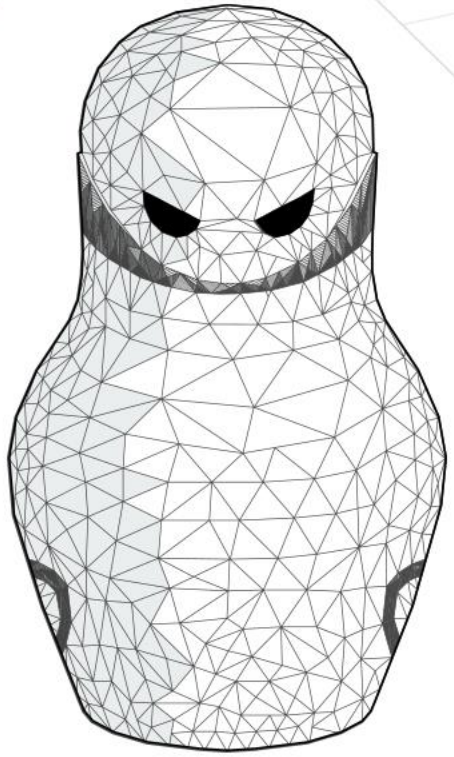


2015 ZERO NIGHTS

HOOKED-BROWSER NETWORK WITH BEEF AND GOOGLE DRIVE

Denis Kolegov, Oleg Broslavsky, Nikita Oleksov
Tomsk State University
Information Security and Cryptography Department

National Research Tomsk State University

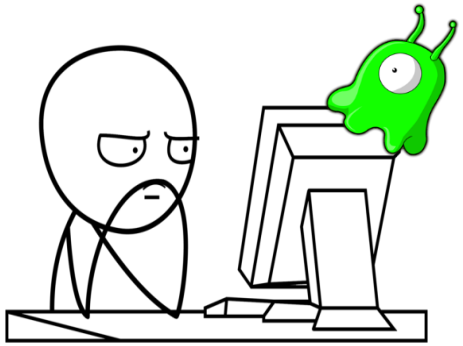
www.zeronights.org

Post-exploitation

After vulnerability exploitation and taking control over victim's system intruder should find a way to establish communication between browser and C&C server

Common communication channels in web application are:

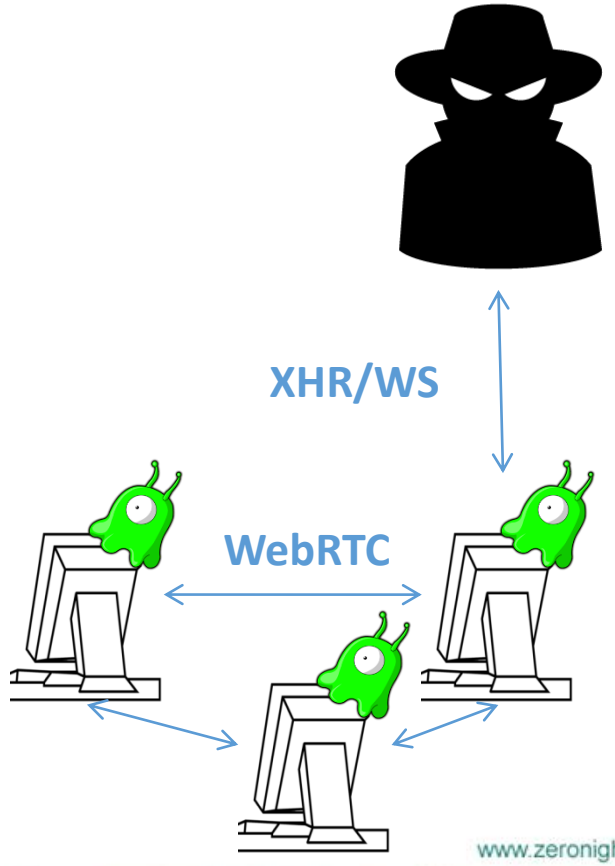
- XML HTTP Request
- WebSockets
- WebRTC



Prehistory

Not long ago **Christian Frichot** (aka @xntrik) in his talk at Kiwicon 2014 presented **BeEF WebRTC** extension

"One of the biggest issues with BeEF is that each hooked browser has to talk to your BeEF server. To try and avoid detection, you often want to try and obfuscate or hide your browsers. Using this bleeding-edge web technology, we can now mesh all those hooked browsers, tunneling all your BeEF come through a single sacrificial beach-head."

