The Only Way To Be Sure:
Obtaining and Detecting Domain Persistence

Grant Bugher
http://perimetergrid.com
DEFCON 101 Track @ DEFCON 23
The research and opinions presented in this talk are my own.
They do not necessarily represent those of my employer.
Who am I?

◊ Hacking and coding since the early 90’s

◊ Working professionally in information security for the last 10 years
  ◊ Developer, security tester, program manager, security engineer, security architect, consultant, a bit of everything
  ◊ Worked on IT, developer tools, programming languages & class libraries, online services, high-security datacenters, application security consulting, SIEM deployment, retail systems
  ◊ Currently a security engineer for a major cloud service
  ◊ Also own Perimeter Grid, security blog & consulting service

◊ Prior speaker at BlackHat USA (2010) and DEF CON (22) and a regular DEF CON attendee since DEF CON 16.
So You Have a Domain Controller

◊ State of monitoring in real enterprises is generally woeful
  ◊ Local event logs with default configurations
  ◊ SIEM designed for compliance, not security and forensics

◊ Basic Monitoring
  ◊ Detailed, granular auditing enabled in Group Policy
  ◊ Event logs pushed or pulled to an SIEM, off the servers and ideally inaccessible to them
  ◊ Centralized host intrusion detection/anti-malware
  ◊ Process start command line auditing and PowerShell auditing enabled in Group Policy
Demo Domain Configuration

- Windows Azure Virtual Network
- Three servers & a workstation:
  - pg-dc: Windows 2008 SP1 Domain Controller
  - pg-website: Windows 2008 SP1 Web Server with ASP & ASP.NET
  - pg-monitor: Windows 2008 SP1 running Splunk Enterprise and collecting logs
  - pg-workstation: Windows 7 SP1 workstation used by the hapless attachment-clicker Bobert.
- Splunk Enterprise runs as a domain user
  - Pulls non-DC logs via WMI
  - DC pushes logs via Splunk Universal Forwarder (so the monitoring account isn’t a Domain Admin)
- Symantec Endpoint Monitoring on all systems, forwarding to Splunk via event log
Something Extra

◊ Process start command line logging & PowerShell logging enabled on all systems
◊ SysMon (SysInternals Monitoring service) installed and configured on all systems
  ◦ Logs process creation with full command line for both current and parent processes.
  ◦ Records the hash of process image files using SHA1 (the default), MD5, SHA256 or IMPHASH.
  ◦ Includes a process GUID in process create events to allow for correlation of events even when Windows reuses process IDs.
  ◦ Include a session GUID in each events to allow correlation of events on same logon session.
◊ Logs loading of drivers or DLLs with their signatures and hashes.
◊ Optionally logs network connections, including each connection’s source process, IP addresses, port numbers, hostnames and port names.
◊ Detects changes in file creation time to understand when a file was really created. Modification of file create timestamps is a technique commonly used by malware to cover its tracks.
So You Want a Domain Controller

◊ Many ways to compromise an AD domain...
  ◊ Get an admin’s password via keylogger
  ◊ Get an admin to click on your malware attachment
  ◊ Steal an AD backup (NTDS.DIT, etc.)
  ◊ Exploit unpatched servers
  ◊ Exploit security software or other privileged services
  ◊ Use your l33t 0-days

◊ This is not what this talk is about
  ◊ On the bright side, you’re at DEF CON, so it’s what a lot of other talks are about!
Domain Persistence

◊ We’re just going to stipulate you have momentarily compromised the domain.
  ◊ You have TCP/IP network access to the domain: a PwnPlug or compromised device inside
  ◊ You have a Meterpreter session with a Domain Admin token: Perhaps they insecurely stored a PowerShell script that the Domain Admin runs on the primary DC
  ◊ Doesn’t matter where you got this; that’s not what the talk is about
◊ The administrators are going to notice you compromised the domain and try to remediate – that is, kick you out – promptly.
  ◊ Our goal: make it easy to re-escalate to Domain Admin using only our TCP/IP network access
  ◊ Their goal: figure out how to kick us out without nuking the entire site from orbit
Demos, Demos, Demos!

- Creating a new Domain Admin account (you might also try banging a gong)
- Backdoor an administrator’s workstation (login scripts, scheduled tasks, autoruns, BHOs, DLL load order hijack)
- Trojan administrative tools (and add your own CAs so they’re signed!)
- Crack hashes, steal PKI keys
- Obtain the Golden Ticket
- Skeleton Key LSASS
- Set PowerShell as a debugger to something important
- Stupid Built-In Group Tricks (overwrite sensitive object ACL templates)
- Hiding administrative privileges in SID history or changing support account RIDs
- Make the typical pentest path easy (create privileged application users, remove patches)
Detection and Remediation

- *All* of these techniques leave traces in the Event Log or in ActiveDirectory
- But an attacker can disable event retrieval/forwarding and purge the Event Log
  - Any system with a purged Event Log is hopelessly compromised and must be rebuilt
  - Yes, this sucks when it’s the primary domain controller
- Of course you need to change the compromised passwords
  - But also *every* password due to possible hash theft... even service accounts... and KRBTGT
  - And a full audit of *every* AD change since compromise for things like group membership and SID history changes
- Don’t have a full AD change history, or the time to go through it?
  - Nuke the entire site from orbit... it’s the only way to be sure.
Questions?

Updated Slides with Screenshots at http://perimetergrid.com/DefCon23.pptx