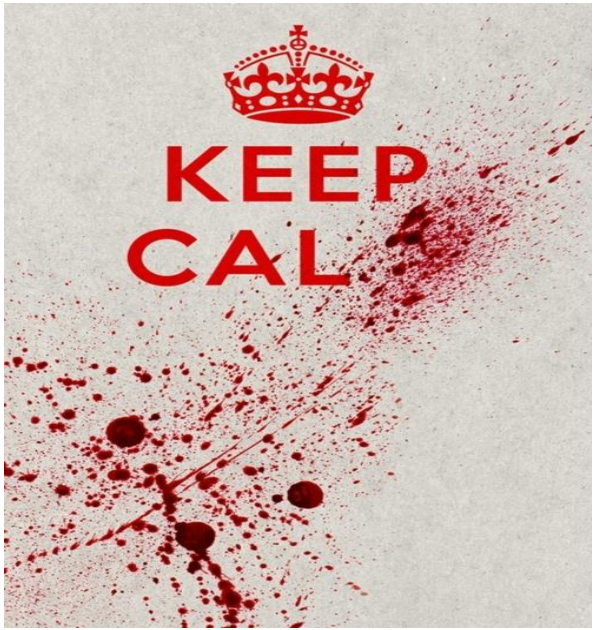
GRR Rapid Response

GRR Workshop @
CERN 2018



Mikhail Bushkov, Ben Galehouse,
Miłosz Łakomy, Andreas Moser

Agenda

- Introduction to GRR
- Hands on work - exercises
- Roadmap and Discussion

Remote Forensics at Google Scale

- Joe saw something weird, check his machine
  - (p.s. Joe is on holiday in Cambodia and on 3G)
- Forensically acquire 25 machines for analysis
  - (p.s. they're in 5 continents and none are Windows)
- Tell me if this machine is compromised
  - (while you're at it, check 100,000 of them - i.e. "hunt" across the fleet)

What is GRR?

- “GRR Rapid Response”
- Agent based forensics investigation tool
- Open source (Apache License 2.0)

What is GRR?

- Features:
  - Machine information (hardware, users, ...)
  - Basic forensics (files, registry, process list, ...)
  - Sleuthkit integration (raw disk access)
  - Process memory acquisition / scanning (using Yara)
  - and many more...

What is GRR?

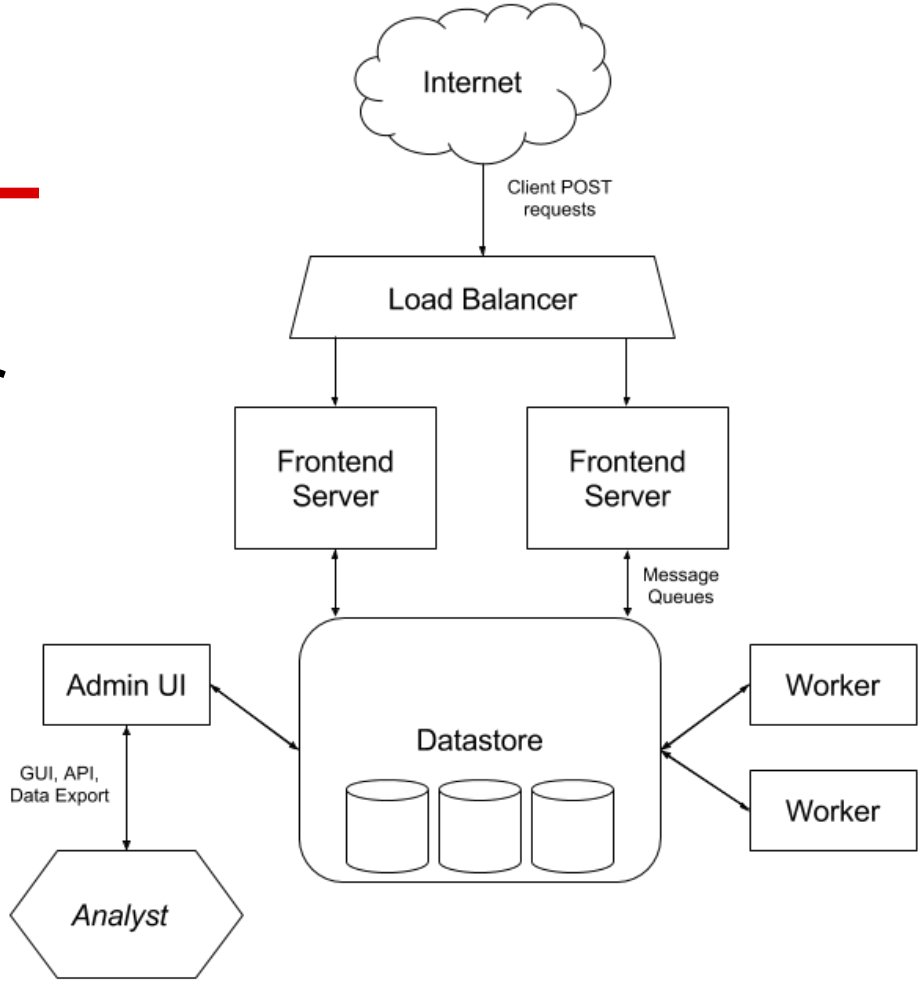
- Built, maintained, used by Google… and others
  - 5.5 full time developers (in ZRH)
  - Lots of people helping out