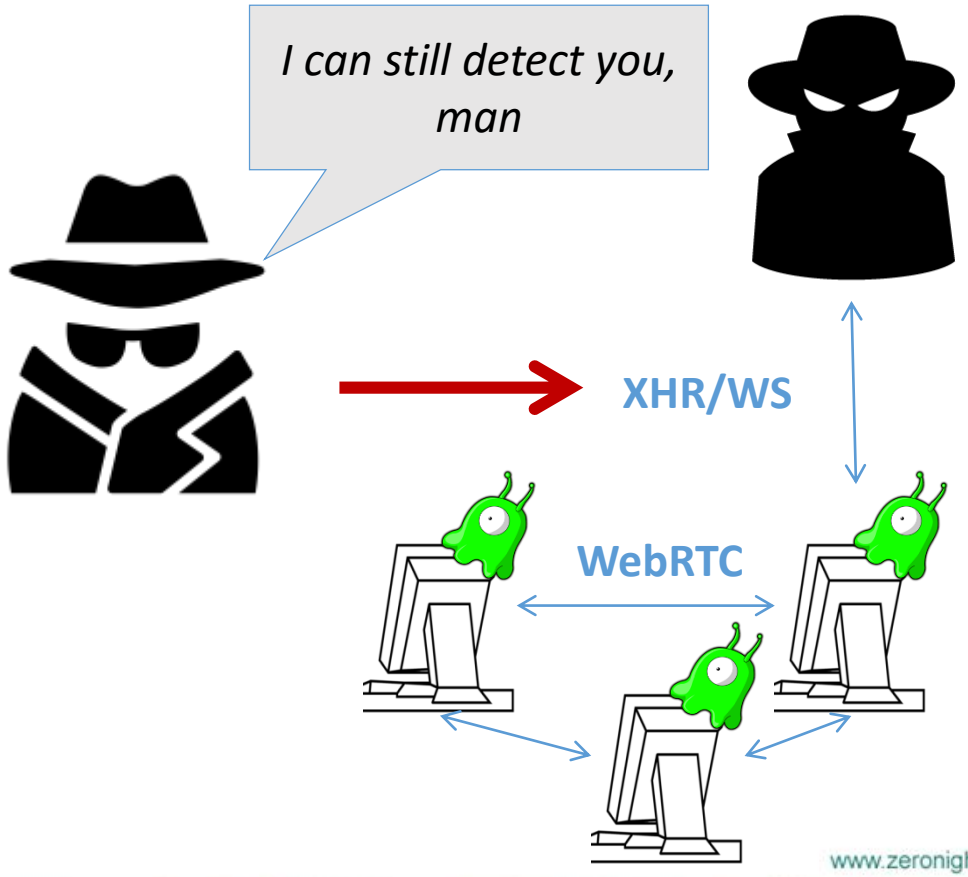


Not totally invisible

The last direct communication channel can be tracked or blocked by IDS/IPS. So we decided to find out a way to get rid of it

The main idea is to use some trusted server as a communication channel as it is done in projects like

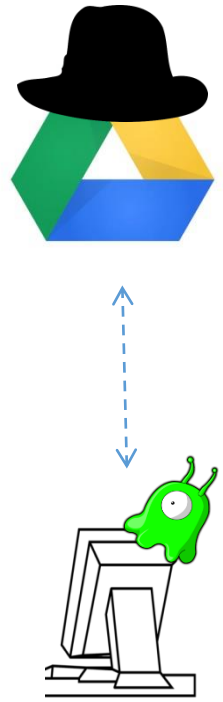
- Gcat
- Twittor



Our previous researches

Our team have researched a new type of covert timing channels based on HTTP cache control headers and couple of ways to implement it in different environments

