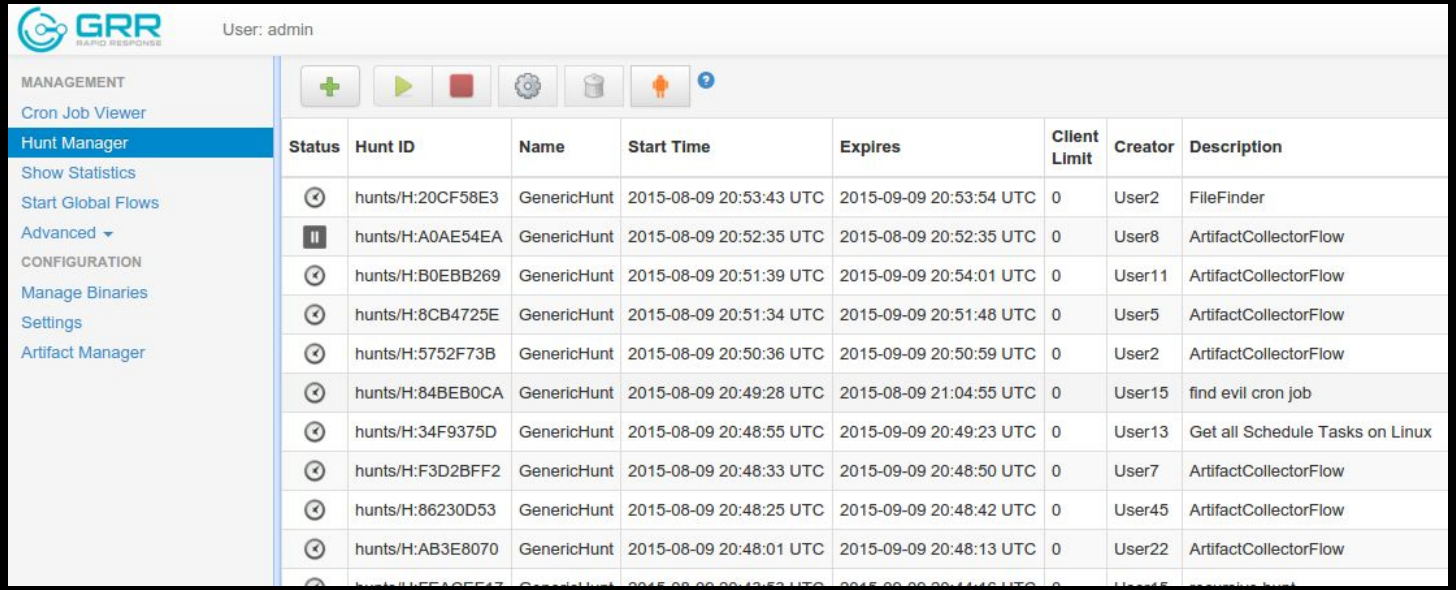
GRR Hunting
GRR Meetup Oct 2015
Greg Castle @mrgcastle

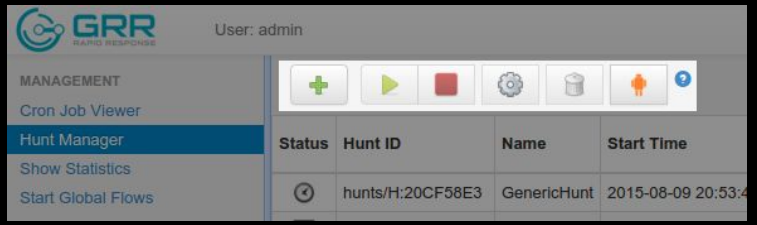
Hunting process



Status	Hunt ID	Name	Start Time	Expires	Client Limit	Creator	Description
	hunts/H:20CF58E3	GenericHunt	2015-08-09 20:53:43 UTC	2015-09-09 20:53:54 UTC	0	User2	FileFinder
	hunts/H:A0AE54EA	GenericHunt	2015-08-09 20:52:35 UTC	2015-08-09 20:52:35 UTC	0	User8	ArtifactCollectorFlow
	hunts/H:B0EBB269	GenericHunt	2015-08-09 20:51:39 UTC	2015-09-09 20:54:01 UTC	0	User11	ArtifactCollectorFlow
	hunts/H:8CB4725E	GenericHunt	2015-08-09 20:51:34 UTC	2015-09-09 20:51:48 UTC	0	User5	ArtifactCollectorFlow
	hunts/H:5752F73B	GenericHunt	2015-08-09 20:50:36 UTC	2015-09-09 20:50:59 UTC	0	User2	ArtifactCollectorFlow
	hunts/H:84BEB0CA	GenericHunt	2015-08-09 20:49:28 UTC	2015-08-09 21:04:55 UTC	0	User15	find evil cron job
	hunts/H:34F9375D	GenericHunt	2015-08-09 20:48:55 UTC	2015-09-09 20:49:23 UTC	0	User13	Get all Schedule Tasks on Linux
	hunts/H:F3D2BFF2	GenericHunt	2015-08-09 20:48:33 UTC	2015-09-09 20:48:50 UTC	0	User7	ArtifactCollectorFlow
	hunts/H:86230D53	GenericHunt	2015-08-09 20:48:25 UTC	2015-09-09 20:48:42 UTC	0	User45	ArtifactCollectorFlow
	hunts/H:AB3E8070	GenericHunt	2015-08-09 20:48:01 UTC	2015-09-09 20:48:13 UTC	0	User22	ArtifactCollectorFlow

Hunting process (with client limit)

1. + (new hunt)
2. play (request approval)
3. play (run hunt)
4. check results
5. cog (set client limit to 0)
6. play



Demo: collect /usr/sbin/*

Hunt approver - what/why?