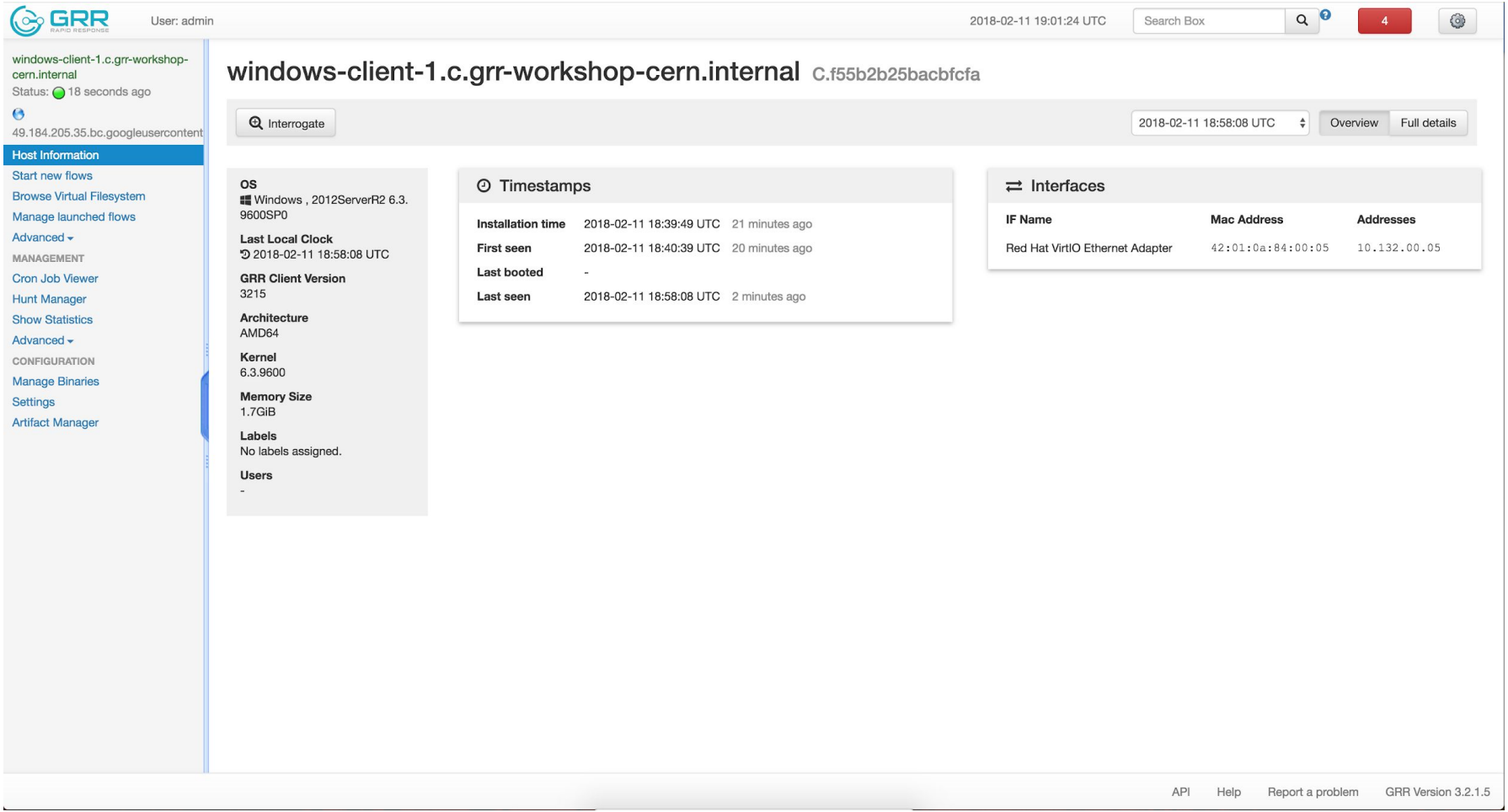
- Built by engineers for engineers

FAQ: http://grr-doc.readthedocs.io/en/v3.2.1/faq.html

Architecture

- Client
- Frontend Server
- Datastore
- Admin UI
- Worker
- API




GRR
RAPID RESPONSE
User: admin
2018-02-11 19:01:24 UTC
Search Box
4
windows-client-1.c.grr-workshop-cern.internal
Status: 18 seconds ago
49.184.205.35.bc.googleusercontent
Host Information
Start new flows
Browse Virtual Filesystem
Manage launched flows
Advanced
MANAGEMENT
Cron Job Viewer
Hunt Manager
Show Statistics
Advanced
CONFIGURATION
Manage Binaries
Settings
Artifact Manager
windows-client-1.c.grr-workshop-cern.internal C.f55b2b25bacbfcfa
Interrogate
2018-02-11 18:58:08 UTC
Overview
Full details
OS
Windows , 2012ServerR2 6.3. 9600SP0
Last Local Clock
2018-02-11 18:58:08 UTC
GRR Client Version
3215
Architecture
AMD64
Kernel
6.3.9600
Memory Size
1.7GiB
Labels
No labels assigned.
Users
-
Timestamps
Installation time 2018-02-11 18:39:49 UTC 21 minutes ago
First seen 2018-02-11 18:40:39 UTC 20 minutes ago
Last booted -
Last seen 2018-02-11 18:58:08 UTC 2 minutes ago
Interfaces
IF Name
Mac Address
Addresses
Red Hat VirtIO Ethernet Adapter
42:01:0a:84:00:05
10.132.00.05
API
Help
Report a problem
GRR Version 3.2.1.5

Datastore

- GRR runs on MySQL (and SQLite for demo purposes)
  - Abstraction makes replacing possible
- Forensic data is stored by client id
  - GRR datastore schema is organized around AFF4 paths: generic but not user-friendly
  - Datastore changes in progress
  - GRR UI/API is used to export data

Datastore

- Forensic data is versioned
  - Usually nothing is deleted ever
    - Just new versions are added