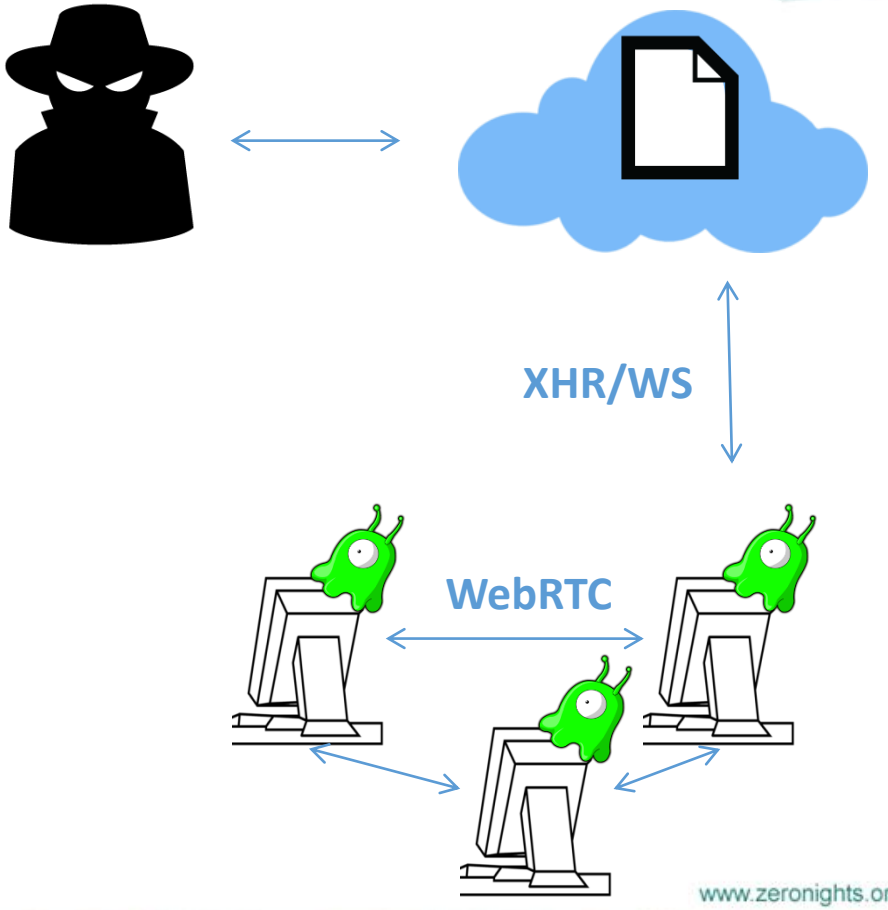
One of such environments was Google Drive, so we decided to use it in a communication channel one more time



We need a cloud

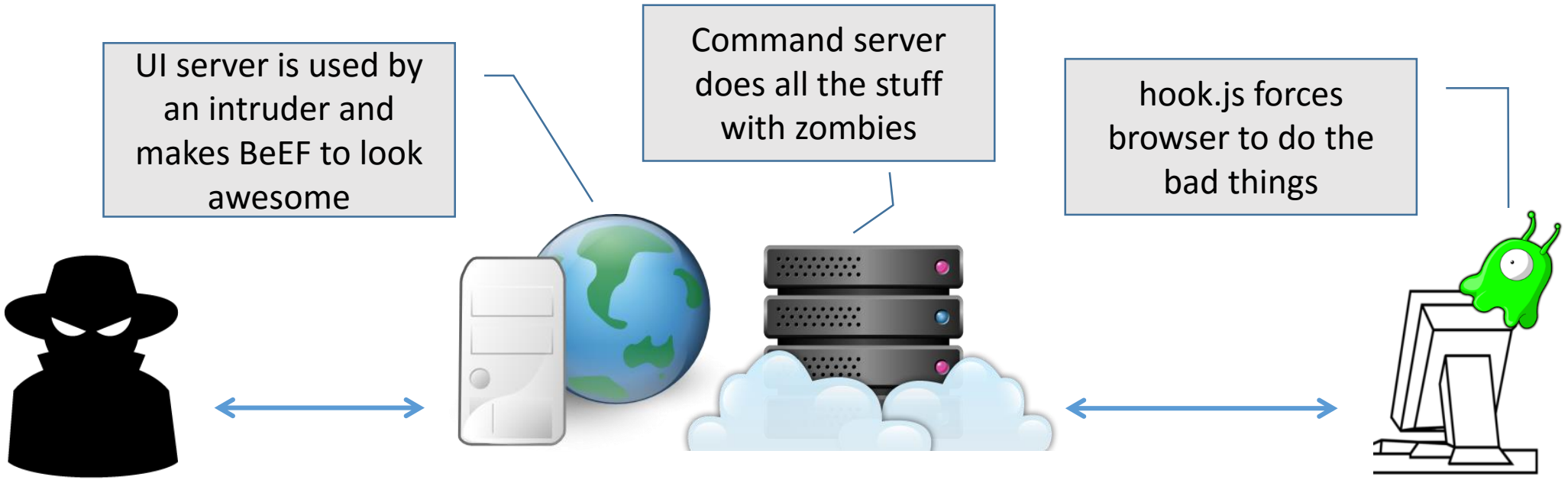
Sometimes BeEF need to send a really huge amount of data so why not to use something that is designed to work with it?

