Privacy & Cookies Advertise Developers Support Blog Feedback Microsoft



We then register on Bing and buy that hash as a keyword to advertise against.

Microsoft Corporation

Bing Ads will be down for maintenance between Saturday, July 18, 2015 7:00 AM (GMT-7) and Saturday, July 18, 2015 12:00 PM (GMT-7). You will not be able to sign in, but campaigns will continue to run. [Learn more](#)

**Account**  
Thinkst Applied Research -  
8813FVRS

Search

All Campaigns

Shared Library

**Bulk Operations**  
Automated rules  
Bulk edits

**Automated rules**  
Save time by automating your most common tasks. [Learn how to create rules](#), and [see some examples](#).

**Edit rule: Increase to estimated top of page bid**

Apply to

Do this: Increase to estimated top of page bid ☒ Maximum bid  USD

When  contains  ☒ Exact ☐ Phrase ☐ Broad ☐ Content X

+ Add another

How often   using data from

Display time: ☐ 12-hour ☒ 24-hour

Time zone:

Rule name:

Email results:

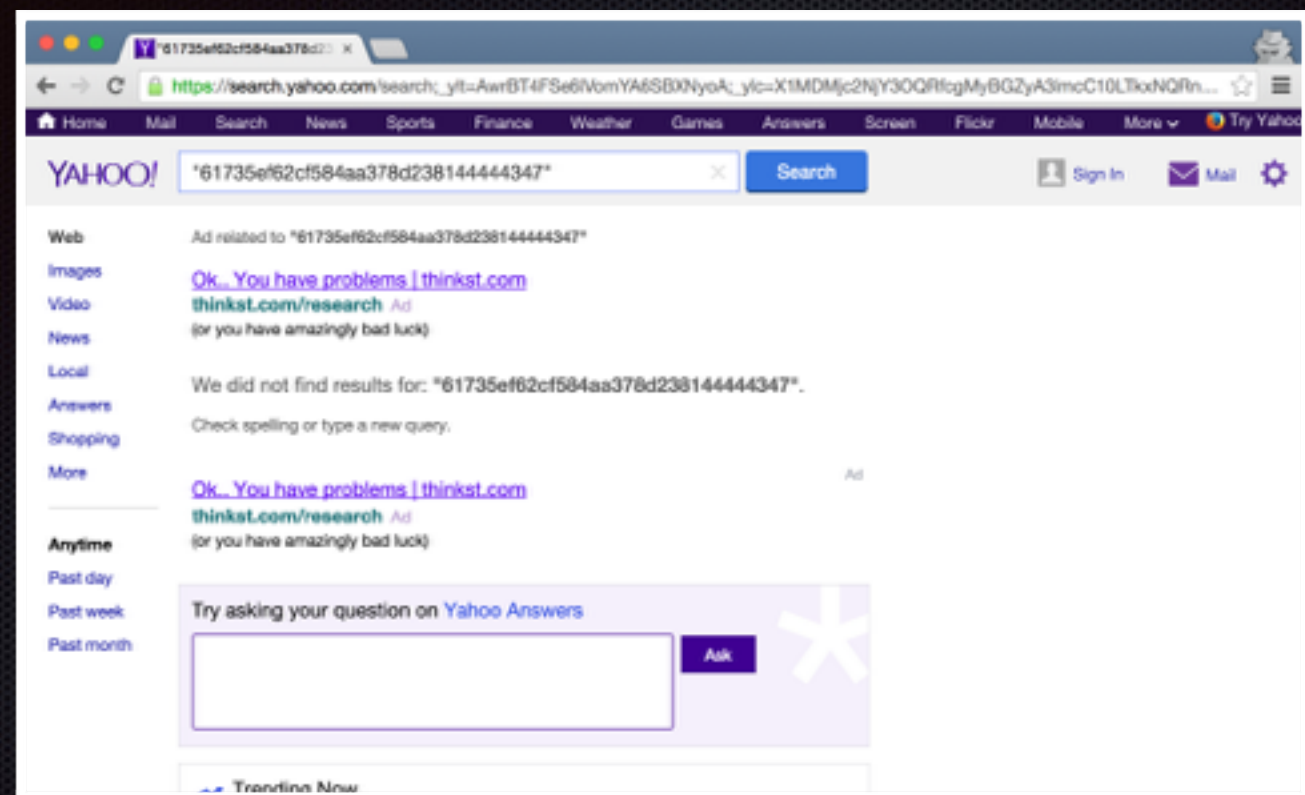
© 2015 Microsoft Legal Privacy & Cookies Advertise Developers Support Blog Feedback

Microsoft



(we can offer to bid high for the hash - which adds a “notify us when run” feature)





Someone searching for the hash, after finding it on our machine, then gets the “no result” - but also gets served our Advert.

