Rapid7_Nexpose_Assets template

Template	Comments
`{ "version": "2.0", "name": "Rapid7 Nexpose Assets management", "comment": "", "type": "REST_EVENT", "event_type": ["LEASE", "NETWORK_IPV4", "RANGE_IPV4", "FIXED_ADDRESS_IPV4", "HOST_ADDRESS_IPV4", "NETWORK_IPV6", "RANGE_IPV6", "FIXED_ADDRESS_IPV6", "HOST_ADDRESS_IPV6"], "action_type": "Rapid7 Nexpose Assets management", "content_type": "text/xml", "vendor_identifier": "Rapid7", "quoting": "XMLA",`	"version" must be set to "2.0" This template can be used with LEASE, NETWORK_IPV4, RANGE_IPV4, FIXED_ADDRESS_IPV4, HOST_ADDRESS_IPV4, NETWORK_IPV6,RANGE_IPV6, FIXED_ADDRESS_IPV6, and HOST_ADDRESS_IPV6 events/notifications. XMLA quoting is used by default.
`{ "name": "defaultValues", "operation": "NOP", "body": "${XC:ASSIGN:{L:IPTo}:{S:}}${XC:ASSIGN:{L:Hostname}:{S:}}" },`	Set default values for the variables: **IPTo** - is used for last IP in a range or a network **Hostname** - an asset's hostname
`{ "name": "checkEType_Network", "operation": "CONDITION", "condition": { "condition_type": "AND", "statements": [ {"left": "${E::event_type}", "op": "==", "right": "LEASE"} ], "next": "checkEType_Lease" } },`	If it is LEASE event jump to checkEType_Lease step
`{ "name": "skip if Site is not defined or sync not requested", "operation": "CONDITION", "condition": { "statements": [{"left": "${E:A:values{extattrs}{R7_Site}{value}}", "op": "==", "right": ""}, {"left": "${E:A:values{extattrs}{R7_Sync}{value}}", "op": "==", "right": ""}, {"left": "${E:A:values{extattrs}{R7_Sync}{value}}", "op": "==", "right": "false"}], "condition_type": "OR",`	Stop if **R7_Site** attribute is not set or **R7_Sync** is not exists or set to false

```
        "stop": true
      }
    },
```

```
    {
      "name": "skip synced host",
      "operation": "CONDITION",
      "condition": {
        "condition_type": "AND",
        "statements": [
          {"left": "${E:A:operation_type}", "op": "==", "right": "INSERT"},
          {"left": "${E:A:values{extattrs}{R7_SyncedAt}{value}}", "op": "!=", "right":
""}
        ],
        "stop": true
      }
    },
```

Stop if the operation is INSERT and **R7_SyncedAt** not empty (the object was synced before, e.g. restored from a trash bin). This step can be removed if it is not a desired behaviour.

```
    {
        "name": "assignLVarsNet",
        "operation": "NOP",
        "body_list": [
            "${XC:COPY:{L:Site}:{E:values{extattrs}{R7_Site}{value}}}",

"${XC:COPY:{L:ScanTemplate}:{E:values{extattrs}{R7_ScanTemplate}{value}}}
",

"${XC:COPY:{L:ScanOnAdd}:{E:values{extattrs}{R7_ScanOnAdd}{value}}}",
            "${XC:COPY:{L:Obj_ref}:{E:values{_ref}}}",
            "${XC:ASSIGN:{L:SaveEA}:{S:true}}"
        ]
    },
```

Set local variables from the extensible attributes:
Site - Site name

ScanTemplate - a template used for scanning assets

ScanOnAdd - request to scan the asset

Obj_ref - object reference in NIOS

SaveEA - defines if extensible attributes values can/should be updated in NIOS

```
    {
        "name": "SetR7_IPF_Network",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
              {"left": "${E::event_type}", "op": "==", "right": "NETWORK_IPV4"},
              {"left": "${E::event_type}", "op": "==", "right": "NETWORK_IPV6"}
            ],
            "eval":
"${XC:COPY:{L:Network}:{E:values{network}}}${XC:NETWORKTORANGE:{L:N
etwork}:{L:RangeFromNet}}${XC:ASSIGN:{L:ObjType}:{S:NETWORK}}${XC:C
OPY:{L:IPFrom}:{L:RangeFromNet{{from}}}}${XC:COPY:{L:IPTo}:{L:RangeFro
mNet{{to}}}}"
        }
    },

    {
        "name": "SetR7_IPF_Range",
```

Set local variables based on a created object type and extensible attributes:
Network - Network

RangeFromNet - contains a range in Rapid7 Nexpose format

ObjType - object type (e.g. NETWORK, RANGE, HOST, FIXEDIP)

