1. Click on the link below, this will open your web browser

http://aucache.autodesk.com/social/visualization.html

2. Use “Extended Display” to project the website on screen if you plan to work on your computer. Use “Duplicate” to display same image on screen and computer.
Security Best Practices for AutoCAD Users, CAD Managers and Developers

Davis Augustine and Albert Szilvasy
Autodesk, Inc. AutoCAD Software Development
FBI warns of 'destructive' malware in wake of Sony attack

BOSTON | Mon Dec 1, 2014 6:34pm EST
We will present effective security practices for AutoCAD software developers, CAD managers, and advanced AutoCAD users. Straight from members of the AutoCAD Security Team. Concepts include types of vulnerabilities and threats, existing AutoCAD software security features, and recommended system settings and administration. For native C++ development, we’ll also briefly cover secure build switches, APIs and testing strategies. We focus on Windows but many of the same issues apply to the Mac.
Key learning objectives

At the end of this class, you will be able to:

- Understand various kinds of attacks against AutoCAD users
- Configure AutoCAD to protect against attacks
- Use safe practices when downloading and installing apps
- Configure Windows to defend against attacks
- Make your native C++ applications more secure
1. Intro – the attackers, what they do, what we do
2. Three Kinds of Attacks – how they work, how to defend
3. Possible Future AutoCAD Security Feature Changes
4. Security Tips for Native C++ Developers
5. Call to Action, Q & A
Security is Trending…

- Both AU sessions filled up
- Adesk internal tech summits have had exponential growth in number of security sessions
Who would want to attack your system?

- Recreational hackers
- Malicious hackers
- Extortionists
- Organized crime
- Corporate competitors
- Nation-States
Random vs. Targeted vs. APT Attacks

- Random attacks are more common
- Targeted may do more damage
- **Advanced Persistent Threat** is the most dangerous
  - An extreme targeted attack
  - Involves Social Engineering
  - Customized exploits, careful planning
Possible Malware Actions

- Amuse or annoy you
- Damage data or system
- Ransom data
- Steal intellectual property
- Install keyloggers, steal credentials
- Launch attacks on other systems or devices
AutoCAD Security Feature Team

- We don’t want to be the vector for penetration of a system
- Raise the bar – make attackers go after easier targets
- Natural tension between security team and feature teams
- Similar to tension betw CAD Managers/Sys Admins and Users
Learning Objective 1: Understand 3 Kinds of Threats to Desktop App Users

- File planting
  - Malware module is unintentionally downloaded and executed
- Unsigned or Mistakenly Trusted Apps
  - Malware is intentionally run by user
- Native code injection
  - Malware code carried inside data file
Three Kinds of Threats to Desktop App Users

- **File planting**
  - Malware module is unintentionally downloaded and executed

- Unsigned or Mistakenly Trusted Apps
  - Malware is intentionally run by user

- Native code injection
  - Malware code carried inside data file
Acad Executable File Types

- arx, crx, dbx, dll, exe (PE files)
- fas, lsp, vlx (AutoLISP)
- cui, cuix, mnl (U/I files)
- dvb (Visual Basic)
- hdi (plotter drivers, PE)
- pgp (command aliases)
- scr (acad script)
- acad.rx (list of startup apps)
Folders and Paths of Interest

- Current (working)
- Drawing (location of current dwg)
- Start-In (initial working folder)
- Exe (location of acad.exe)
- Support (list of support folders)
- Project (list of project folders)
Acad (2015) Search Rules for Lisp and ARX Apps

1. Start-In folder (“startinfolder” sysvar)
2. Drawing folder (“dwgprefix” sysvar)
3. Support path (1\textsuperscript{st} part of “acadprefix”)
4. Driver path (2\textsuperscript{nd} part of “acadprefix”)
5. Exe folder (vla-get-path (vlax-get- acad-object))
… Which Leads to a “File Planting” Vulnerability

1. Start-In folder
2. Drawing folder
3. Support path
4. Driver path
5. Exe folder

Danger, Will Robinson!!
“Planted” Auto-Loaded File in Start-In/Dwg Folder

- Planted file loads instead of the one in the support path
- Exploit can be done with any auto-loaded file
- Malware can run other payloads, access system, copy itself to other folders, etc
- Original “bursted” virus worked this way
Defenses Against File Planting

- Acad 2014 added **TrustedPaths** sysvar
- Future acad may remove **Start-In** and **Drawing** folders from executable file search rules
Windows DLL Search Rules Changed After XP

1. Directory from which the application loaded
2. **Current directory** (moved to #5 after XP)
3. System directory (See `GetSystemDirectory`)
4. 16-bit system directory
5. Windows directory (See `GetWindowsDirectory`)
6. Directories in the PATH environment variable

Current directory search can also be disabled altogether
Three Kinds of Threats to Desktop App Users

- **File planting**
  - Malware module is unintentionally downloaded and executed
- **Unsigned or Mistakenly Trusted Apps**
  - Malware is intentionally run by user
- **Native code injection**
  - Malware code carried inside data file
Mistakenly Trusted Apps

- Naïve users run things they shouldn’t
- Sophisticated users can fall prey to social engineering attacks
- Once app gets control, possible damage depends on account rights
Standard User Rights and UAC Limit System Damage

- UAC more important
- Protects system data and integrity, but *not* user data
- CIP data show a lot of users running as admin without UAC!
- Support calls show many users not up to date with anti-virus
Using Group Policy to Restrict Installs

- Windows gpedit tool
- Doesn’t prevent other methods of installing
Digital Signature Guarantees “Chain of Custody”

- Future acad versions may warn more about unsigned apps
- No guarantee that app is safe
- User must decide whether to trust publisher and certificate authority
Signing ARX Apps (and Maybe LISP in the Future)

- SignTool.exe for arx apps
- Signed arx apps not on trusted paths can load
- We might extend this to lisp in future
Possible Future Missing Signature Warning

File Loading - Security Concern

An unsigned executable file was found outside of the trusted locations. What do you want to do?

