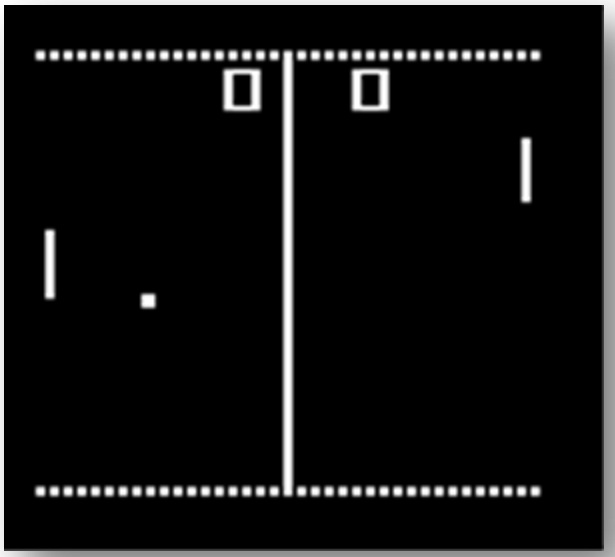


Gaming - The Next Overlooked Security Hole

Ferdinand Schober

Overview

- Overview
- Historical development
- Know thy gamer
- Know thy developer
- Know thy engine
- Profit?
  - Virtual Economies
- Current malware
- Games 2.0 & Privacy
- Exercise in Exploits
  - The little nude patch that could
  - View my post and get owned
  - The ad from hell





Clarifications

- In the context of this talk:
  - Games = PC games
    - No video games (don't think about consoles!)
    - Dominant OS: Windows
  - Non-casual games
    - 'Hard-core' games
    - Looking at the game client
    - Not genre-specific
  - No web-based games
    - Some issues are shared, but this is not the focus here
  - Limited view at online games
    - Think of your generic MMO (client yet again)



Historical development

Games (late 1990ies)



Games (2008)





Historical development

Games (late 1990ies)

Non-mainstream

- 'Geeks play games'
- Estimated budget:
  2-3M US$
  1-3 years
- Self-publishing common, but declining

Games (2008)

Mass-market applications

- GTA4 review in the NY Times?
- Estimated budget:
  15-20M US$
  2-3 years
- ~3 major publishers provide funding
  - EA being biggest
  - Homogenization



Historical development

Games (late 1990ies)

Custom graphics solutions

- Most games still implement their own graphics engine
- Only DirectX/OpenGL shared



Games (2008)

Full-featured engines

- Limited group of engines (~3) used by most games, number is dropping
- Includes physics, scripting, audio, AI
- One exploit covers multiple games!



Historical development

Games (late 1990ies)

Middleware is limited

- In-game physics not really feasible yet (but getting there)
- Mostly custom solutions



Games (2008)

Middleware is standard

- Engines provide features, rest is done through middleware
- Same as before!



Historical development

Games (late 1990ies)

Physical media is king

- CDs/DVDs are the distribution media

Games (2008)

Push for shared platforms that require online presence (Games 2.0)

- Steam, GfW Live
- Used for distribution and content protection
- Platform also used for multiplayer
- Automatic patching



Historical development

Games (late 1990ies)

Offline games

- Only very few MMOs and they are not really there yet
- Multiplayer game modes available
- Direct connections - no common platform



Games (2008)

Online is default

- MMOs are a mass market (WoW!)
- Online presence becoming fully integrated through platform
- I can see what you play!



Historical development

Games (late 1990ies)

Little custom content

- Editors considered a 'goodie' to keep the game going
- Very limited group of people were producing content



Games (2008)

Custom content is part of feature set

- Editors shipped almost by default
- Custom content is expected - the next big thing
- XMLification of content
- Custom content is automatically pulled into the game



Historical development

Games (late 1990ies)

Community through sites/boards

- Mostly web/chat based
- Not integrated into game

Games (2008)

Built-in community features

- Based in the common online platform
- Available in any game
- Again: I can see what you play!

What did not change?

- Developers are pressed for time/money
  - Time is spent on 'making it pretty' – 'pretty sells'
  - Products need to use more middleware
    - Canned code cannot be fully reviewed, inherits issues
  - Security generally not a major concern
    - Favorite quote: 'It's just a game!'
- Release games are not that stable
  - Crashes do not raise suspicion
- Patching is common
  - But mostly for gameplay features
- Hacks/Cracks/Trainers are readily accessible
  - Just google it!
  - Do you really trust the serial generator?
- Piracy
  - Push towards alternatives (now: online distribution or authentication platforms) for the wrong reasons



Do we see the problem?

It doesn't look pretty so far!