IPFrom - an IP address of host/fixed IP/lease/reservation or first IP address in a

```
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
              {"left": "${E::event_type}", "op": "==", "right": "RANGE_IPV4"},
              {"left": "${E::event_type}", "op": "==", "right": "RANGE_IPV6"}
            ],
            "eval":
"${XC:COPY:{L:IPFrom}:{E:values{start_addr}}}${XC:COPY:{L:IPTo}:{E:values{
end_addr}}}${XC:ASSIGN:{L:ObjType}:{S:RANGE}}"
        }
    },

    {
        "name": "SetR7_IPF_Host_IPv4",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
              {"left": "${E::event_type}", "op": "==", "right":
"HOST_ADDRESS_IPV4"}
            ],
            "eval":
"${XC:COPY:{L:IPFrom}:{E:values{ipv4addr}}}${XC:COPY:{L:Hostname}:{E:val
ues{host}}}${XC:ASSIGN:{L:IPv}:{S:ipv4addr}}${XC:ASSIGN:{L:ObjType}:{S:H
OST}}"
        }
    },

    {
        "name": "SetR7_IPF_Host_IPv6",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
              {"left": "${E::event_type}", "op": "==", "right":
"HOST_ADDRESS_IPV6"}
            ],
            "eval":
"${XC:COPY:{L:IPFrom}:{E:values{ipv6addr}}}${XC:COPY:{L:Hostname}:{E:val
ues{host}}}${XC:ASSIGN:{L:IPv}:{S:ipv6addr}}${XC:ASSIGN:{L:ObjType}:{S:H
OST}}"
        }
    },

    {
        "name": "SetR7_IPF_Fixed_IPv4",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
              {"left": "${E::event_type}", "op": "==", "right":
"FIXED_ADDRESS_IPV4"}
            ],
            "eval":
"${XC:COPY:{L:IPFrom}:{E:values{ipv4addr}}}${XC:ASSIGN:{L:ObjType}:{S:FI
XEDIP}}"
        }
    },
```

network/range

IPTo - last IP-address in a network/range, contains an empty value for other object types

IPv - ipv4addr or ipv6addr

NetToSite - defines if a network should be added to defined assets

RangeToSite - defines if a range should be added to defined assets

AddByHostname - defines if a host should be added by a hostname

SiteID - Rapid7 Nexpose Site ID

```
    {
        "name": "SetR7_IPF_Fixed_IPv6",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
              {"left": "${E::event_type}", "op": "==", "right":
"FIXED_ADDRESS_IPV6"}
            ],
            "eval":
"${XC:COPY:{L:IPFrom}:{E:values{ipv6addr}}}${XC:ASSIGN:{L:ObjType}:{S:FI
XEDIP}}"
        }
    },

    {
        "name": "SetR7_NetToSite",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
                {"left": "${E:A:values{extattrs}{R7_NetToSite}{value}}", "op": "==",
"right": ""}
            ],
            "eval": "${XC:ASSIGN:{L:NetToSite}:{S:false}}",
            "else_eval":
"${XC:COPY:{L:NetToSite}:{E:values{extattrs}{R7_NetToSite}{value}}}"
        }
    },

    {
        "name": "SetR7_RangeToSite",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
                {"left": "${E:A:values{extattrs}{R7_RangeToSite}{value}}", "op": "==",
"right": ""}
            ],
            "eval": "${XC:ASSIGN:{L:RangeToSite}:{S:false}}",
            "else_eval":
"${XC:COPY:{L:RangeToSite}:{E:values{extattrs}{R7_RangeToSite}{value}}}"
        }
    },

    {
        "name": "SetR7_AddByHostname",
        "operation": "CONDITION",
        "condition": {
            "condition_type": "OR",
            "statements": [
                {
                    "left": "${E:A:values{extattrs}{R7_AddByHostname}{value}}",
                    "op": "==",
                    "right": ""
                }
            ],
            "eval": "${XC:ASSIGN:{L:AddByHostname}:{S:false}}",
          "else_eval":
"${XC:COPY:{L:AddByHostname}:{E:values{extattrs}{R7_AddByHostname}{val
ue}}}"
       }
   },

   {
       "name": "SetR7_SiteID",
       "operation": "CONDITION",
       "condition": {
          "condition_type": "OR",
          "statements": [
             {"left": "${E:A:values{extattrs}{R7_SiteID}{value}}", "op": "==", "right":
""}
          ],
          "eval": "${XC:ASSIGN:{L:SiteID}:{I:0}}",
          "else_eval":
"${XC:COPY:{L:SiteID}:{E:values{extattrs}{R7_SiteID}{value}}}"
       }
   },
```

```
   {
     "name": "findRef_Host",
     "operation": "CONDITION",
     "condition": {
       "condition_type": "AND",
       "statements": [
          {"left": "${L::ObjType}","op": "!=","right": "HOST"}
       ],
       "next": "Fin_Vars_Init"
     }
   },

   {
     "name": "findRef_Host_ch_Delete",
     "operation": "CONDITION",
     "condition": {
       "condition_type": "AND",
       "statements": [
          {"left": "${E:A:operation_type}", "op": "==", "right": "DELETE"}
       ],
       "next": "Fin_Vars_Init"
     }
   },

   {
      "name": "Get Host _ref",
      "operation": "GET",
      "transport": {"path":
"record:host?_return_fields=name,extattrs&network_view=${E::values{network
_view}}&name=${L::Hostname}&${L::IPv}=${L::IPFrom}"},
      "wapi": "v2.6"
   },

   {
      "operation": "CONDITION",
      "name": "wapi_response_get_ref",
      "condition": {
         "statements": [
```

If object type not equal HOST jump to **Fin_Vars_Init** step.

HOST events are triggered per IP address so if a host has 3 ip addresses 3 events will be triggered (for each IP-address) and **_ref** field in the event contains a reference to record:host_ipv4addr object. Extensible attributes can be saved only on a host level (record:host).