Cloud storage services like Google Drive or Dropbox are trusted (not marked as suspicious activity) in most networks and have a nice API to work with them using JavaScript



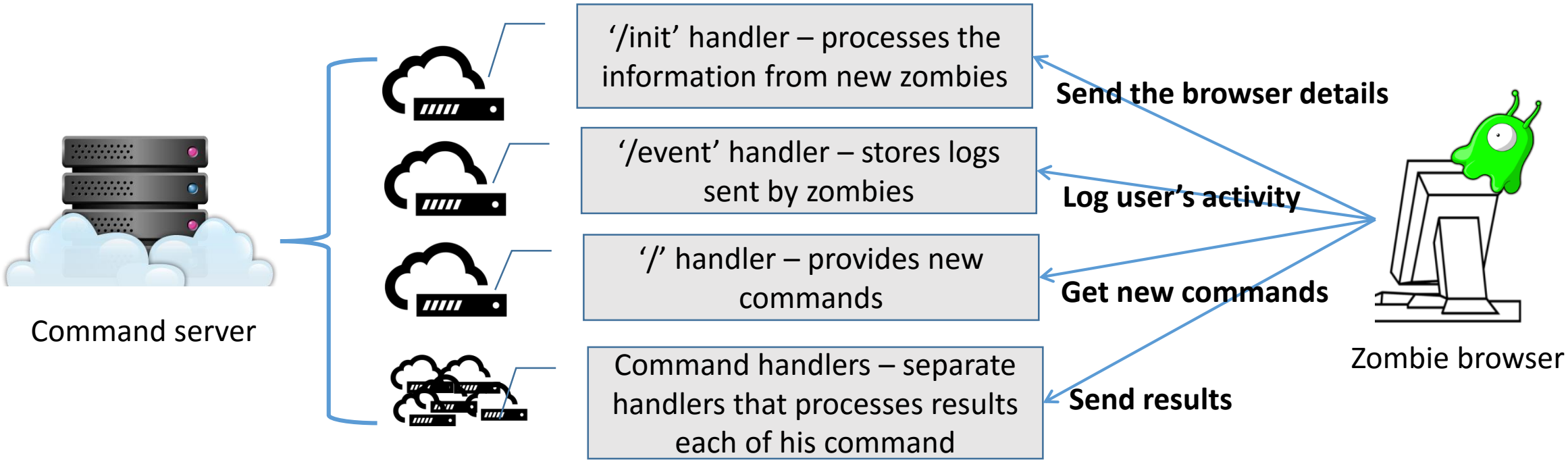
Under the BeEF's hood

Let's understand what's going on in the BeEF



Command server details

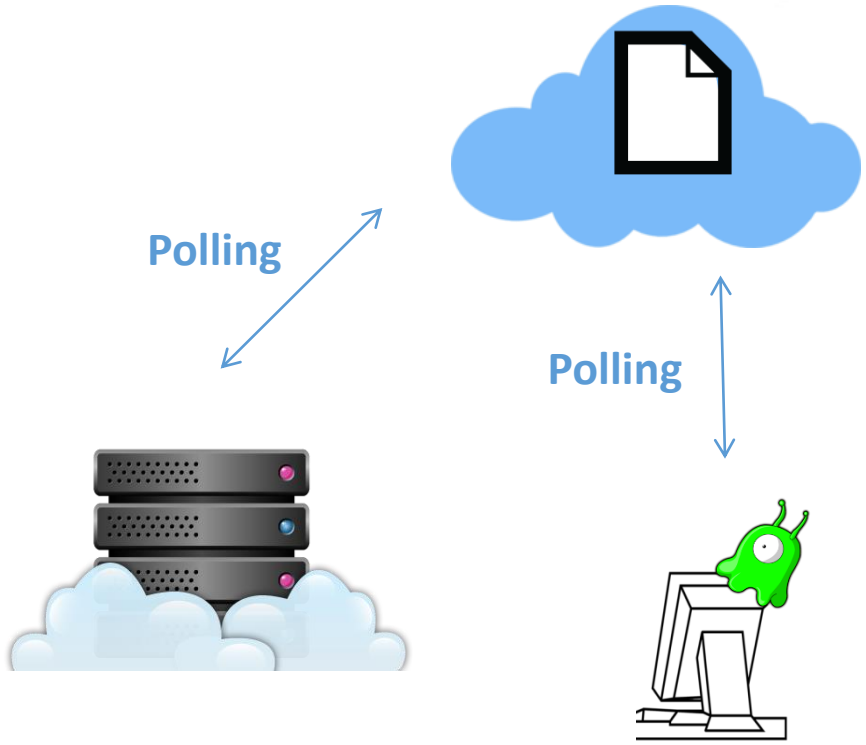
Command server can be viewed as a bunch of handlers each of which is doing its own job



