

IPSWITCH

TECHNICAL NOTE

Monitoring Redundancy Solutions in WhatsUp Gold v.12

WhatsUp Gold v.12 incorporates a monitoring redundancy or fail-over solution built into the base product. This Technical Note details two possible implementations for companies that require monitoring redundancy.

Please note that the solutions provided in this Technical Note require some knowledge of SQL and the more advanced features of the WhatsUp application.



Introduction

When monitoring business critical devices, you cannot afford to be 'in the dark' when it comes to the state of those resources. If and when something goes down, you need to know about the problem before you can fix it. As someone who uses or is finding out more about WhatsUp Gold v.12, you recognize the need for network monitoring, and know how important that need is. So what happens when the problem device on your network is the computer running WhatsUp Gold v.12 itself?

Even though you've set everything up correctly, if that computer crashes or loses its network connection, you will be completely in the dark again. That's why a Failover/Fault Tolerance or monitoring redundancy solution is important. The terms Failover and Fault Tolerance refer to second-tier or backup network monitoring. When the primary installation of WhatsUp Gold v.12 is no longer responsive on the network, a second installation on another computer takes over the monitoring for the primary.

This paper describes two possible solutions that should provide 24/7 monitoring for most WhatsUp Gold v.12 configurations. Each solution provides a different type and amount of coverage, but each is built on the same foundation.

The solutions here are simple solutions that all users should be able to configure and use. These are not 'complete' failover solutions, in that no databases are exchanged, and monitor data is not synchronized between systems. These solutions are to give 24/7 monitoring coverage to your devices.

Running Multiple Installs

For both of the solutions in this paper, you must be running two copies of WhatsUp Gold v.12 concurrently. To do this, you will need to purchase two activation codes from our sales department. If you state that the second copy will be used in the failover role, you will qualify for a discount on the second activation code.

Create the Foundation

Once you have completely configured your primary system, you are ready to build your failover foundation. This foundation is made up of a regularly scheduled automatic backup of your primary WhatsUp Gold v.12 database. Since the database is the heart of the application's functionality, it is necessary to make sure the data stored there is as up-to-date and safe as possible. Imagine what you would lose if the hard drive the database is stored on failed to load. You would lose all of your configuration information, all of your historical data, and any customized feature you created during the setup and operation of the application.

1. **Decide how often you want to back up your database.** The real decision is to determine how long you are willing to go without data. If you only back up the database once a day, you have the potential of losing 23 hours worth of data.
2. **Create a .bat file to store the backup osql script.** The batch file should be located in the MSDE program installation directory. The default location is **C:\Program Files\Microsoft SQL Server\80\Tools\Binn.**

The following is an example of how the osql script works with the default WhatsUp Gold v.12 installation:

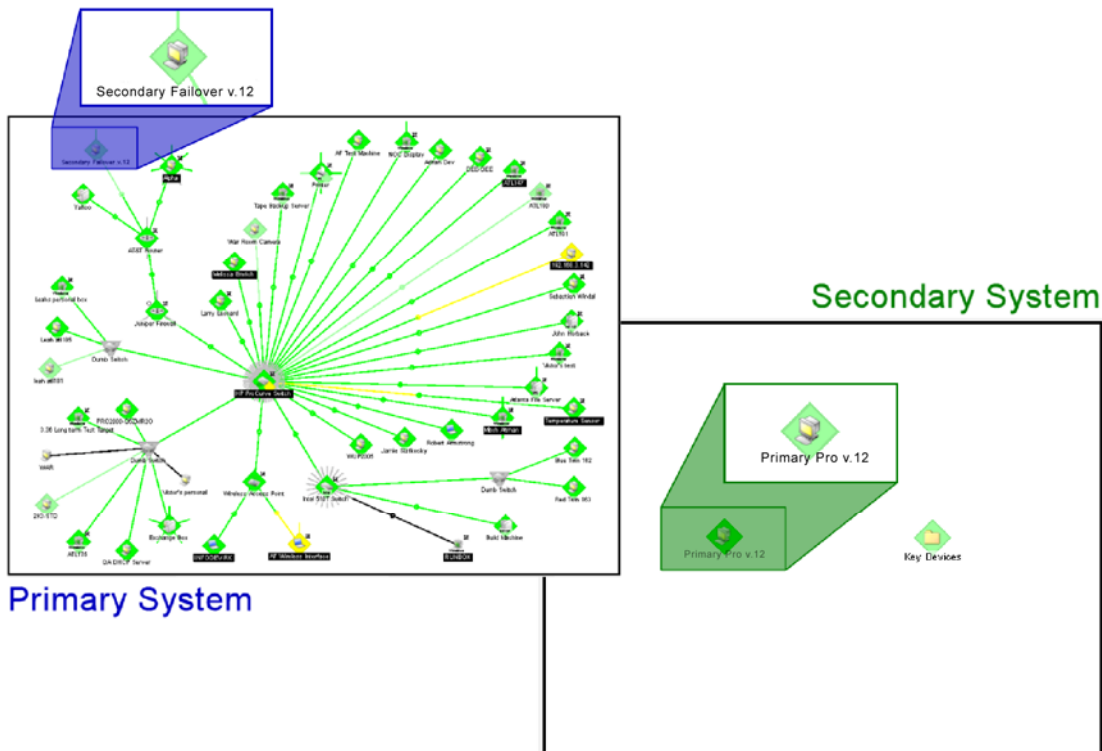
```
OSQL -E -n -D WhatsUp -Q "BACKUP DATABASE WhatsUp TO DISK =  
'C:\Program Files\Microsoft SQL  
Server\MSSQL$WHATSUP\Data\WhatsUp.dat' WITH INIT"
```