These steps retrieve a host's _ref attribute and save it in **Obj_ref** variable.

```
         {
           "op": "==",
           "right": "${P:A:PARSE[0]{_ref}}",
           "left": ""
         }],
       "condition_type": "AND",
       "error": true,
       "else_eval": "${XC:COPY:{L:Obj_ref}:{P:PARSE[0]{_ref}}}"
     }
   },

   {
     "name": "check if host already synced",
     "operation": "CONDITION",
     "condition": {
       "statements": [
         {"left": "${P:A:PARSE[0]{extattrs}{R7_SyncedAt}}", "op": "!=", "right": ""}
       ],
       "condition_type": "AND",
       "stop": true
     }
   },
```

```
   {
    "name": "checkEType_Lease",
    "operation": "CONDITION",
    "condition": {
     "condition_type": "AND",
     "statements": [
        {"left": "${E::event_type}", "op": "!=","right": "LEASE"}
     ],
     "next": "Fin_Vars_Init"
    }
   },

   {
     "name": "skip if not defined for lease",
     "operation": "CONDITION",
     "condition": {
       "statements": [
         {"left": "${E:A:ip.extattrs{R7_Site}}", "op": "==", "right": ""},
         {"left": "${E:A:ip.extattrs{R7_Sync}}", "op": "==", "right": ""},
         {"left": "${E:A:ip.extattrs{R7_Sync}}", "op": "==", "right": "false"}
       ],
       "condition_type": "OR",
       "stop": true
     }
   },

   {
     "name": "assignLVarsLease",
     "operation": "NOP",
     "body_list": [
        "${XC:COPY:{L:Network}:{E:values{network}}}",
        "${XC:COPY:{L:IPFrom}:{E:values{address}}}",
        "${XC:COPY:{L:Site}:{E:ip.extattrs{R7_Site}}}",
        "${XC:COPY:{L:Sync}:{E:ip.extattrs{R7_Sync}}}",
```

Set local variables for LEASE event.

We need to distinguish leases and other objects because of the different event variables are used.

```
          "${XC:COPY:{L:ScanTemplate}:{E:ip.extattrs{R7_ScanTemplate}}}",
          "${XC:COPY:{L:ScanOnAdd}:{E:ip.extattrs{R7_ScanOnAdd}}}",
          "${XC:COPY:{L:Hostname}:{E:values{client_hostname}}}",
          "${XC:ASSIGN:{L:SaveEA}:{S:false}}",
          "${XC:ASSIGN:{L:ObjType}:{S:LEASE}}"
        ]
    },

    {
      "name": "SetR7_L_SiteID",
      "operation": "CONDITION",
      "condition": {
         "condition_type": "OR",
         "statements": [
            {"left": "${E:A:ip.extattrs{R7_SiteID}}", "op": "==", "right": ""}
         ],
         "eval": "${XC:ASSIGN:{L:SiteID}:{I:0}}",
         "else_eval": "${XC:COPY:{L:SiteID}:{E:ip.extattrs{R7_SiteID}}}"
      }
    },
```

```
    {
     "name": "Fin_Vars_Init",
     "operation": "NOP",
     "body": "${XC:DEBUG:{L:}}"
    },


    {
      "name": "handle delete",
      "operation": "CONDITION",
      "condition": {
        "statements": [{"left": "DELETE", "op": "==", "right":
"${E:A:operation_type}"}],
        "condition_type": "AND",
        "next": "DeleteObject"
      }
    },
```

If object was deleted jump to DeleteObject step

```
    {
      "name": "Check SiteID",
      "operation": "CONDITION",
      "condition": {
         "condition_type": "AND",
         "statements": [
            {"left": "${L:A:SiteID}", "op": "!=", "right": "0"}
         ],
         "next": "GetSiteConf"
      }
    },
```

If **SiteID** is defined jump to GetSiteConf

```
    {
      "name": "Request R7 sites",
      "parse": "XMLA",
      "operation": "POST",
      "body_list": [
        "<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
        "<SiteListingRequest session-id=\"${S::SESSID}\" />"
      ]
```

The code (from this step to “GetSiteConf”) is executed if R7_SiteID attribute was not set and it tries to determinate **SiteID** base on **Site** name

```
    },

    {
      "name": "Check sites request on errors",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "SiteListingResponse", "op": "!=","right":
"${P:A:PARSE[[name]]}"},
          {"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
        ],
        "condition_type": "AND",
        "else_eval": "${XC:COPY:{L:site_list}:{P:PARSE}}",
        "error": true
      }
    },

    {
      "name": "Check if sites list is empty",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "${L:L:site_list}", "op": "==","right": "0"}
        ],
        "condition_type": "AND",
        "stop": true
      }
    },

    {
       "name": "Pop site from the list",
       "operation": "VARIABLEOP",
       "variable_ops": [
         {
           "operation": "POP",
           "type": "COMPOSITE",
           "destination": "L:a_site",
           "source": "L:site_list"
         }
      ]
    },

    {
      "name": "check_a_site",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "${L:A:Site}", "op": "!=", "right": "${L:A:a_site{{name}}}"}
        ],
        "condition_type": "AND",
        "next": "Check if sites list is empty",
        "else_eval": "${XC:COPY:{L:SiteID}:{L:a_site{{id}}}}"
      }
    },

    {
     "name": "checkSaveSiteID",
     "operation": "CONDITION",
     "condition": {
```

SiteListingRequest is used to retrieve a list of sites from Rapid 7 Nexpose. Session is identified by a S:SESSID variable.