    - -> The complete history is kept in the GRR DB

Clients

- Clients for Windows, Mac, Linux
  - Stable, robust, low-impact
    - Python + PyInstaller
    - Memory, CPU limited
    - Watchdog process
  - Contains very little logic
    - encoded in “Flows” on the server
  - We’re experimenting with making the client smarter

Communications

- Client polls the server for work
- Defaults to once every 10 minutes
- Messages are protobufs
- Signed and encrypted end to end
- Default connection via “HTTP”
- Fleetspeak subproject will modularize and modernize comms
  - Separate process (only running GRR when needed)
  - TLS

Flows

- Flows encapsulate logic
  - Clients are “dumb”
    - Client actions are basic building blocks
      - “Get me this file”, “List this directory”
    - -> Clients don’t need to be updated frequently
  - Flows interpret the data received
    - Ex.: Get browser plugins
      - Downloads file(s) with known paths
      - Parses received data to find plugin directories
      - Downloads those directories

Flow Processing

- Flows are processed on the Worker(s)
  - Flows can be suspended and stored in the datastore
    - If client goes away, flow just resumes at a later time
  - In the end, results are produced
    - For analysis in the UI
    - Can be exported from the UI (SQLite, csv)
    - Automatic export plugins (BigQuery)

Hunts

- Hunting is running a flow on all the clients in the fleet
- Fleet checks
  - I found this suspicious file on one machine, which other boxes have it too?
- Baselining
  - Download the RunOnce Keys from all machines
  - Which ones stand out?
  - Which ones are new compared to last week?