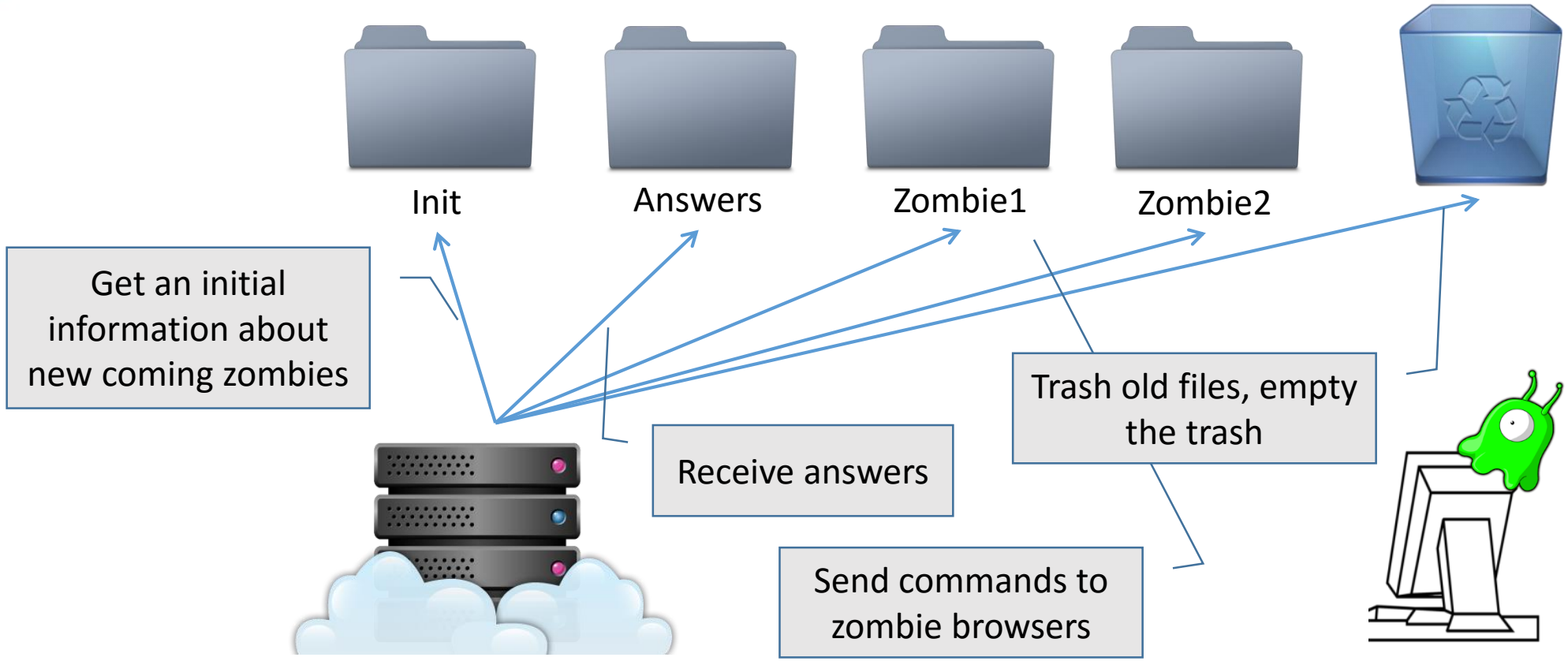
The beginning of indirect communication

As we can we BeEF is designed as common network application with active client and passive server