Name: test.lsp

Location: C:\Users\varghg\AppData\Local\Autodesk\testing\testing2\testing4\testing5\testing6\testing7

Make sure this file comes from a trusted source and does not contain malicious code.
Possible Future Invalid Certificate Warning

File Loading - Invalid Certificate

An executable with an invalid certificate was found outside of the trusted locations. What do you want to do?

Name: FeaturedAppsPlugin.dll

Location: C:\ProgramData\Autodesk\ApplicationPlugins\Autodesk
FeaturedApps.bundle\Contents\Windows\2015\Win64

An invalid certificate could mean that the certificate was revoked or that the file has been tampered with. We recommend that you do not load this file.

Load  Do Not Load  Help
Other Executable File Types

- .cui, .cuix, .scr, .mnl cannot (yet) be signed
- Best to keep those in read-only folders maintained by admins
- Some day probably nearly all executable files will be signed
Three Kinds of Threats to Desktop App Users

- File planting
  - Malware module is unintentionally downloaded and executed
- Unsigned or Mistakenly Trusted Apps
  - Malware is intentionally run by user
- **Native code injection**
  - *Malware code carried inside data file*
When Data Attack!

- Input files are crafted to cause buffer overflow
- Buffer can be in stack, heap or global memory
- Foreign code may be included in the data
- Or “ROP” code can take advantage of existing code fragments in the app
Managed code detects buffer overflows
Working exploits against acad have been reported by researchers (we fixed them)
Defenses Against Buffer Overflow Attacks

- Enable Data Execution Prevention (DEP) in Windows and BIOS
- SEHOP – enable in registry
- MSoft EMET tool for protecting specific apps
Other Windows Mitigations

- See support.microsoft.com/fixit
- Disable Remote Registry Service
Learning Objective 2: Understand Other Possible Acad Security-Related Changes
Dwg Password Feature

- Added in 2004 – but didn’t keep up with technology
- Fairly easily cracked – so we may drop it
Open a Dwg File Directly from a ZIP File

- Avoids risk of unzipping all the zip file’s contents into local folder

- Xrefs and other support files also loaded from zip file (but not executables!)

- Not clear how to save back to zip file
Locking Acad Sysvars

- Until now, sysvars kept in HKCU
- In future, we may allow them to be overridden in HKLM
MSVC Build Switches For Security

- /GS and /SDL to guard against buffer overflow
- /NxCompat: Data execution prevention
- /DynamicBase: DLL load addresses less predictable (ASLR)
Secure APIs Guard Against Buffer Overflow

- wcsncpy(wchar_t *, const wchar_t *) is unsafe!
- wcsncpy_s(wchar_t *, size_t nLen, const wchar_t *) is safe
- Arx APIs may change to also take a buffer size argument
- Your APIs should also always take a size arg for destination buffers
- Microsoft banned.h header finds unsafe calls
# Microsoft BinScope Tool

- Free tool for checking on binaries’ build switches

## Checks

<table>
<thead>
<tr>
<th>Check</th>
<th>SDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATL header version check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>ATL vulnerability check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>Function Pointers Check</td>
<td></td>
</tr>
<tr>
<td>Shared sections check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>APTCA Check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>NXCOMPAT check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>/GS check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>/GS function level checks - don't disable optimizations</td>
<td>(SDL)</td>
</tr>
<tr>
<td>/GS function level checks - don't declare functions __declspec(safebuffers)</td>
<td>(SDL)</td>
</tr>
<tr>
<td>Compiler version check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>SAFESEH check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>GS-friendly initialization check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>Strong name check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>/DYNAMICBASE check</td>
<td>(SDL)</td>
</tr>
<tr>
<td>VB6 check</td>
<td>(SDL)</td>
</tr>
</tbody>
</table>
Static Analyzers

- HP Fortify
- MS Visual Studio
- KlocWork
- SonarQube
- CppCat
- CppCheck
- VeraCode
Fuzz Testing Finds Real Bugs

- Fuzzer creates “munged” input files, is automatable
- Finds uncovered code paths and states
- Is also used by hackers to find vulnerabilities
Thank You, and Be Careful Out There!

Contact us at: autocad.security@autodesk.com
Join the beta: www.autodesk.com/autocadbeta

Q & A...
Session Feedback

- Via the Survey Stations, email or mobile device
- AU 2015 passes given out each day!
- Best to do it right after the session
- Instructors see results in real-time
Students, educators, and schools now have **FREE** access to Autodesk design software & apps.

Download at [www.autodesk.com/education](http://www.autodesk.com/education)
Earn your professional Autodesk Certification at AU

Visit the AU Certification Lab