Equivalent of a code review for your hunt, make sure:

Achieves right results efficiently
Client and server impact is reasonable

Achieves right results efficiently

Find a webshell for IIS from a report. Two md5s and an a couple of strings from the aspx.

Hunts:

C:**100, Action=HASH

C:**100.aspx, Context regex=pwnies

Achieves right results efficiently

C:**100, Action=HASH

Nope, incredibly wasteful of CPU and disk IO. Hunting hashes is almost always wrong.

C:**100.aspx, Content regex=pwnies

Less bad, but full filesystem scans perform very poorly with current system (we want to fix this)

Achieves right results efficiently

C:\Inetpub**20.aspx (and other specific paths)

Content regex=pwnies

Size < 5 MB

Windows machines

Action=Download

Minimal impact, all results are interesting

Rule of thumb #1

Constrain your file hunt to the point that any results are interesting enough to download and analyse

Achieves right results efficiently

Fleetcheck: HackingTeam on linux

%%users.homedir%%/.config/autostart/.whoopsie-*.desktop

Linux machines

Action=Download

Achieves right results efficiently

%%users.homedir%%/.config/autostart/.whoopsie-*.desktop

Too specific to this version of the implant, i.e. fragile. We should hunt techniques where possible, not specific malware.

Achieves right results efficiently

%%users.homedir%%/.config/autostart/**

Size < 20 MB

Linux machines

Action=Download

Run query on results (CSV export for now) for HT specific malware.

Rule of thumb #2

Collect general persistence techniques from the client, make malware-specific queries on the server

Client/server impact is reasonable

What if you are unsure about impact?

Complicated hunts should always be tested first.

Run as a flow on your own machine first and check CPU, bytes sent.

Client limit vs. Client rate

Limit: use for impact testing e.g. run on 100 machines.

Rate: use for spreading out impact of lots of results hitting the server at the same time.

Client/server impact is reasonable

Always use client limits, unless you're *very* confident impact is *very* low.

Use 100 clients as first sample size.

Look at worst performers stats

Client/server impact is reasonable

Demo: check hunt and flow client impact stats.

Client/server impact is reasonable

Default client rate is slow (20/min), time to schedule the hunt on various fleet sizes:

- 5k machines: 4 hrs
- 10k machines: 8 hrs
- 20k machines: 16 hrs
- 100k machines: 80 hrs

Client/server impact is reasonable

Go slower (client rate = 5) if:

- not time sensitive;
- lots of results expected; or
- hunt is expensive client resource-wise (more chance to stop it)

e.g. Collecting C:\Windows\System32*.exe

Client/server impact is reasonable

Go a bit faster (client rate = 100) if:

- time sensitive;
- few results expected; and
- hunt is cheap client resource-wise

5k machines: 1 hr

10k machines: 100 min

20k machines: 3 hrs

100k machines: 17 hrs

Client/server impact is reasonable

Go fast as possible (client rate = 0) if:

- very time sensitive;
- very few results expected; and
- hunt is very cheap client resource-wise

e.g. Stat’ing a few specific files

What a ‘bad hunt’ looks like

Client crashes (nanny kills)

Lots of clients stuck in ‘outstanding’

Hitting per-host byte limits

High CPU usage (> 20 min)

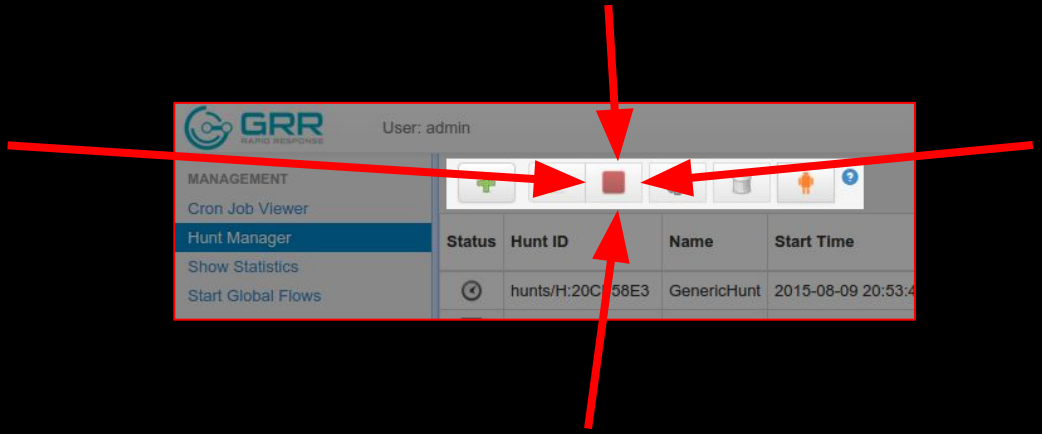
Flows not progressing

Big DB rows

The stop button is your friend

No new clients pick up the hunt

Kills in-progress actions at next state transition



Links

These slides and more on our publications page:

https://goo.gl/sigU0g

For a general GRR intro check out my “Intro to GRR”