In a loop a single value is retrieved from the list and compared with the **Site** attribute.
If the Site was found and **SaveEA** set to true SiteID attribute saved in R7_SiteID attribute and the template jumps to “GetSiteConf”.

```
      "condition_type": "AND",
      "statements": [
          {"left": "${L::SaveEA}", "op": "!=", "right": "true"}
      ],
      "next": "GetSiteConf"
    }
  },

  {
     "name": "Update SiteID",
     "operation": "PUT",
     "transport": {"path": "${L:A:Obj_ref}"},
     "wapi": "v2.6",
     "wapi_quoting": "JSON",
      "body_list": [
          "{",
          "\"extattrs+\":{\"R7_SiteID\": { \"value\": \"${L:A:SiteID}\"}}",
          "}"
      ]
  },
```

```
  {
    "name": "GetSiteConf",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "${L:A:ObjType}", "op": "==", "right": "NETWORK"},
          {"left": "${L:A:NetToSite}", "op": "!=", "right": "true"}
      ],
      "stop": true
    }
  },

  {
    "name": "CheckSyncRanges",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "${L:A:ObjType}", "op": "==", "right": "RANGE"},
          {"left": "${L:A:RangeToSite}", "op": "!=", "right": "true"}
      ],
      "stop": true
    }
  },
```

Stop if a Network or a Range should not be synchronized with Rapid7 Nexpose

```
  {
    "name": "GetSiteConf_R7",
    "parse": "XMLA",
    "operation": "POST",
    "body_list": [
       "<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
       "<SiteConfigRequest session-id=\"${S::SESSID}\"
site-id=\"${L:A:SiteID}\"/>"
    ]
  },

  {
```

Retrieve a site configuration

```
      "name": "get_site_config(errorcheck)",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "SiteConfigResponse", "op": "!=", "right":
"${P:A:PARSE[[name]]}"},
          {"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
        ],
        "condition_type": "OR",
        "else_eval":
"${XC:COPY:{L:SiteConfig}:{P:PARSE{SiteConfigResponse}}}",
        "error": true
      }
    },
```

```
    {
      "name": "add by host name",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "${L:A:Hostname}", "op": "==", "right": ""},
          {"left": "${L:A:ObjType}", "op": "!=", "right": "HOST"},
          {"left": "${L:A:AddByHostname}", "op": "==", "right": "false"}
        ],
        "condition_type": "OR",
        "next": "Net_to_Site_conf"
      }
    },

    {
      "name": "Hostname_to_Site_conf",
      "operation": "VARIABLEOP",
      "variable_ops": [
        {
          "operation": "PUSH",
          "type": "COMPOSITE",
          "name": "host",
          "destination": "L:SiteConfig{Site}{Hosts}",
          "composite_value": "${L:A:Hostname}"
        }
      ]
    },

    {
      "name": "save by hostname",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"right": "1", "op": "==", "left": "1"}
        ],
        "condition_type": "OR",
        "next": "Save site config"
      }
    },
```
	Add a host by hostname (if it was requested and hostname is not empty) into the Site configuration

```
    {
      "name": "Net_to_Site_conf",
      "operation": "CONDITION",
      "condition": {
```
	Add a network into the Site configuration

"condition_type": "AND",
"statements": [
{"left": "${L:A:ObjType}", "op": "!=", "right": "NETWORK"}
],
"next": "Other_to_Site_conf"
}
},
 	
{
"name": "Push_Network_to_Site_conf",
"operation": "VARIABLEOP",
"variable_ops": [
{
"operation": "PUSH",
"type": "COMPOSITE",
"name": "host",
"destination": "L:SiteConfig{Site}{Hosts}",
"source": "L:RangeFromNet"
}
]
},
 	
{
"name": "save network to site",
"operation": "CONDITION",
"condition": {
"statements": [
{"right": "1", "op": "==", "left": "1"}
],
"condition_type": "OR",
"next": "Save site config"
}
},	

{
"name": "Other_to_Site_conf",
"operation": "VARIABLEOP",
"variable_ops": [
{
"operation": "PUSH",
"type": "COMPOSITE",
"name": "range",
"keys": ["from","to"],
"destination": "L:SiteConfig{Site}{Hosts}",
"composite_value": "",
"values": ["${L:A:IPFrom}","${L:A:IPTo}"]
}
]
},
	Add FixedIP, Lease, Host by IP, Range in the Site configuration

{ "name": "Save site config", "parse": "XMLA", "operation": "POST", "body_list": ["<?xml version=\"1.0\" encoding=\"UTF-8\"?>", "<SiteSaveRequest session-id=\"${S::SESSID}\">", "${L:x:SiteConfig}",	Save Site configuration, raise an error in case of any issues

```
      "</SiteSaveRequest>"
    ]
  },
  {
    "name": "update_site(errorcheck)",
    "operation": "CONDITION",
    "condition": {
      "statements": [
        {
          "op": "!=",
          "right": "${P:A:PARSE[[name]]}",
          "left": "SiteSaveResponse"
        },
        {
          "op": "!=",
          "right": "1",
          "left": "${P:A:PARSE{{success}}}"
        }
      ],
      "condition_type": "OR",
      "error": true
    }
  },
```

```
  {
    "name": "checkSaveSyncedAt",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
         {
           "left": "${L::SaveEA}", "op": "!=", "right": "true"
         }
      ],
      "next": "check_Scan_on_Add"
    }
  },

  {
     "name": "Update R7_SyncedAt",
     "operation": "PUT",
     "transport": {"path": "${L:A:Obj_ref}"},
     "wapi": "v2.6",
     "wapi_quoting": "JSON",
      "body_list": [
         "{",
         "\"extattrs+\":{\"R7_SyncedAt\": { \"value\": \"${UT:U:TIME}\"}}",
         "}"
      ]
  },
```

If **SaveEA** is true, update **R7_SyncedAt** extensible attribute

```
  {
    "name": "check_Scan_on_Add",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "OR",
      "statements": [
         {"left": "${L::ScanOnAdd}", "op": "==", "right": "false"},
         {"left": "${E::event_type}", "op": "==", "right": "NETWORK_IPV4"},
```

Stop if scan after the object creation was not requested or it is a network/range