Microsoft Corporation

bing ads Home Campaigns Reports Opportunities Tools Import Campaigns

Account  
Thinkst Applied Research -  
B013FVY3

Search

All Campaigns  
Campaign #1  
Ad group #1

Shared Library  
Bulk Operations

All Campaigns  
Campaign: Campaign #1  
This month: 7/1/2015 - 7/19/2015

Status: Enabled Budget: 5.00/day Locations: All locations

Ad Groups Settings Ads Keywords Ad Extensions Product Targets Change History Dimensions

Create ad group Edit Details Automate NEW Export All ad groups Columns NEW Filters Ad group name

	Ad group	Delivery	Search bid	Content bid	Clicks	Imp.	CTR	Avg. CPC	Spend	Avg. pos.
	Ad group #1	Eligible	0.30	0.30	0	1	0.00%	0.00	0.00	
	Search total				0	1	0.00%	0.00	0.00	
	Content total				-	-	-	-	-	
	Deleted items total				0	0	0.00%	0.00	0.00	
	Overall total - 1 ad groups				0	1	0.00%	0.00	0.00	

Show rows: 50

© 2015 Microsoft Legal Privacy & Cookies Advertise Developers Support Blog Feedback Microsoft



We get to see that we now have one impression for the advert



Your Bing Ads automated rule has run

Customer ID: 14440352

Account ID: 43050898

Dear Bing Ads user,

The automated rule you created, Increase to estimated top of page bid, resulted in the following:

0 change(s)

0 error(s)

Below are the first 10 results. [View all results.](#)

Entity	Old value	New value	Status
--------	-----------	-----------	--------

You are receiving this mail because in the above automated rule, you indicated that you wanted to be notified about certain results. You can change this setting by following the instructions [here](#).

Thank you for using Bing Ads.

Sincerely

The Bing Ads team

and bing will actually mail us to say that it served our ad..

Again.. People say this is all nice to have if everything else is done...

I'm all for focus.. but if its easy enough.. and if you can fire and forget.. why wouldn't you do it?

3 minutes to setup canary.. maybe 10 for openCanary.. you could use tokens in 30 seconds.. literally one weekend of work.. why wouldn't you?



# Arguments (against)

- This is just going to start an arms race
- This introduces new risk to the org.
- They are painful to deploy
- One more device spitting logs that will be ignored
- Will the attacker bump into it?
- I can fingerprint it / discover it / This is useless!
- It is a nice to have, after everything else is done
- This is hobbyist fun, not an enterprise solution
- Not another one!



Another complaint that comes up then, is that this is all hobbyist fun and not “real”

## Canary Migration Patterns



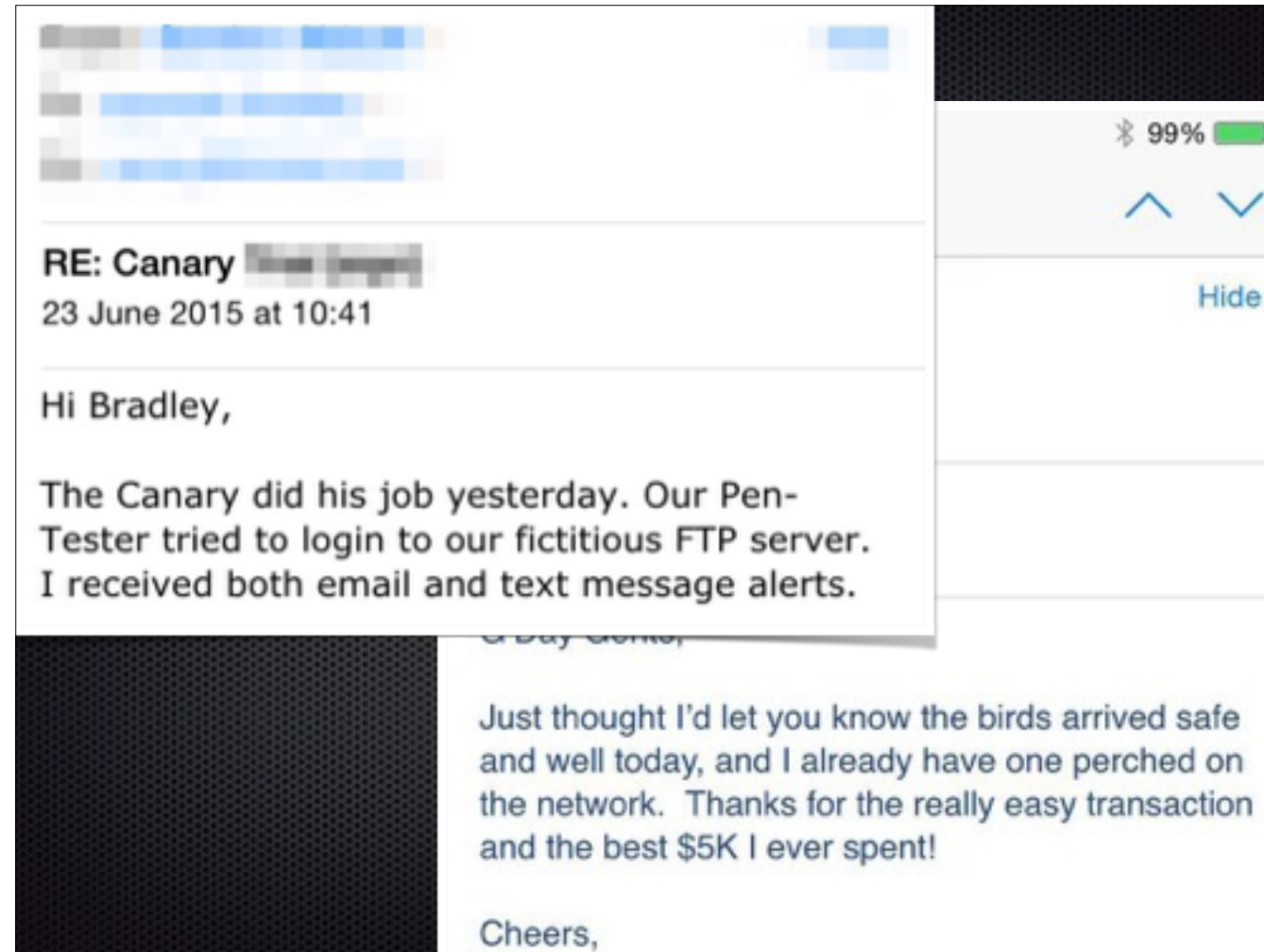
We have been selling our Canaries since the middle of the year, and they are running at nice shops / networks.

