C2 SERVER INFILTRATION AND FORENSIC ANALYSIS

Thabiso Mashifana (21)

TECHNICAL REPORT

Velociraptor Query Language:

Windows.System.TaskSchedular

The Windows task scheduler is a common mechanism that malware uses for persistence. It can be used to run arbitrary programs at a later time. Commonly malware installs a scheduled task to run itself periodically to achieve persistence.

This artifact enumerates all the task jobs (which are XML files). The artifact uploads the original XML files and then analyses them to provide an overview of the commands executed and the user under which they will be run.

Source Analysis

```
LET Uploads = SELECT Name, OSPath, if(
     condition=AlsoUpload='Y',
     then=upload(file=OSPath)) as Upload
  FROM glob(globs=TasksPath)
  WHERE NOT IsDir

// Job files contain invalid XML which confuses the parser - we
// use regex to remove the invalid tags.
LET parse_task = select OSPath, parse_xml(
         accessor='data',
         file=regex_replace(
              source=utf16(string=Data),
              re='<[?].+?>',
              replace='')) AS XML
  FROM read_file(filenames=OSPath)

SELECT OSPath,
      XML.Task.Actions.Exec.Command as Command,
      XML.Task.Actions.Exec.Arguments as Arguments,
      XML.Task.Actions.ComHandler.ClassId as ComHandler,
      XML.Task.Principals.Principal.UserId as UserId,
      XML as _XML
  FROM foreach(row=Uploads, query=parse_task)

```

Windows.System.PsList

List process and their running binaries

Source

```
LET ProcList = SELECT * FROM if(condition=UseTracker,
then={
  SELECT Pid, Ppid, NULL AS TokenIsElevated,
         Username, Name, CommandLine, Exe, NULL AS Memory
  FROM process_tracker_pslist()
}, else={
  SELECT * FROM pslist()
})

SELECT Pid, Ppid, TokenIsElevated, Name, CommandLine, Exe,
    token(pid=Pid) as TokenInfo,
    hash(path=Exe) as Hash,
    authenticode(filename=Exe) AS Authenticode,
    Username, Memory.WorkingSetSize AS WorkingSetSize
FROM ProcList
WHERE Name =~ ProcessRegex
    AND Pid =~ PidRegex
    AND Exe =~ ExePathRegex
    AND CommandLine =~ CommandLineRegex
    AND Username =~ UsernameRegex
    AND NOT if(condition= UntrustedAuthenticode,
               then= Authenticode.Trusted = 'trusted' OR NOT Exe,
               else= False )

```

Windows.Registry.UserAssist

Windows systems maintain a set of keys in the registry database (UserAssist keys) to keep track of programs that are executed. The number of executions and last execution date and time are available in these keys.

The information within the binary UserAssist values contains only statistical data on the applications launched by the user via Windows Explorer. Programs launched via the commandline (cmd.exe) do not appear in these registry key

Exports

```
LET userAssistProfile = ''' [ ["Header", 0, [ ["NumberOfExecutions", 4, "uint32"], ["LastExecution", 60, "uint64"] ]] ] '''
```

Source

```
LET TMP = SELECT OSPath.Path AS _KeyPath,
    parse_string_with_regex(
         string=OSPath.Path,
         regex="^.+Count\\\\\"?(?P<Name>.+?)\"?$") AS Name,
      OSPath,
      parse_binary(
         filename=Data.value,
         accessor="data",
         profile=userAssistProfile,
         struct="Header"
       ) As ParsedUserAssist,
       Username AS User
FROM Artifact.Windows.Registry.NTUser(KeyGlob=UserAssistKey)

LET UserAssist = SELECT _KeyPath,
    if(condition=Name.Name,
       then=rot13(string=Name.Name),
       else=OSPath.Path) AS Name,
    User,
    timestamp(winfiletime=ParsedUserAssist.LastExecution) As LastExecution,
    timestamp(winfiletime=ParsedUserAssist.LastExecution).Unix AS LastExecutionTS,
    ParsedUserAssist.NumberOfExecutions AS NumberOfExecutions
  FROM TMP
  ORDER BY LastExecution
LET A1 = SELECT * FROM if(
    condition=UserFilter,
    then={
      SELECT * FROM UserAssist WHERE User =~ UserFilter
    },
    else={ SELECT * FROM UserAssist})

SELECT * FROM if(
    condition=ExecutionTimeAfter,
    then={
      SELECT * FROM A1 WHERE LastExecutionTS > ExecutionTimeAfter
    },
    else={ SELECT * FROM A1})

```