Virtualized hackers

András Veres-Szentkirályi

Silent Signal
vsza@silentsignal.hu

Bez(a)Dis Security Weekend
November 12–14, 2010

Who am I

- IT security expert @ Silent Signal
  - 1 year old startup focusing on penetration testing
- Proud member of H.A.C.K. (Hackerspace Budapest)
- Currently attending M.Sc. course in IT @ BUTE
- Hacking at every level
  - hardware, firmware, software
  - networks of all kind
  - people
  - life

What am I going to talk about

1. Theory: wargames 101
   - Intro
   - Planning
2. Practice: Hacktivity 2010 wargame
   - Design
   - Implementation
   - Lessons learnt

WTF wargame

- military and hobby people use it as conflict simulation

WTF wargame

- military and hobby people use it as conflict simulation
- made it into hacker slang after WarGames (1983)
  - (see wardialing, wardriving, warbiking, warboating, . . .)
- "a server that is set up specifically for the purpose of being hacked into. This allows the hacker to have a server to hack into, without the need to worry about the legal issues, as the owner is knowingly allowing this to happen."

  – WordIQ.com

History

July 5–6, 2003 first Hacktivity

August 14–15, 2004 first Hacktivity with wargame

September 20–21, 2008 "our" first wargame

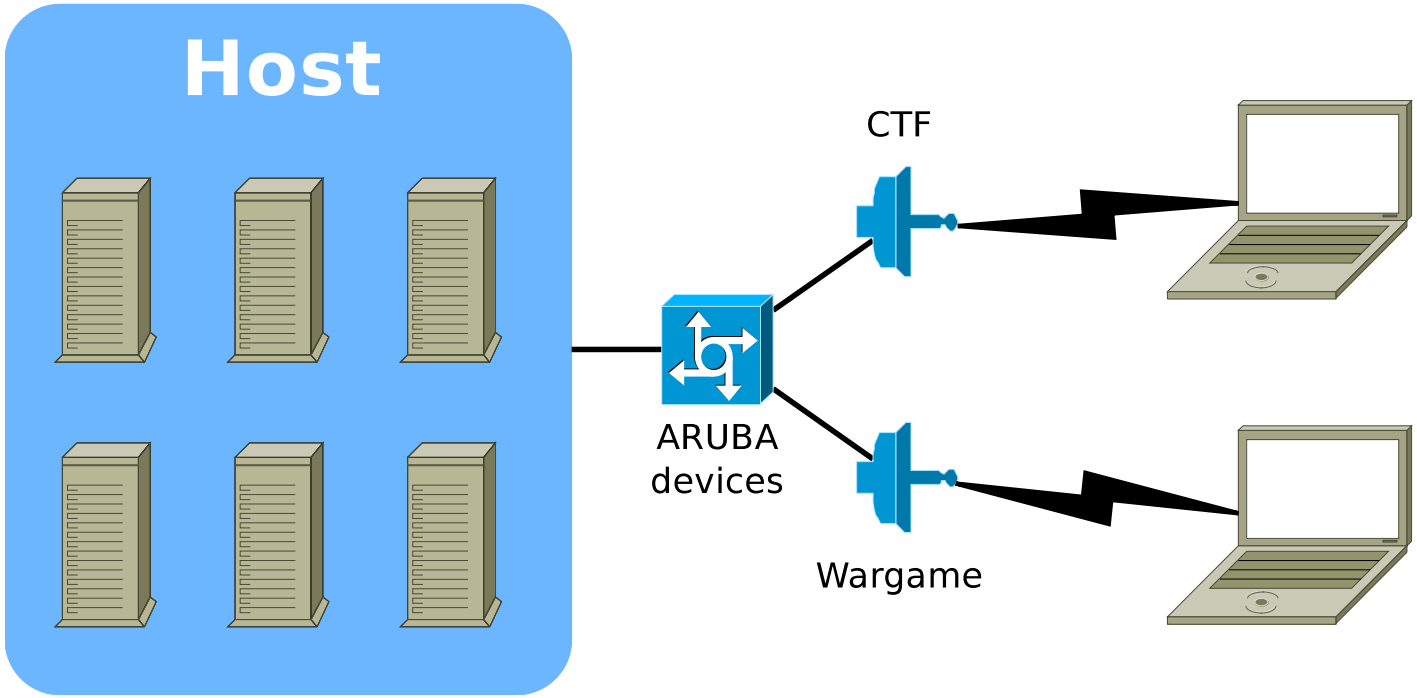
May 8–9, 2010 "How Strong is Your Fu?" – inspiration