`{"left": "${E::event_type}", "op": "==", "right": "NETWORK_IPV6"}, {"left": "${E::event_type}", "op": "==", "right": "RANGE_IPV4"}, {"left": "${E::event_type}", "op": "==", "right": "RANGE_IPV6"} ], "stop": true } },`	
`{ "name": "assignScanVars", "operation": "NOP", "body_list": [ "${XC:COPY:{L:ScanDate}:{UT:TIME}}${XC:FORMAT:TRUNCATE:{L:ScanDate}:{10t}}", "${XC:COPY:{L:R7ScanSchTime}:{UT:EPOCH}}${XC:FORMAT:DATE_STRFTIME:{L:R7ScanSchTime}:{%Y%m%dT%H%M59000Z}}" ] },`	Set local variables: **ScanDate** is used as a value for R7_LastScan attribute **R7ScanSchTime** is used as a scheduled scan time in Rapid7 Nexpose API call
`{ "name": "Create a schedule", "operation": "SERIALIZE", "serializations": [{"destination": "L:R7ScanSch","content": "<Schedules><AdHocSchedule start=\"${L:A:R7ScanSchTime}\" template=\"${L:A:ScanTemplate}\" /> </Schedules>"}, {"destination": "L:R7ScanByHost","content": "<Hosts><host>${L:A:Hostname}</host></Hosts>"}, {"destination": "L:R7ScanByIP","content": "<Hosts><range from=\"${L:A:IPFrom}\"/></Hosts>"}] },`	XML templates are created for an API request: **R7ScanSch** - contains a schedule with a scan template **R7ScanByHost** - contains a target hostname to scan **R7ScanByIP** - contains a target IP-address to scan
`{ "name": "scanByHostname", "operation": "CONDITION", "condition": { "condition_type": "AND", "statements": [{"left": "${L::AddByHostname}", "op": "==", "right": "true"}, {"left": "${L::Hostname}", "op": "!=", "right": ""}], "eval": "${XC:COPY:{L:R7ScanHostsRanges}:{L:R7ScanByHost}}", "else_eval": "${XC:COPY:{L:R7ScanHostsRanges}:{L:R7ScanByIP}}" } },`	if an event was triggered by a host which was added to Rapid7 Nexpose by a hostname and a hostname is exists use **R7ScanByHost** as a scan target, otherwise use **R7ScanByIP**
`{ "name": "skipSchedule", "operation": "CONDITION", "condition": { "condition_type": "OR", "statements": [ {"left": "${L::ScanTemplate}", "op": "==", "right": "default"}, {"left": "${L::ScanTemplate}", "op": "==", "right": ""} ],`	“default” is a fake scan template name. If a “default” scan was requested we do not add a schedule section into the API request. Default parameters defined for a Site in Rapid7 Nexpose will be

```
      "eval": "${XC:ASSIGN:{L:R7ScanSch}:{S:}}"
    }
  },
```

used

```
  {
    "name": "RequestAssetScan",
    "parse": "XMLA",
    "operation": "POST",
    "body_list": [
      "<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
      "<SiteDevicesScanRequest session-id=\"${S::SESSID}\"
site-id=\"${L:A:SiteID}\">",
      "${L:A:R7ScanHostsRanges}",
      "${L:A:R7ScanSch}",
      "</SiteDevicesScanRequest>"
    ]
  },

  {
    "name": "scan_site(errorcheck)",
    "operation": "CONDITION",
    "condition": {
      "statements": [
        {"left": "SiteDevicesScanResponse", "op": "!=", "right":
"${P:A:PARSE[[name]]}"},
        {"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
      ],
      "condition_type": "OR",
      "error": true
    }
  },
```

Send SiteDevicesScanRequest API request to Rapid7 Nexpose

If the request was not executed successfully, raise an error and stop execution

```
  {
    "name": "checkSaveLastScan",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
        {"left": "${L::SaveEA}", "op": "!=", "right": "true"}
      ],
      "next": "FinInsert"
    }
  },
  {
     "name": "Update R7_LastScan",
     "operation": "PUT",
     "transport": {"path": "${L:A:Obj_ref}"},
     "wapi": "v2.6",
     "wapi_quoting": "JSON",
      "body_list": [
         "{",
         "\"extattrs+\":{\"R7_LastScan\": { \"value\": \"${L:U:ScanDate}\"}}",
         "}"
      ]
  },
```

If **SaveEA** set to true and **EASource** is set to IP or HOST, update **R7_LastScan** extensible attribute.

```
  {
    "name": "FinInsert",
    "operation": "NOP",
```

If log level set to DEBUG, print all variables in the debug log.

```
    "body": "${XC:DEBUG:{L:}}${XC:DEBUG:{E:}}${XC:DEBUG:{S:}}"
  },
```

```
  {
    "name": "StopInsert",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "1", "op": "==", "right": "1"}
      ],
      "stop": true
    }
  },
```
	Stop template execution for Insert action

```
  {
    "name": "DeleteObject",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "${L:A:SiteID}", "op": "==", "right": "0"}
      ],
      "stop": true
    }
  },

  {
    "name": "CheckIfNetSynced",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "${L:A:ObjType}", "op": "==", "right": "NETWORK"},
          {"left": "${L:A:NetToSite}", "op": "!=", "right": "true"}
      ],
      "stop": true
    }
  },

  {
    "name": "CheckIfRangeSynced",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "${L:A:ObjType}", "op": "==", "right": "RANGE"},
          {"left": "${L:A:RangeToSite}", "op": "!=", "right": "true"}
      ],
      "stop": true
    }
  },
```
	Stop if SiteID is not defined (all objects) or Network/Range were not added into the assets.

```
  {
    "name": "GetSiteConf_R7_deletion",
    "parse": "XMLA",
    "operation": "POST",
```
	Retrieve a Site’s configuration. Save site’s configuration in **SiteConfig**. In case of any

```
      "body_list": [
        "<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
        "<SiteConfigRequest session-id=\"${S::SESSID}\"
site-id=\"${L:A:SiteID}\"/>"
      ]
    },

    {
      "name": "GetSiteConf_R7_deletion_errorcheck",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "SiteConfigResponse", "op": "!=", "right":
"${P:A:PARSE[[name]]}"},
          {"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
        ],
        "condition_type": "OR",
        "else_eval":
"${XC:COPY:{L:SiteConfig}:{P:PARSE{SiteConfigResponse}}}",
        "error": true
      }
    },
```

