# ARPSweep COM Component

## General information

| | |
|---|---|
| Functionality: | Utilize ARP protocol to scan the network |
| Developer: | Constantine V. Sharlaimov (jShadow) |
| Project version: | 1.0.0 (build 156) |
| Project status: | release candidate |
| Dependencies: | WinPCap |

## Properties

`Adapter` (integer) – this is a number of network adapter in WinPCap adapter list. Note: the list may differ when retrieving it with COM component and when retrieving it with user-mode application. Usage of adapter name and `LookupAdapterName` method is a safe method to specify a network adapter.

`AdapterCount` (integer, read-only) – a total number of network adapters in WinPCap adapter list. It can be used to verify the adapter number before setting the `Adapter` property. It can also be used with `GetAdapterInfo` method.

`StartIP` (string) and `EndIP` (string) – IP address (in dotted notation) defining the range to be scanned. Note: values are not verified to satisfy `StartIP` <= `EndIP`.

`HostCount` (integer, read-only) – number of addresses in range defined with `StartIP` and `EndIP`. If one of the addresses is not valid or `StartIP <= EndIP` is not satisfied, value of `HostCount` is zero.

`ScannedHostCount` (integer, read-only) – property value indicates a number of already scanned hosts. Property value is valid only after `OnScanStart` and before next scan starts.

## Methods

`GetAdapterInfo([in] adapter as integer, [out] name as string, [out] description as string, [out] ip as string, [out] mac as string) as boolean`. The method returns WinPCap name, readable description, IP-address and MAC-address of specified adapter. If the function fails, return value is `false`, otherwise it is `true`. Note: if the network adapter has multiple IP addresses assigned, the first IP address is returned. If the method is unable to determine MAC address, the value of FF-FF-FF-FF-FF will be returned.

`StartScan()` – this method starts scanning the specified IP range (`StartIP` and `EndIP` properties) using the network adapter specified by the `Adapter` property. Method returns immediately after starting a separate scanning thread. Note: `OnScanHost` events are raised in context of the scanning thread.

`Scan([in] adapter as integer, [in] ip as string, [in] mac as string) as integer`. The method performs an "ARP ping" determining if a host with the specified IP and MAC is alive. The return value is the response time (in ms). Note: the method will fail if a background scan is started (using `StartScan` method). Valid MAC address formats are XX-XX-XX-XX-XX-XX and XX:XX:XX:XX:XX:XX (both uppercase and lowercase are valid).

`StopScan()` – this method stops background scanning (started with `StartScan`). The `OnScanStop` event is raised after scan is stopped.

`LookupAdapterName([in] adapter_name as string) as integer`. This function is designed to make interaction with this COM component easier. It will search for an adapter name in WinPCap adapter list and return its number in that list. This is safer way to select an adapter, for item order in the adapter list may differ in the COM component and user-mode application.

## Events

`OnScanHost([in] ip as string, [in] mac as string, [in] response_time as integer)`. This event is raised by background scanning thread when the component receives an ARP response to previously sent ARP request. Note: there can be multiple `OnScanHost` events with same IP and different MAC addresses (in case of some host doing network snooping). The situation when multiple `OnScanHost` events with same IP and MAC are raised can also appear when native TCP/IP protocol stack is performing a concurrent ARP request to a host, currently being scanned. This may change in release version.

`OnScanStart()` and `OnScanStop()` – these two events signal a start and end of a background scan. `OnScanStart` is triggered by `StartScan` call, but appear asynchronously, so `OnScanStart` can be raised before as well as after `StartScan` method returns.

`OnScanError(code as integer)` – this event is raised in case of any component's fatal or non-fatal internal error. `OnScanError` always appears synchronously with setting properties or calling component's methods.

## Error codes

`E_INVALID_STARTIP = 1` – the value of `StartIP` property is invalid (appears when setting StartIP property or calling StartScan method).

`E_INVALID_ENDIP = 2` – the value of `EndIP` property is invalid (appears when setting EndIP property or calling StartScan method).

`E_INVALID_ADAPTER = 3` – the value of `Adapter` property is invalid (appears when setting Adapter property or calling StartScan method).

`E_SCAN_IN_PROGRESS = 4` – a background scan is already in progress. This error appears when calling `StartScan` or `Scan` methods.

`E_WINPCAP_NOT_FOUND = 5` – the component can not load WinPCap library.

`E_COULD_NOT_OPEN_ADAPTER = 6` – the WinPCap failed to open the specified adapter. WinPCap being called from a COM component sometimes behaves strangely and fails to open some network adapters (during testing only non-Ethernet ones: dialup connections etc).

`E_INVALID_IP = 7` – this error may appear only in `Scan` method when bad IP address was specified.

`E_INVALID_MAC = 8` – this error appears only in `Scan` method when provided MAC address is invalid.

# Testing environment

The ARPSweep component was tested on a Windows XP (with and without SP1 and SP2). The following remote equipment and software was used in testing process:

- D-Link DI-604 broadband router;
- Allied Telesyn AT-8326GB and AT-8350GB managed switches;
- Realtek, Allied Telesyn, Intel, D-Link, Planet, 3COM and other Ethernet network interface cards;
- D-Link AirPlus DWL-520+ WiFi adapter (AdHoc mode);
- Windows 2000 (Pro, Server, Advanced server), Windows XP (SP1, SP2), Windows 2003 (SP1), Linux kernel (both 2.4 and 2.6);
- ZoneAlaram, Outpost, internal Windows XP and Linux kernel firewalls.

The component scanning core determines MAC addresses without any noticeable problems. Firewalls have no effect on ARP scanning (exactly as expected, for ARP is a vital component of TCP/IP stack for Ethernet networks).