3. **Create a recurring action to execute the .bat file.** Go to the Action Library and configure a Program action to execute the .bat file. Then go to the Recurring Action dialog and schedule that action to execute based on the frequency of your choosing.

Now that the foundation has been established, you can begin the next steps in configuring a WhatsUp failover solution.

Configuring Your Solution

The following three failover solutions defined below offer differing amounts of failover coverage, each with different levels of difficulty in setting up. Choose and implement the solution that meets your requirements and needs.



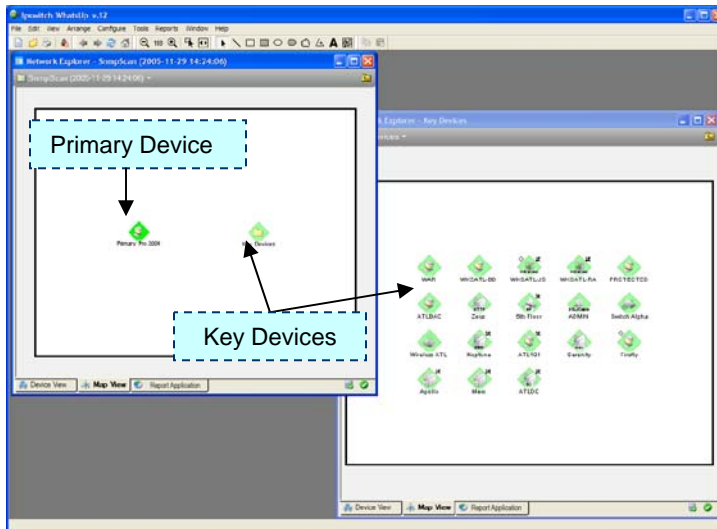
In each solution, there is a device on the primary dedicated to monitoring the secondary, and a device on the secondary monitoring the primary. These devices are the keys to the entire failover system, and must be configured properly to ensure complete coverage in the case of the failure of the primary.

Solution 1

In this solution, the specific key devices are configured on the secondary system, and are only monitored when the primary system is down. When the primary system goes back up, the devices are no longer polled.

Use this solution if:

- You only want to monitor key devices while in the failover state.
- You do not want to monitor performance data during the failover state.



Pros:

- Configuration is simpler.
- 24/7 coverage on key devices.
- Requires little bandwidth.

Cons:

- Does not provide complete coverage of network.
- Performance Data should be turned off.

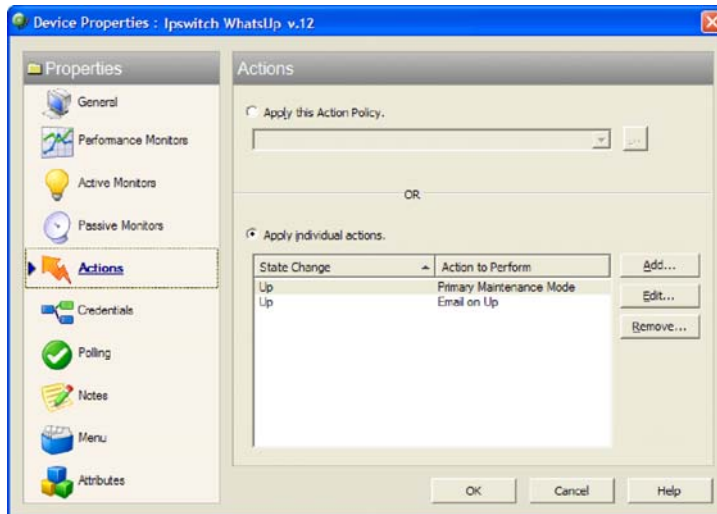
How to configure this solution:

For 'how-to' information on specific topics, such as creating a device group, or configuring active monitors, please refer to the WhatsUp Gold v.12 Online Help.

