Hacking and securing MSBuild

Jan Krivanek

Materials

dotutils.net/wug-talk

For long lived linking: dotutils.net/wug-talk-securing-msbuild

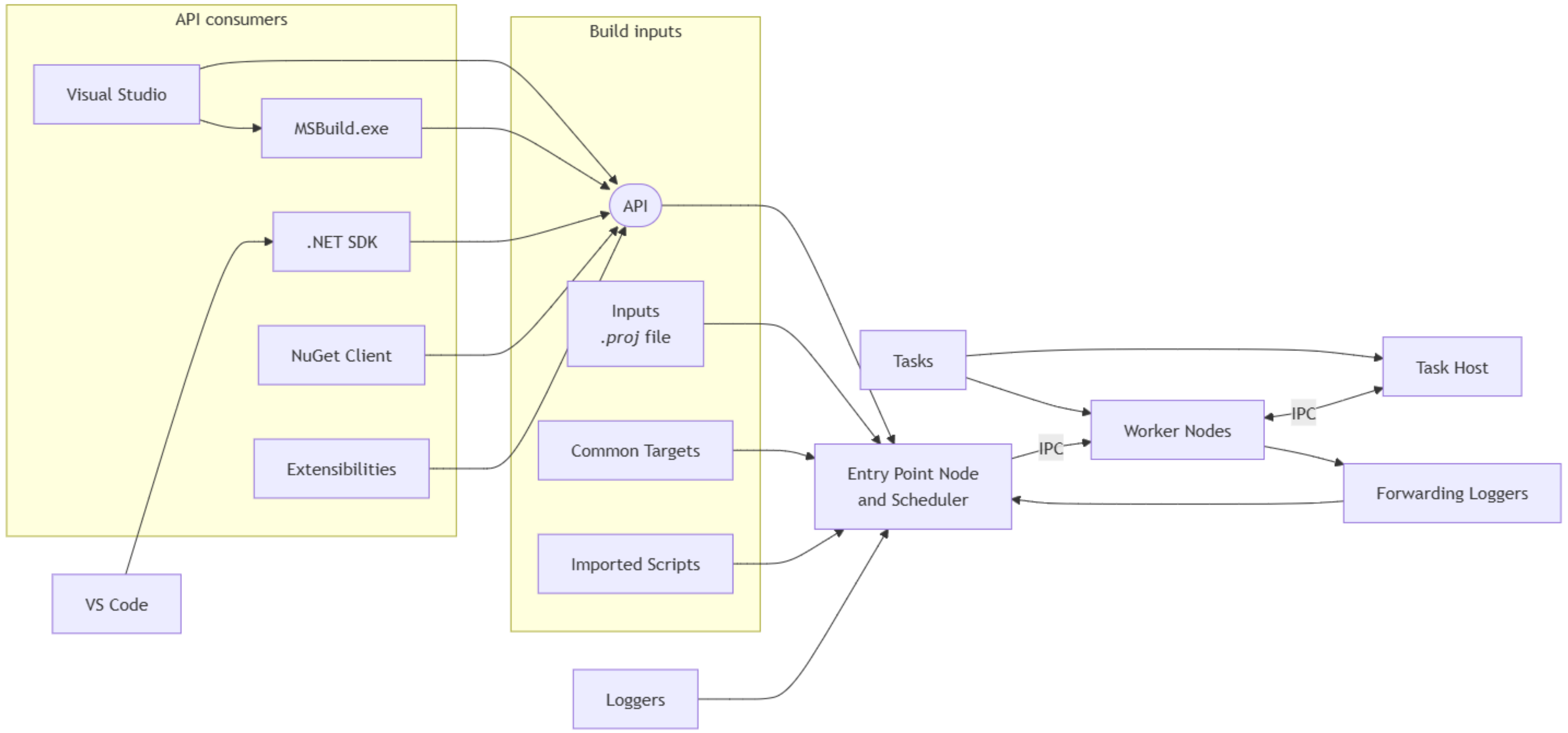
Agenda

- MSBuild intro
- Extendibility intro
- Use cases of extending and prevention

What is MSBuild?

- Engine – performs the build
  - Exposed via CLI and API
- Language – allows to define the build
  - 4 entities: Properties, Items, Targets, Tasks
- Common Targets – predefined behavior of the build
  - Shipped with SDK and VS

What is MSBuild?



What can be extended?

- Engine
  - Environment variables
  - Discovered Plugins
  - API/CLI injected plugins
  - .rsp files
  - Scripts injected plugins
- Language
  - Custom tasks
- Common Targets
  - Auto-discovered imports (Directory.Build.[props|targets])
  - Generated imports (.g.[props|targets])
  - User imports (<Import Project="…">)
  - 'magic' properties

Sample of code execution (Language extending)

- Custom task (RoslynCodeTaskFactory, AssemblyTaskFactory)
- Custom task factory
- Exec (UNC)
- DownloadFile
- Redefine common target
- Design Time Build!

- Prevention: Operate only on trusted code
- Configure Trust Settings
- Currently no ability to disable custom or certain tasks 🙁

Engine extending – Environment variables

- Important Recognized Environment Variables
- Traits
- ChangeWaves
- Recognizing environment variables as implicit properties

- Prevention: BC0103 BuildCheck

Engine extending – Discovered Plugins

- SdkResolvers folder in sdk/VS installation
- SDK
- ProjectCachePlugin

- Prevention: Install MSBuild to location with controlled R/W access

Engine Extending – CLI, .rsp

- Loggers
- Magic properties
  - DependsOn
  - PreBuildEvent
- Response file
  - MSBuild.rsp
  - Directory.Build.rsp

- Prevention:
  - Environment variables checking
  - -noAutoResponse
  - Installing MSBuild into location with restricted R/W access

Common Targets Extending - .props|.targets

- Directory.Build.[props|targets]
  - dotutils.net/build-helpers/Directory.Build.targets
  - Danger of Downloads
- .user file
- [before|after].{solution}.targets
- Bonus: .suo

- Prevention:
  - Empty Directory.Build.[props|targets]
  - Review files in project dir

Common Targets Extending - .props|.targets

- MSBuildExtensionsPath
- MSBuildUserExtensionsPath
- Those are properties! (CLI, .rsp, env vars – all apply here)

- Prevention: $(ImportUserLocationsByWildcardBefore{ImportingFileNameWithNoDots})
- dotnet build /p:ImportUserLocationsByWildcardBeforeMicrosoftCommonProps=false

Common Targets Extending – nuget.g.[props|targets]

- Build logic auto-included from build
  - Build buildTransitive buildMultitargeting
- Visible in binlog

- Prevention: ExcludeAssets

Summary

- Custom Tasks
- Common Targets redefine, Common Targets hooking
- DownloadFile, Exec (incl from UNC)
- Logic inject via CLI, Env, .rsp files
- Logic inject via Directory.Build.[props|targets]
- Logic inject via MSBuild[User]ExtensionsPath (incl custom location)
- Logic inject from .user, .suo, [before|after].{solution}.targets files
- Logic inject from nugets

Thank you!

Questions

Jan Křivánek
Microsoft
Jan.Krivanek@microsoft.com