issue raise an error and stop execution

```
    {
      "name": "CheckIfNetRangeSynced_delete",
      "operation": "CONDITION",
      "condition": {
        "condition_type": "AND",
        "statements": [
           {"left": "${L:A:ObjType}", "op": "!=", "right": "NETWORK"},
           {"left": "${L:A:ObjType}", "op": "!=", "right": "RANGE"},
           {"left": "${L:A:ObjType}", "op": "!=", "right": "HOST"}
        ],
        "next": "RemoveByIP"
      }
    },

    {
      "name": "CheckDeleteByHostname_delete",
      "operation": "CONDITION",
      "condition": {
        "condition_type": "AND",
        "statements": [
           {"left": "${L:A:ObjType}", "op": "!=", "right": "HOST"},
           {"left": "${L:A:AddByHostname}", "op": "!=", "right": "true"}
        ],
        "next": "RemoveByIP"
      }
    },
```

If ObjType FIXEDIP, LEASE, HOST synced by IP jump to RemoveByIP step

```
    {
        "name": "assignEmptySiteVars_Delete",
        "operation": "NOP",
        "body_list": [
"${XC:ASSIGN:{L:SiteConfigDescription}:{S:}}${XC:ASSIGN:{L:SiteConfigHosts
}:{S:}}${XC:ASSIGN:{L:SiteConfigCredentials}:{S:}}${XC:ASSIGN:{L:SiteConfig
Alerting}:{S:}}${XC:ASSIGN:{L:SiteConfigScanConfig}:{S:}}${XC:ASSIGN:{L:Sit
eConfigTags}:{S:}}"
        ]
```

Rapid7 Nexpose Site configuration consists of several block. In order to delete a network/range or a hostname we should modify Hosts block, because of limitations we need to rebuild

```
   },
   {
     "name": "SiteConf_Description",
     "operation": "CONDITION",
     "condition": {
       "condition_type": "AND",
       "statements": [
          {"left": "${P:A:PARSE{SiteConfigResponse}{Site}{Description}}", "op":
"==", "right": ""}
       ],
       "next":"SiteConf_Hosts"
     }
   },

   {
     "name": "SiteConf_Description_Assign",
     "operation": "VARIABLEOP",
     "variable_ops": [
       {
         "operation": "ASSIGN",
         "type": "COMPOSITE",
         "name": "Description",
         "destination": "L:SiteConfigDescription",
         "source": "P:PARSE{SiteConfigResponse}{Site}{Description}"
       }
     ]
   },

   {
     "name": "SiteConf_Hosts",
     "operation": "CONDITION",
     "condition": {
       "condition_type": "AND",
       "statements": [
          {"left": "${P:A:PARSE{SiteConfigResponse}{Site}{Hosts}}", "op": "==",
"right": ""}
       ],
       "next":"SiteConf_Credentials"
     }
   },

   {
     "name": "SiteConf_Hosts_Assign",
     "operation": "VARIABLEOP",
     "variable_ops": [
       {
         "operation": "ASSIGN",
         "type": "COMPOSITE",
         "name": "Hosts",
         "destination": "L:SiteConfigHosts",
         "source": "P:PARSE{SiteConfigResponse}{Site}{Hosts}"
       }
     ]
   },

   {
     "name": "SiteConf_Credentials",
     "operation": "CONDITION",
     "condition": {
```

the configuration. The variables (SiteConfigDescription, SiteConfigHosts, SiteConfigCredentials, SiteConfigAlerting, SiteConfigScanConfig, SiteConfigTags) contain the relevant XML blocks from the site configuration

```
        "condition_type": "AND",
        "statements": [
          {"left": "${P:A:PARSE{SiteConfigResponse}{Site}{Credentials}}", "op":
"==", "right": ""}
        ],
        "next":"SiteConf_Alerting"
      }
    },


    {
      "name": "SiteConf_Credentials_Assign",
      "operation": "VARIABLEOP",
      "variable_ops": [
        {
          "operation": "ASSIGN",
          "type": "COMPOSITE",
          "name": "Credentials",
          "destination": "L:SiteConfigCredentials",
          "source": "P:PARSE{SiteConfigResponse}{Site}{Credentials}"
        }
      ]
    },

    {
      "name": "SiteConf_Alerting",
      "operation": "CONDITION",
      "condition": {
        "condition_type": "AND",
        "statements": [
          {"left": "${P:A:PARSE{SiteConfigResponse}{Site}{Alerting}}", "op": "==",
"right": ""}
        ],
        "next":"SiteConf_ScanConfig"
      }
    },

    {
      "name": "SiteConf_Alerting_Assign",
      "operation": "VARIABLEOP",
      "variable_ops": [
        {
          "operation": "ASSIGN",
          "type": "COMPOSITE",
          "name": "Alerting",
          "destination": "L:SiteConfigAlerting",
          "source": "P:PARSE{SiteConfigResponse}{Site}{Alerting}"
        }
      ]
    },

    {
      "name": "SiteConf_ScanConfig",
      "operation": "CONDITION",
      "condition": {
        "condition_type": "AND",
        "statements": [
          {"left": "${P:A:PARSE{SiteConfigResponse}{Site}{ScanConfig}}", "op":
"==", "right": ""}
```

```
    ],
    "next":"SiteConf_Tags"
   }
  },

  {
   "name": "SiteConf_ScanConfig_Assign",
   "operation": "VARIABLEOP",
   "variable_ops": [
    {
     "operation": "ASSIGN",
     "type": "COMPOSITE",
     "name": "ScanConfig",
     "destination": "L:SiteConfigScanConfig",
     "source": "P:PARSE{SiteConfigResponse}{Site}{ScanConfig}"
    }
   ]
  },

  {
   "name": "SiteConf_Tags",
   "operation": "CONDITION",
   "condition": {
    "condition_type": "AND",
    "statements": [
      {"left": "${P:A:PARSE{SiteConfigResponse}{Site}{Tags}}", "op": "==",
"right": ""}
    ],
    "next":"DeleteHostname"
   }
  },

  {
   "name": "SiteConf_Tags_Assign",
   "operation": "VARIABLEOP",
   "variable_ops": [
    {
     "operation": "ASSIGN",
     "type": "COMPOSITE",
     "name": "Tags",
     "destination": "L:SiteConfigTags",
     "source": "P:PARSE{SiteConfigResponse}{Site}{Tags}"
    }
   ]
  },
```

```
  {
   "name": "DeleteHostname",
   "operation": "CONDITION",
   "condition": {
    "condition_type": "AND",
    "statements": [
      {"left": "${L:A:ObjType}", "op": "==", "right": "HOST"}
    ],
    "next": "RemoveByHostname"
   }
  },
```