Audit Controls

- GRR is remote root equivalent
- Audit controls
  - Multi-party authorization
  - Audit hooks
  - Audit log
- Approval-based system built in
  - User, reason, expiry
  - Disabled for the workshop for simplicity

GRR Quick Install

- Not necessary for the workshop today

- Setting up your own GRR server
  - Many moving parts but DEB package is available
    - Should be up and running in a few mins
  - DEBs releases are done periodically
  - HEAD DEB is built on every commit

GRR Quick Install

- Can also be installed via:
  - Docker
  - Pip from released packages or source (for development)
- Terraform for demo purposes
  - Instructions for GCE on Github

Workshop test environment

- GRR server at
  - https://35.189.221.117
- Clients connected:
  - 2 Windows Server machines
  - 2 Ubuntu Linux machines
- Don’t connect your machines

Demo

- Workshop server

Exercise 1 - Introduction to GRR

- Server: https://35.189.221.117
- User accounts: user<n>/password<n>
- Search the client database
  - “.” gives all clients
- Look at client info
  - Look at different OSs
- Check out /fs/os in the VFS
  - Also /fs/tsk, /registry for Win clients

Launching Flows

- Launching flows demo

FileFinder

- Flow to search for files by multiple criteria
  - path, name, contents (literal / regex), time
- When a file matches, an action is run
  - Download, hash, stat (report existence)

FileFinder

- Demo, this will be next exercise

Exercise 2 - File downloading

- Client C.0acaeff8f5d6ef64
- Get a list of all DLLs (*.dll) in C:\Windows\System32
- Get the partition boot sector C:\$BOOT
  - Windows API will hide this! Use pathtype TSK
- There is a file (or two) containing the string "malware" in <Desktop>\Browsercache. Try to find it.

Registry Analysis

- Registry analysis works like file analysis
  - Keys / Directories, Values / Files
  - Same operations supported!
    - Globbing
    - Content match on values

Exercise 3 - Registry

- Client C.0acaeff8f5d6ef64
- Poke around using the Registry finder
  - Should be straightforward - similar to FileFinder
  - Please don’t schedule huge recursive listings.
- One of the values in HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer contains the string "malware". Which one?

YARA process memory scanning

- YARA process memory scanning built in for all platforms
- Process memory collection is in the works
- GRR has Rekall built in but:
  - It’s deprecated/unsupported
  - Turned off by default

YARA process memory scanning

- Demo

Exercise 4 - YARA memory scanning

- Use the “Yara Process Scan” flow to run YARA scanning directly on a client.
  - Enable “Advanced” (click the gear icon on top-right) mode to be able to use the flow
  - Set per-process timeout to 60
  - Use sample YARA signature:

```
rule Example
{
    strings:
        $text_string = "grr"
    condition:
       $text_string
}
```

Hunts

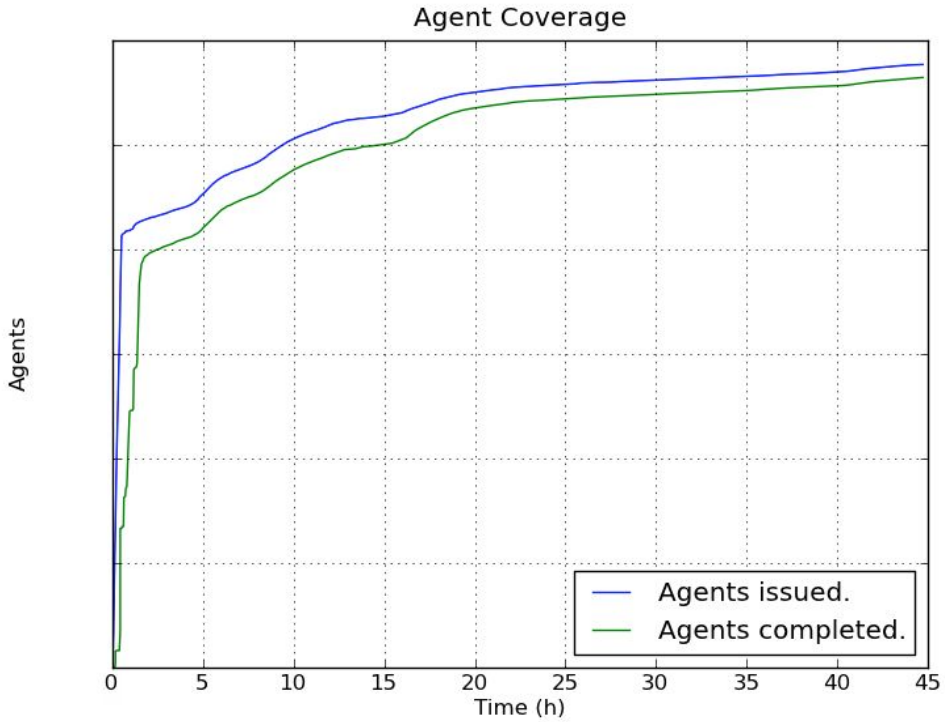
- Demo time - Collect Notepads and Export