So the first of all we should teach the server to tell with zombies via cloud using polling model



Indirect communication on the server side



Indirect communication on the client side

Send browser default as a first request to the server

Pull commands from its own folder and move read commands to the trash

Store all information for command server

One more thing is access

To perform action via Google Drive API we need 3 different keys:

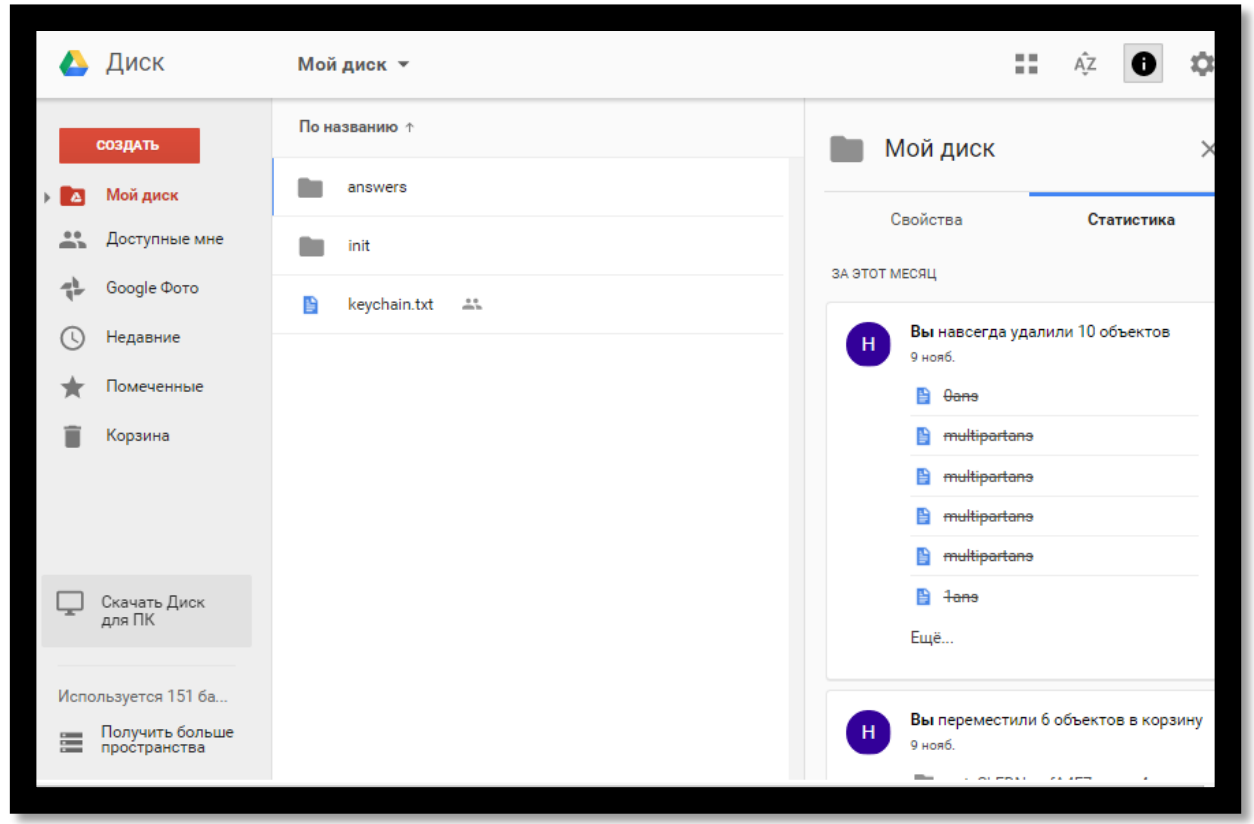
Master key – used by server to update **Auth key** via OAuth

Auth key – used by client and server to perform any write access on the Google Drive

API key – used by client to read renewed **Auth key** from special **keychain** file

Hooked-browser network with BeEF and Google Drive

Proof of Concept: https://youtu.be/_RfBUEcvynM
https://github.com/tsu-iscd/beef-drive



Thank you for the attention!

Denis Kolegov @dnkolegov dnkolegov@gmail.com

Oleg Broslavsky @yalegko ovbroslavsky@gmail.com

Nikita Oleksov @neoleksov neoleksov@gmail.com