Know thy gamer

- PC gamers
  - Generally more hard-core than the console gamer
    - Higher learning curve to get a game running
    - Install nightmares, configuration issues, etc…
  - Know the OS and hardware fairly well
    - Need to know about drivers and configuration
    - Higher-end hardware required
    - Changes OS settings to get games running (faster)
      - Experienced gamers will disable anything (!) to get more performance
        - Yes, that means everything that uses CPU cycles
  - Use PC as multi-purpose system
    - Not only for gaming like consoles
    - Used for web browsing, data storage, etc…
    - System has plenty of personal information



Know thy gamer

- PC gamers
  - PC gamers are not paranoid
    - Gamers are used to crashes and erratic behavior
    - Used to frequent patching
    - Will use custom content as long as it is 'pretty'
      - Generally custom content is trusted ('What harm can a new model do?')
  - Games need to be run with highest privileges
    - Games can do everything that the admin can do
    - Slow shift towards more privilege security (due to Windows Vista)
  - Most gamers spend a lot of time online
    - Due to the nature of the games MMOs, multiplayer games, …
    - Also for community activities (boards, …)
    - Distribution platform might require it (e.g. Steam)



Know thy gamer

- PC gamers
  - Given a choice they will take performance or ‘fancy’ hardware over security





Know thy dev

- Game developers
  - Are like most other developers
    - Will make the same mistakes (we are all human)

  And:

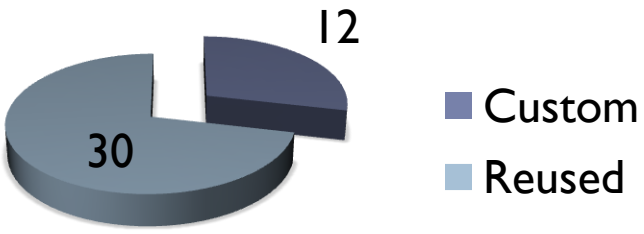
  - Are under severe time pressure
    - Hard deadlines (has to make holiday season, ship date)
    - Most games run late
    - Need to use canned code
  - Love latest and greatest (aka. 'shiny complex')
    - Does not help with schedule and testing
    - Quick design = quick exploits
    - New features = new bugs



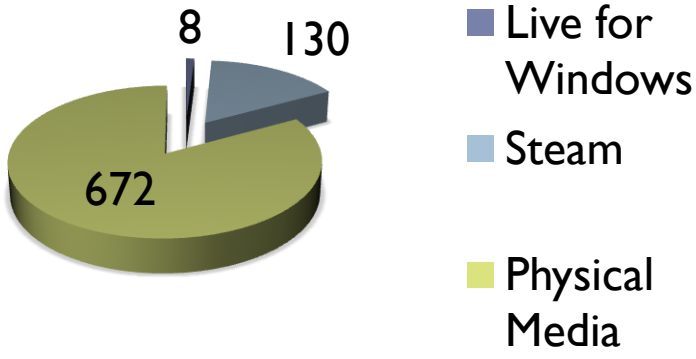
Know thy engine

- ~810 PC games released in 2007*
  - 42 games considered major selling games
  - Results:
    - Still multiple contenders for graphics engine
    - Not so for physics engines

Graphics Engine 'Major Sellers'