Hunt Performance

- Longish lead time
  - Foreman delay
  - Client poll delay

- Once started, checks the whole fleet in hours
  - Mostly depending on client availability

Hunt Performance



Exercise 5 - Fleetwide Process List

- Get a list of all processes running on Windows machines in the test setup
  - Bonus task, do it also for Linux
- Look at hunt stats
  - Cpu used, network used, worst performers
- Export hunt results in CSV, YAML or SQLite format and find all processes with the process name containing “net”

Embedded Flash Malware

- Inspired by Hacking Team attack
  - Flash based attack inside Office document
- How would we go about finding this using GRR?

Exercise 6 - Hunt Embedded Flash

- There are .doc files in C:\Windows\Temp on the Windows machines
- Run a hunt to find the documents that contain embedded Flash
  - That is, they contain the literal “ShockwaveFlash.ShockwaveFlash”
- Download ZIP archive with found files

Artifacts

- Flows are too tricky for simple things
- We wish we could share information better
- Too much duplicated code
  - -> Let's generalize to Artifacts

Artifacts

- Define what to collect
- Define how to parse it
- Define the result they produce
- Data only, no code
- Yaml based format

https://github.com/ForensicArtifacts/artifacts

Artifacts

- Example Artifact:

```
name: WindowsEventLogSecurity
doc: Security Windows Event Log.
sources:
- type: FILE
 attributes:
   paths: ['%%environ_systemroot%%\System32\winevt\Logs\SecEvent.evt']
   separator: '\'
conditions: [os_major_version < 6]
labels: [Logs]
supported_os: [Windows]
urls: ['http://www.forensicswiki.org/wiki/Windows_Event_Log_(EVT)']
```

Artifacts

Knowledge Base Interpolation

%%environ_allusersprofile%% → c:\Documents and Settings\All Users

%%systemroot%% → c:\Windows\System32

%%users.appdata%%

→ c:\Documents and Settings\foo\AppData\Roaming

→ c:\Documents and Settings\bar\AppData\Roaming

→ c:\Documents and Settings\baz\AppData\Roaming

https://github.com/google/grr/blob/master/grr/proto/grr_response_proto/knowledge_base.proto

Artifacts

- Demo - Artifact Collector flow

Exercise 7 - Artifacts

- Check out the Artifact Collector flow
  - Collect an artifact
    - Event Log? …
- You suspect that the machine C.0acaeff8f5d6ef64 was owned by a drive by download. Can you show one of the users went to www.bugtrack.net using Chrome?

Roadmap - Discussion Points

- Scaling GRR
- Memory forensics
- Tool integration

Scaling GRR

- Goal:
  - 100k connected clients work out of the box

- Backend migration Bigtable to relational DB.
  - Currently WIP

Memory Forensics in GRR

- Rekall integration deprecated
  - Future of memory forensics unclear in general

- Process memory analysis as a stop-gap

Tool Integration

- Always looking for more tools to integrate
  - Have TSK, Yara, cloud services (Terraform, BigQuery, …)
  - osquery is a good candidate?!

- GRR API makes it easy to integrate your own!
  - Python GRR API library included
  - PowerGRR - PowerShell based GRR automation done by Swisscom

Fleetspeak

- New comms layer
  - Golang!
  - Easy integration of new client side services

Discussion

Questions?? Comments?? Suggestions??

grr-users@googlegroups.com
https://github.com/google/grr