September 18–19, 2010 first official Silent Signal wargame

Parameters

- Number of hosts
  - available time (1-2-3 days, with or without talks)
  - size of audience (distribution, fun for everyone)
  - platform (see below)
- Platform and level of pwnage (webadmin, user, root)
  - budget (HW, SW license fees)
  - size of audience
  - skills of audience (hardcore exploit coders vs. web kiddos)
- Exploitability
  - trivial – fun for beginners
  - treasure hunt – fun for people with weird thoughts (or lucky)
  - googleable – fun for kiddos
  - google+effort – fun for novices
  - ungoogleable – double-edged sword

Playground



- one host
- multiple guest VMs (targets and scoreboard)
- two networks (Wargame and CTF)

Targets

- target: a VM that players have to pwn in order to get points
- pwning can be proved by showing evidence
  - UNIX-like systems have `/root/proof.txt`
  - Windows systems have an Administrator's desktop
- players can share proofs
  - it needs to be changed in a regular way
  - it should be automated
  - scoreboard–target sync needs to be maintained
- targets are designed to be pwnd $\rightarrow$ revert is needed every 30 minutes

Host

- HW: 8 cores, 12 GB memory
- OS: VMware vSphere Hypervisor 4.1 (former “free ESXi”)
- free (as in beer), quick setup, well known
- lots of HOWTOs about automation (needed for revert)

Network

- separate SSIDs → separate VLANs
- ideal solution: separate VLANs for machine groups too
- time-saver solution we used this year:
  - single VLAN
  - L3 separation (firewall)
  - a lot can go wrong
  - pwned machines can be used to circumvent L3 separation
- potential pitfall: scoreboard availability

Automated revert a'la vSphere

- we need revert every 30 minutes, vSphere has automation, well done
- Java application can read everything needed

Automated revert a'la vSphere

- we need revert every 30 minutes, vSphere has automation, well done
- Java application can read everything needed
- `com.vmware.vim25.RestrictedVersion` exception
- the remote interface of the free version is read only

Automated revert a'la vSphere

- we need revert every 30 minutes, vSphere has automation, well done
- Java application can read everything needed
- `com.vmware.vim25.RestrictedVersion` exception
- the remote interface of the free version is read only
- solution: enable SSH (Google is my friend), `vim-cmd vmsvc/snapshot.revert`
- revert uses an ID which we haven't been able to found using any kind of GUI so far (only `vim-cmd` reports it)
- vSphere SSHd can only execute a single revert per session
- success of revert is easy to check (ROOT)

The script

- chosen language: Python (see `import antigravity`)
- libs used:

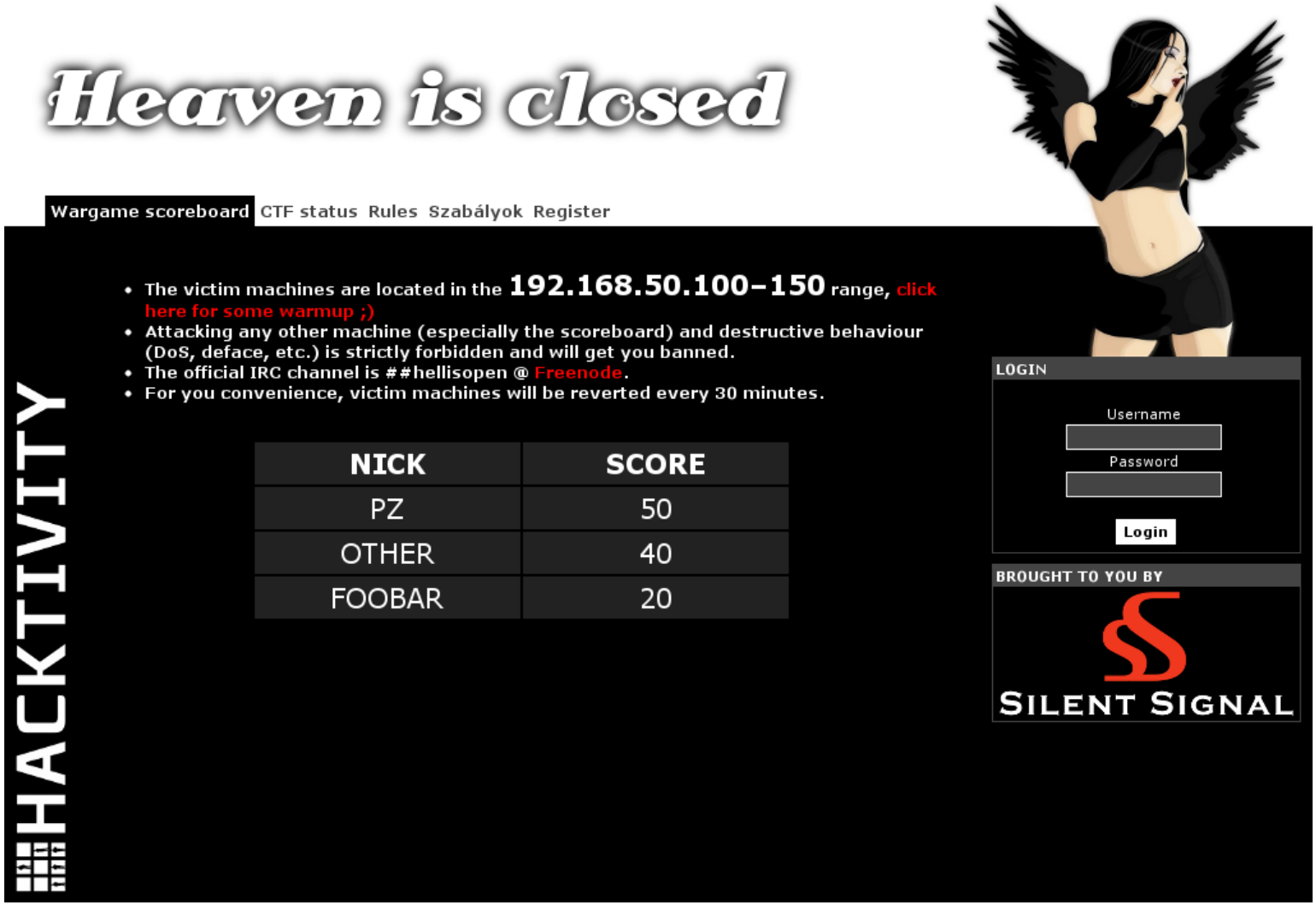
libssh2	revert and proof upload (SCP)
mysqldb	DB backend (proofs and hosts)

- started by cron every 30 minutes
- reverts hosts
- uploads proofs (waits for SSHd to start)
- updates rewards (CTF only)
- pubkey login is not (yet) implemented in Python libssh2

Scoreboard

- trivial at first sight, easy to fsck up
- should have access to machines and proofs (DB)
- needs to be reachable from every network
- needs to be secure
  - 31337 d00dz d150b3y r00l5
  - DRY: there are great frameworks for secure ORM
- provides registration, login, submission, board
- nuances: separate view for beamers, auto refresh

Scoreboard screenshot



Lessons learnt

- organizing a wargame is fun
- balancing between under- and overestimating the audience is crucial, although very hard (impossible?)
- players have a tendency to ignore the rules
  - especially those related to scoreboard hacking
- network reliability hugely impacts user experience
- running VMWare Workstation inside ESXi is far from being trivial (other direction is OK)
- "Good artists copy, great artists steal" – Pablo Picasso

Thanks for your attention!