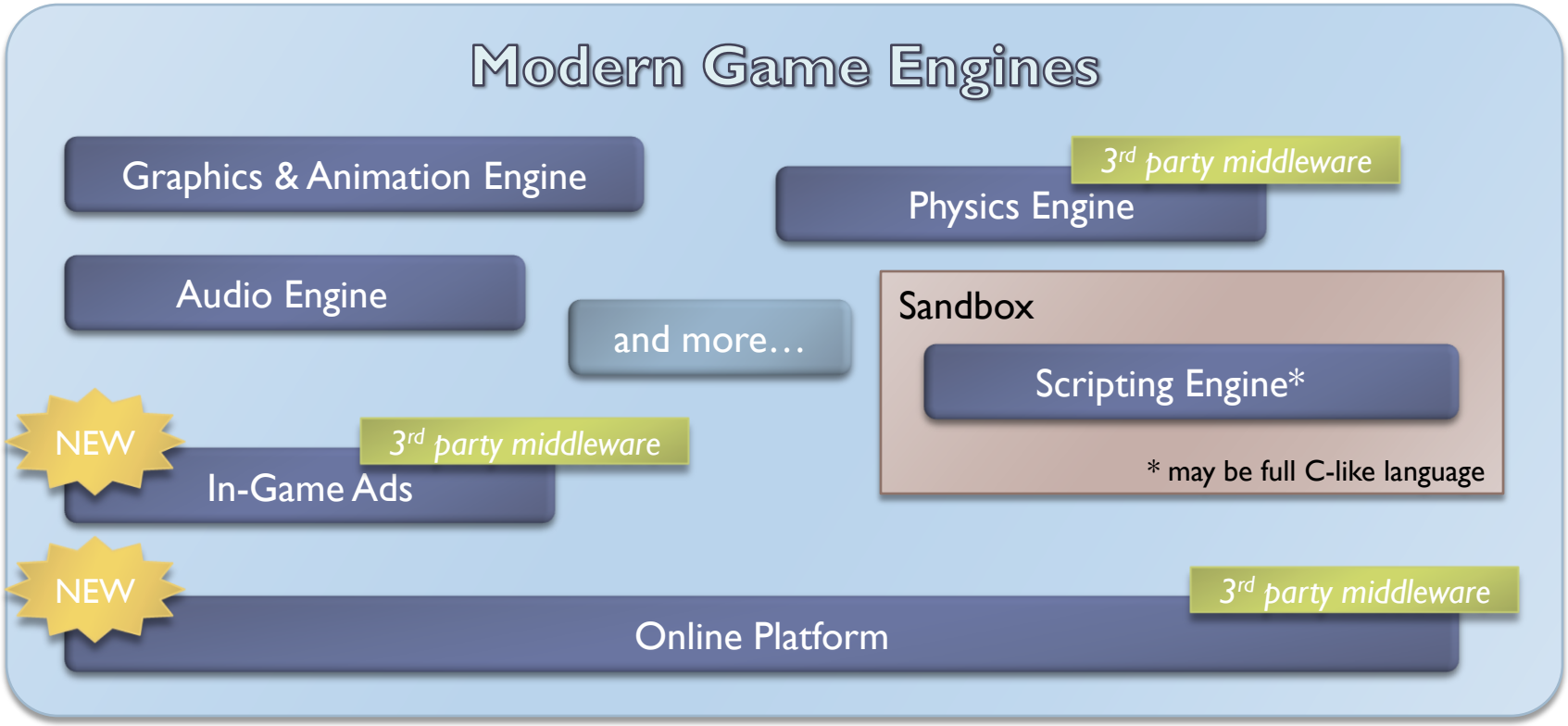


Distribution Platform total



* based on Wikipedia

Know thy engine



- Engine fixes by users are rarely shared
  - Fixes from other engine users are usually not shared
  - Fixes are custom for game/developer



Know thy engine

- Engine/platform versions easy to trace
  - Engine binaries are easy to locate
  - Binaries provide version numbers
- Shared engines provide easy exploits
  - Issue in one game becomes issue for all games with the same engine binaries
  - Customization of engine actually might help
- Can't we patch the engine?
  - Historically games are patched, not engines
  - Engine developers do not roll out patches, game developers do
    - Might in some cases be the same developer
  - Customization doesn't help engine patch process
  - Automatic patching can be spoofed too



Profit?

It's just a game, there is nothing to gain, right?

Wrong!



Profit?

Game Side

- Griefing/Cheating
- Personal information
- Payment information
  - Think platform/MMO
- Existing virtual assets

System Side

- Pretty much everything you get from hacking a system

  - and -

- Systems of gamers are usually 'beefier'
  - More high-end then average systems (= more CPU cycles)
  - Broadband network connection
  - Overall: Good staging system



Virtual Economies

- Virtual economies have large user bases
  - Very gamer is a member and spends money
  - Currently 14+M gamers active in MMOs alone
  - User base is huge target for current malware
- Virtual economies now create significant revenue
  - Traditionally through in-game assets
  - Also through services (gold-farming, auto-leveling, …)
  - Revenue is in real money!
- Some games are built purely on in-game micro-transactions
  - Even easier to gain real money from digital assets



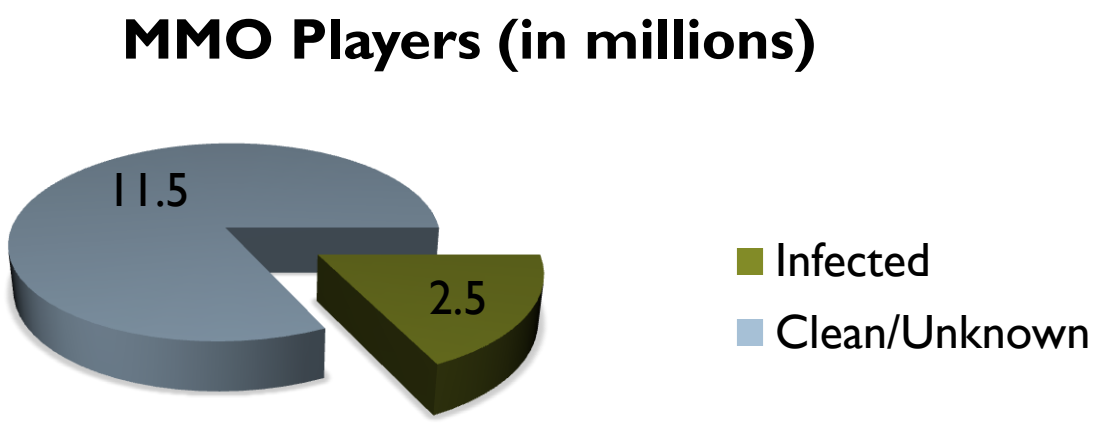
Virtual Economies

- Any kind of exploit can result in quick gains for the attacker
  - Stealing assets from legitimate player
  - Selling assets produced through exploits
  - Leveraging the player's account
  - Payment information is available too…
- Exploits need to be quickly fixed on server/client side
  - Slow fixes can cause whole in-game economy to crash
  - There is a lot at stake here!