If HOST (delete by hostname) jump to RemoveByHostname

{
"name": "RemoveNetRange",
"operation": "VARIABLEOP",
"variable_ops": [
{
"operation": "POP",
"type": "COMPOSITE",
"source": "L:SiteConfigHosts",
"destination": "L:TMP",
"values": ["<range from=\"${L:A:IPFrom}\" to=\"${L:A:IPTo}\"/>"]
}
]
},
 	
{
"name": "Bypass_RemoveByHostname",
"operation": "CONDITION",
"condition": {
"condition_type": "AND",
"statements": [
{"left": "1", "op": "==", "right": "1"}
],
"next":"Delete_Save_site_config"
}
},
	Remove network(via range)/range from the configuration and jump to Delete_Save_site_config

{
"name": "RemoveByHostname",
"operation": "NOP",
"body": "${XC:DEBUG:{L:}}${XC:DEBUG:{E:}}${XC:DEBUG:{S:}}"
},
{
"name": "RemoveHostbyHostname",
"operation": "VARIABLEOP",
"variable_ops": [
{
"operation": "POP",
"type": "COMPOSITE",
"source": "L:SiteConfigHosts",
"destination": "L:TMP",
"values": ["<host>${L:A:Hostname}</host>"]
}
]
},
	Remove a hostname from the configuration

{ "name": "Delete_Save_site_config", "parse": "XMLA", "operation": "POST", "body_list": ["<?xml version=\"1.0\" encoding=\"UTF-8\"?>", "<SiteSaveRequest session-id=\"${S::SESSID}\">", "<Site id=\"${L:A:SiteID}\" name=\"${L:A:SiteConfig{Site}{{name}}}\" description=\"${L:A:SiteConfig{Site}{{description}}}\" riskfactor=\"${L:A:SiteConfig{Site}{{riskfactor}}}\" isDynamic=\"${L:A:SiteConfig{Site}{{isDynamic}}}\">", "${L::SiteConfigDescription}", "${L::SiteConfigHosts}", "${L:x:SiteConfigCredentials}",	Save configuration on Rapid7 Nexpose

"${L:x:SiteConfigAlerting}", "${L:x:SiteConfigScanConfig}", "${L::SiteConfigTags}", "</Site>", "</SiteSaveRequest>"] },	

{
"name": "CleanIPdevices",
"operation": "CONDITION",
"condition": {
"condition_type": "AND",
"statements": [
{"left": "${L:A:ObjType}", "op": "==", "right": "HOST"}
],
"next": "assignLVars_Delete"
}
},
{
"name": "Save_NetRange_Site_Delete",
"operation": "CONDITION",
"condition": {
"condition_type": "AND",
"statements": [
{"left": "1", "op": "==", "right": "1"}
],
"stop": true
}
},
	If object was HOST jump to assignLVars_Delete (remove an asset from discovered assets)

{ "name": "RemoveByIP", "operation": "NOP", "body": "${XC:DEBUG:{L:}}${XC:DEBUG:{E:}}${XC:DEBUG:{S:}}" },	Following steps are removing HOST, LEASE, FIXEDIP from defined and discovered assets by an IP-address

{
"name": "doNotRemoveHostIPfromNet",
"operation": "CONDITION",
"condition": {
"condition_type": "AND",
"statements": [
{"left": "${L:A:ObjType}", "op": "==", "right": "HOST"},
{"left": "${L:A:NetToSite}", "op": "==", "right": "true"}
],
"next": "assignLVars_Delete"
}
},
 	
{
"name": "doNotRemoveFixedIPfromNet",
"operation": "CONDITION",
"condition": {
"condition_type": "AND",
"statements": [
{"left": "${L:A:ObjType}", "op": "==", "right": "FIXEDIP"},
{"left": "${L:A:NetToSite}", "op": "==", "right": "true"}
],
	If a range or network was added to the defined assets (do not remove an IP from a defined assets) jump to assignLVars_Delete step.

