Analysis Report for: 90941E12A1CF4B58440151C7785EEE36.au3

Overall Functionality

This C code is a malicious AutoIt script designed to download and execute arbitrary code from remote servers. It disguises its actions by using seemingly innocuous function names and obfuscation techniques. The script fetches a list of URLs, retrieves data from these URLs, parses the data for instructions, and then executes those instructions which involve downloading and running executables or PowerShell scripts. The script attempts to hide its activity by running downloaded files in various ways (directly, via `ShellExecute`, using AutoIt's built-in script execution, or PowerShell) and reporting back to the C2 server on the success or failure of these operations. The use of randomly generated filenames and temporary directories adds to the obfuscation.

Function Summaries

* **`_HexToString($shex)`:** Converts a hexadecimal string (prefixed with "0x" or not) to a UTF-8 string. Parameters: `$shex` (hexadecimal string). Return value: UTF-8 string or an error code.

* **`_StringBetween($sstring, $sstart, $send, $imode = $str_endisstart, $bcase = False)`:** Extracts a substring between two specified delimiters using regular expressions. Parameters: `$sstring` (input string), `$sstart` (start delimiter), `$send` (end delimiter), `$imode` (matching mode), `$bcase` (case-sensitive flag). Return value: Array of matched substrings or an error code.

* **`_StringExplode($sstring, $sdelimiter, $ilimit = 0x0)`:** Splits a string into an array based on a delimiter. Parameters: `$sstring` (input string), `$sdelimiter` (delimiter), `$ilimit` (limit on number of splits). Return value: Array of substrings.

* **`_StringInsert($sstring, $sinsertion, $iposition)`:** Inserts a string into another string at a specified position. Parameters: `$sstring` (input string), `$sinsertion` (string to insert), `$iposition` (insertion position). Return value: Modified string or an error code.

* **`_StringProper($sstring)`:** Converts a string to proper case (e.g., "hello world" to "Hello World"). Parameter: `$sstring` (input string). Return value: Proper-cased string.

* **`_StringRepeat($sstring, $irepeatcount)`:** Repeats a string a specified number of times. Parameters: `$sstring` (input string), `$irepeatcount` (number of repetitions). Return value: Repeated string or an error code.

* **`_STRINGTITLECASE($sstring)`:** Converts a string to title case. Parameter: `$sstring` (input string). Return value: Title-cased string.

* **`_StringToHex($sstring)`:** Converts a UTF-8 string to a hexadecimal string. Parameter: `$sstring` (input string). Return value: Hexadecimal string.

* **`UPDATING()`:** The main function. It sets a window title, sleeps for a random time, fetches URLs containing commands, parses them, downloads and executes files based on those commands.

* **`__HTTP_ONERROR(ByRef $omyerror)`:** An error handler for HTTP requests. Parameter: `$omyerror` (error object). Return value: Error code.

* **`REPORTOK($surl, $staskid, $ssucc)`:** Reports the success or failure of task execution to a remote server. Parameters: `$surl` (base URL), `$staskid` (task ID), `$ssucc` (success code). Return value: None.

* **`HEXING($strhex)`:** Converts a hexadecimal string to a binary string and removes null characters. Parameter: `$strhex` (hexadecimal string). Return value: Binary string (with nulls removed).

* **`_HTTPGET($surl)`:** Performs an HTTP GET request. Parameter: `$surl` (URL). Return value: HTTP response text or an error code.

* **`_DOWNLOADFILE($surl, $sdata)`:** Downloads a file from a URL and saves it to a specified path. Parameters: `$surl` (URL), `$sdata` (filepath). Return value: An error code.

* **`_BASE64ENCODE($sdata)`:** Encodes data to Base64. Parameter: `$sdata` (data to encode). Return value: Base64 encoded string.

* **`_SFIRSTTIME()`:** Checks a registry key to see if the script has run before. Return value: String indicating first run or not.

* **`_MAKESTRING()`:** Generates a random alphanumeric string. Return value: Random string.

* **`_SECUREDOWNLOAD($surl, $sext)`:** Downloads a file from a URL to a random location in several Microsoft directories. Parameters: `$surl` (URL), `$sext` (file extension). Return value: Path to the downloaded file or an empty string.

Control Flow

The `UPDATING()` function is the main control flow. It iterates through a list of URLs. For each URL, it retrieves data, checks for specific markers (e.g., ".DOW", ".DAW", ".DAH", ".DAE", ".DAC", ".PS"), and executes commands based on these markers. Each marker type dictates a different action, typically involving downloading and executing a file (`.exe`, AutoIt script, or PowerShell script). The code uses nested loops and conditional statements (`If`, `Select Case`) to manage the different command types and handle execution errors. Error handling is minimal and mostly involves reporting back to the C2 server.

Data Structures

The primary data structures are arrays:

* `$arrurls`: An array of URLs to fetch commands from.
* `$arrlines`: An array of lines extracted from the HTTP response.
* `$arrtask`: An array of task details (ID and data) parsed from each line.
* `$arr`: Array of directories to search for potentially malicious files.

Malware Family Suggestion

Based on its functionality, this AutoIt script strongly suggests a **downloader/dropper** type of malware, possibly belonging to a larger malware family that utilizes modular commands. The script acts as an intermediary, fetching instructions and payloads from a command-and-control (C2) server and executing them on the infected machine. The variety of execution methods (direct execution, `ShellExecute`, AutoIt script execution, PowerShell) and the obfuscation techniques indicate an attempt to evade detection and analysis. The use of Base64 encoding would further help to hide the true nature of the downloaded payloads. The reporting mechanism to the C2 server suggests this is part of a larger botnet or malware operation. The behavior resembles that of many information-stealing or remote access trojans (RATs).