## Canary Migration Patterns



In fact they are now on some really serious networks worldwide.





We also have some really good fanmail on this..  
and the main reason i say this is, we are not going away

# Arguments (against)

- This is just going to start an arms race
- This introduces new risk to the org.
- They are painful to deploy
- One more device spitting logs that will be ignored
- Will the attacker bump into it?
- I can fingerprint it / discover it / This is useless!
- It is a nice to have, after everything else is done
- This is hobbyist fun, not an enterprise solution
- Not another one!



In fact it ties well into the last complaint: If someone says: “not another honeypot?”

honevd	<a href="http://www.honevd.org/release.php">http://www.honevd.org/release.php</a>
Kippo	<a href="https://github.com/desaster/kippo/commits/">https://github.com/desaster/kippo/commits/</a>
Kojoney	<a href="http://sourceforge.net/projects/kojoney/files/">http://sourceforge.net/projects/kojoney/files/</a>
conpot	<a href="https://pypi.python.org/pypi/Conpot">https://pypi.python.org/pypi/Conpot</a>
artemisa	<a href="http://sourceforge.net/projects/artemisa/files/">http://sourceforge.net/projects/artemisa/files/</a>
delilah	<a href="https://github.com/Novetta/delilah">https://github.com/Novetta/delilah</a>
Elastichoney	<a href="https://github.com/jordan-wright/elastichoney">https://github.com/jordan-wright/elastichoney</a>
mhn	<a href="https://github.com/threatstream/mhn/commits/">https://github.com/threatstream/mhn/commits/</a>
Glastopf	<a href="https://github.com/mushorg/glastopf/commits/">https://github.com/mushorg/glastopf/commits/</a>
Dionaea	<a href="http://src.carnivore.it/dionaea">http://src.carnivore.it/dionaea</a>
Amun	<a href="http://amunhoney.sourceforge.net">http://amunhoney.sourceforge.net</a>
Honeytrap	<a href="http://src.carnivore.it/honeytrap/loq/">http://src.carnivore.it/honeytrap/loq/</a>
Wordpot	<a href="https://github.com/qbrindisi/wordpot/commits/">https://github.com/qbrindisi/wordpot/commits/</a>
Shockpot	<a href="https://github.com/threatstream/shockpot/">https://github.com/threatstream/shockpot/</a>
Artillery	<a href="https://github.com/trustedsec/artillery/commits/">https://github.com/trustedsec/artillery/commits/</a>
Decoy Server	<a href="http://www.svmantec.com/region/req_eu/">http://www.svmantec.com/region/req_eu/</a>
Nepenthes	<a href="http://src.carnivore.it/nepenthes/loq/">http://src.carnivore.it/nepenthes/loq/</a>
Tom's Honeyd	<a href="http://labs.inquardians.com/tomshoneyd.html">http://labs.inquardians.com/tomshoneyd.html</a>
..	

We understand.. The graveyard is filled with unmaintained honeypot projects..

artemisa	2010-03-08	2011-02-22
Kojoney	2006-03-31	2010-02-16
Honeyd	2003-02-15	2007-05-27
Amun	2008-10-07	2010-03-04
Nepenthes	2006-02-19	2009-10-16



# We are all in



But we have a strong incentive to keep this up.. We earn money of commercial canary.  
We believe in the technology & We believe it matters..  
Because we sell canary - we are financially incentivised to keep at it..

# Parting Thoughts

1. Canaries / Honeypots are a good ideas that are largely ignored;
2. Detecting infiltrations months or years after the initial compromise is increasingly inexcusable;
3. Canary, OpenCanary, and CanaryTokens, you really shouldn't have any excuse..



So 3 parting thoughts..

# Questions ?



Check out:

<https://canary.tools>

<http://opencanary.org>

<http://canarytokens.org>

ping us at @marcoslaviero , @azhrdesai or @haroonmeer