"next": "assignLVars_Delete"
    }
  },
 	
  {
    "name": "doNotRemoveLeaseIPfromNet",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "${L:A:ObjType}", "op": "==", "right": "LEASE"},
          {"left": "${L:A:RangeToSite}", "op": "==", "right": "true"}
      ],
      "next": "assignLVars_Delete"
    }
  },	

{
    "name": "RemoveIPFromRanges",
    "operation": "NOP",
    "body_list": [
        "${XC:REMOVEIP:{L:IPFrom}:{L:SiteConfig{Hosts}}}"
    ]
  },
 	
  {
    "name": "Save_IP_Site_Delete",
    "operation": "CONDITION",
    "condition": {
      "condition_type": "AND",
      "statements": [
          {"left": "1", "op": "==", "right": "1"}
      ],
      "next": "Save_Site_Config_Delete"
    }
  },
	Remove the IP-address from the defined assets

{
    "name": "Save_Site_Config_Delete",
    "operation": "NOP",
    "body": "${XC:DEBUG:{L:}}${XC:DEBUG:{E:}}${XC:DEBUG:{S:}}"
  },
 	
  {
    "name": "Save site config Delete",
    "parse": "XMLA",
    "operation": "POST",
    "body_list": [
      "<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
      "<SiteSaveRequest session-id=\"${S::SESSID}\">",
      "${L:x:SiteConfig}",
      "</SiteSaveRequest>"
    ]
  },
  {
    "name": "Save_site_delete(errorcheck)",
    "operation": "CONDITION",
    "condition": {
      "statements": [
	Save site configuration. In case of any issues rise an error and stop execution

```
        {"left": "${P:A:PARSE[[name]]}", "op": "!=", "right": "SiteSaveResponse"},
        {"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
      ],
      "condition_type": "OR",
      "error": true
    }
  },
```

```
  {
      "name": "assignLVars_Delete",
      "operation": "NOP",
      "body_list": [
          "${XC:ASSIGN:{L:DeviceID}:{S:}}"
      ]
  },
  {
    "name": "GetSiteDeviceListR7_del",
    "parse": "XMLA",
    "operation": "POST",
    "body_list": [
       "<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
       "<SiteDeviceListingRequest session-id=\"${S::SESSID}\"
site-id=\"${L:A:SiteID}\"/>"
    ]
  },
  {
     "name": "GetSiteDeviceListR7_del_errorcheck",
     "operation": "CONDITION",
     "condition": {
        "statements": [
           {"left": "SiteDeviceListingResponse", "op": "!=","right":
"${P:A:PARSE[[name]]}"},
           {"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
        ],
        "condition_type": "AND",
        "else_eval": "${XC:COPY:{L:site_list}:{P:PARSE}}",
        "error": true
     }
  },
```

Next steps remove an asset from the discovered assets by an IP-address.

Retrieve a list of discovered assets for the site

```
  {
     "name": "Check_site_list_empty",
     "operation": "CONDITION",
     "condition": {
        "statements": [
           {"left": "${L:L:site_list}", "op": "==","right": "0"}
        ],
        "condition_type": "AND",
        "next": "FinDelete"
     }
  },

  {
      "name": "Pop_device_list",
      "operation": "VARIABLEOP",
      "variable_ops": [
         {
            "operation": "POP",
            "type": "COMPOSITE",
```

In a loop check all assets by IP and if the asset was found set **DeviceID** variable

```
          "destination": "L:device_list",
          "source": "L:site_list"
        }
      ]
    },

    {
      "name": "Check_device_list_empty",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "${L:L:device_list}", "op": "==","right": "0"}
        ],
        "condition_type": "AND",
        "next": "Check_site_list_empty"
      }
    },

    {
       "name": "Pop_a_device",
       "operation": "VARIABLEOP",
       "variable_ops": [
         {
          "operation": "POP",
          "type": "COMPOSITE",
          "destination": "L:a_device",
          "source": "L:device_list"
        }
      ]
    },

    {
      "name": "check_if_device_found",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "${L:A:IPFrom}", "op": "!=", "right": "${L:A:a_device{{address}}}"}
        ],
        "condition_type": "AND",
        "next": "Check_device_list_empty",
        "else_eval": "${XC:COPY:{L:DeviceID}:{L:a_device{{id}}}}"
      }
    },


    {
      "name": "loop_sites",
      "operation": "CONDITION",
      "condition": {
        "statements": [
          {"left": "${L:A:DeviceID}", "op": "==", "right": ""}
        ],
        "condition_type": "AND",
        "next": "Check_site_list_empty"
      }
    },

    {
      "name": "Check_DeviceID",
```

"operation": "CONDITION",
"condition": {
"statements": [
{"left": "${L:A:DeviceID}", "op": "==", "right": ""}
],
"condition_type": "AND",
"next": "FinDelete"
}
},	

{
"name": "DeleteDeviceR7",
"parse": "XMLA",
"operation": "POST",
"body_list": [
"<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
"<DeviceDeleteRequest session-id=\"${S::SESSID}\"
device-id=\"${L:A:DeviceID}\"/>"
]
},
 	
{
"name": "DeleteDeviceR7_errorcheck",
"operation": "CONDITION",
"condition": {
"statements": [
{"left": "DeviceDeleteResponse", "op": "!=","right":
"${P:A:PARSE[[name]]}"},
{"left": "${P:A:PARSE{{success}}}", "op": "!=", "right": "1"}
],
"condition_type": "AND",
"error": true
}
},
	Delete device by **DeviceID**

{
"name": "FinDelete",
"operation": "NOP",
"body":
"${XC:DEBUG:{L:}}${XC:DEBUG:{E:}}${XC:DEBUG:{S:}}${XC:DEBUG:{P:}}"
}
]
}
	If log level set to DEBUG, print all variables in the debug log.