Current malware

- 'Account stealers'
  - Targeted at acquiring account credentials for MMOs
  - Various different families of malware
    - Most common: Win32/Taterf
    - Top 8 malware families detected on ~2.5M systems in June*
      (~ 18% of the total user base)



* based on MS Malware Protection Center

Current malware

- MMOs are most common target for malware in the game segment
- Propagation mostly through community
  - Browser exploits through community sites
  - Through unofficial patches/tools
  - As part of social engineering



Current malware



Games 2.0 & Privacy

- Games 2.0: Fully integrated online games or games that use online platforms
  - Examples: MMOs, Steam, GfW Live

- Privacy concerns
  - System knows when game is run, required to 'unlock' game
  - Online status is published
    - Can be hidden, but is still known underneath the hood
    - Future features can show location as well (think: mobile status)
  - Games that have been played and progress are visible
    - Think 'achievements' and beyond
    - Ad systems keep even more detailed track of gamers
  - List of friends is very easy to obtain
  - Basically everything a social platform has – and more
  - Hard to get around this if you want to play any game, especially online



Exercise in Exploits

Case Studies



- The little nude patch that could
- View my post and get owned
- The ad from hell



The little nude patch that could

- *<Engine Exploit>*
- Scenario:
  - Alice plays game but is not happy with the scene on the right (for purely academic reasons)
  - Alice finds a nude patch provided by Bob at *nudepatchworld.com**.
  - Alice downloads and reviews files
    - Patch adds new character file (NudeAlexis.chr) and replaces references in game configuration to point to the new content
    - Alice considers changes harmless (no executable in package)
  - Alice installs patch and enjoys new content
  - Meanwhile, Bob enjoys Alice's credit limit



* Does not exist as of August 2008, why not?

The little nude patch that could

- What happened?
  - Let's look at the new content in detail:



  - Wait? Script code?
    - This is the problem
    - Crafted so it gets executed when model is loaded for the first time
    - Executed in script engine with game permissions
      - Usually Administrator, remember?



The little nude patch that could



- Exploiting the script code
  - Script code runs in sandbox
    - Nothing bad can happen, right?
  - Script engines are highly complex, finding a flaw is just a matter of time
    - Grab your favorite fuzzer and go
    - Sufficient flaw allows system access with game permissions
    - Flaw can be reused in multiple approaches
    - Flaw probably works with other games if scripting engine is shared
    - Might crash the game, but gamers are used to that (sadly)
  - Once in the system with Administrative access, everything is lost



View my post and get owned

<Social Engineering>



- Scenario:
  - Alice is playing a popular MMO (> 9 mil. Users) and likes to share information with other players through the game's message board
  - Bob posts a question and a little flash tag with it
  - Alice views Bob's post and responds
  - Next time Alice logs into the game all her items are gone
  - Meanwhile Bob sells Alice's items on Ebay and uses her account for more posts



View my post and get owned

- What happened?
  - Pretty straight-forward:
    - Bob exploits a known Flash vulnerability to get access to Alice's machine and obtain her account credentials
    - Bob isn't interested in anything else (for now)
    - Single board attack can yield hundreds of accounts for Bob



  - Even if game is patched and secure, web browser might not be
  - Securing games is not everything
    - Community locations need to be locked-down as well
    - Gamers need to be educated properly
    - Sometimes these locations are outside the reach of developers



The ad from hell

- *<Middleware Exploit>*
- Scenario:
  - Alice plays her favorite game that contains new in-game ads
  - Ads are provided through new middleware
  - Bob uploads a custom image file to the ad system
    - Through spoofing server data, gaining access to servers or submitting it as content
  - Alice (and all other gamers) experience a crash when they view this image in the game
  - Meanwhile, all machines are belong to Bob





The ad from hell

- What happened?
  - Image exploited a flaw in the display engine of the ad system
    - One flaw covers all gamers
    - User can only prevent this attack by not playing the game
  - Exploit in in-game advertising can put the whole cloud at risk
    - Server might need to be breached, but servers can also be spoofed
    - Data submitted to the system needs to be sanitized
  - In-game advertising solutions have significantly more logic than just rendering
    - Who, when, where, what, for how long?
    - More code increases attack surface



Questions?