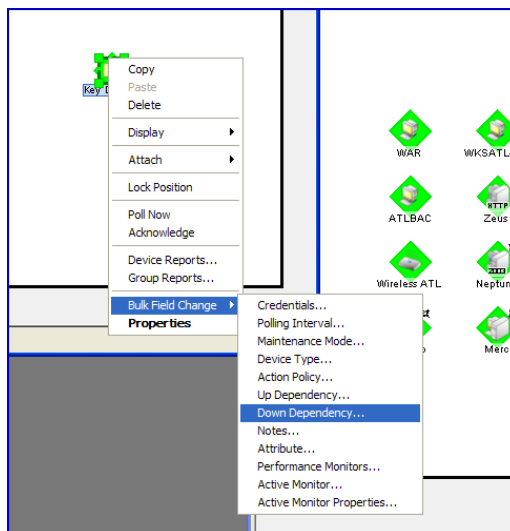
1. Once your primary system has been configured for normal operation, install WhatsUp Gold v.12 on your secondary system.
2. On the secondary system, create a new device group called Key Devices.
3. In the Key Devices group, create new devices for each of the key resources you need to monitor when the primary goes down.
4. Configure those devices with the same monitors and actions you have in the primary.

Note: Be sure to disable Performance Monitoring on these devices during configuration. Since this solution uses the dependencies feature, you will constantly gather performance data, even when you are not actively using the secondary system for monitoring.

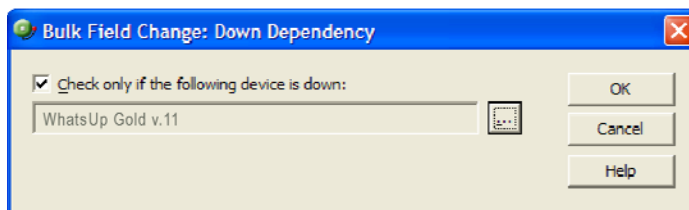
5. In the top level of the device list, create a device for the primary system.
6. Make sure that Ping is the only monitor that is configured on the primary system device.
7. Place an action on this device that alerts you when the primary goes down, and the failover state begins.
8. Place an action on this device that alerts you when the primary system is up again, and the failover state ends.



- Right-click the device group (or groups) in the device list and select **Bulk Field Change > Down Dependency**.



- On the Down Dependency dialog, select **Check only if the following device is down**, then use the browse (...) button to select the primary device created in step 6.

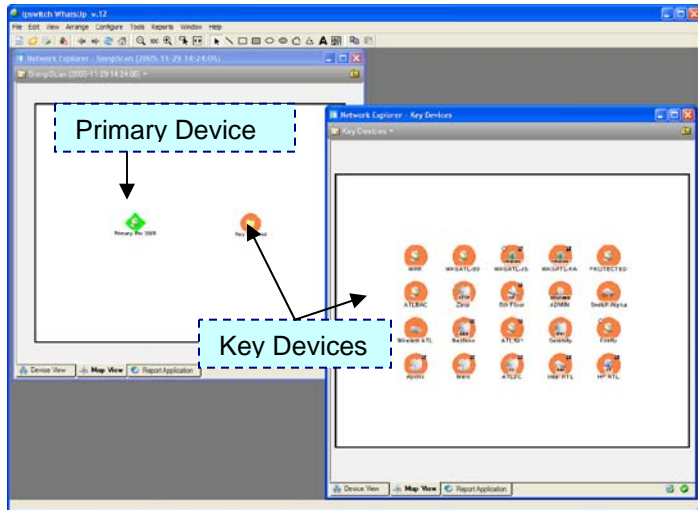


Solution 2

In this solution, all of the key devices are in maintenance mode on the secondary system until the primary device goes down. When the device goes down, the Active Script action fires, turning off maintenance mode to all of the devices. When the primary goes back up, maintenance mode is turned back on.

Use this solution if:

- You need to monitor performance data during the failover state.



Pros:

- 24/7 coverage on key devices.
- Requires little bandwidth.
- Performance Data can be collected while in Failover state.

Cons:

- Does not provide complete coverage of network, unless you choose to configure all devices in the secondary.
- Requires some scripting knowledge (or at least exposure.)

How to configure this solution:

For 'how-to' information on specific topics, such as creating a device group, or configuring active monitors, please refer to the WhatsUp Professional Online Help.

1. Once your primary system has been configured for normal operation, install WhatsUp Gold v. 12 on your secondary system.
2. On the secondary system, create a new device group called Key Devices.
3. In the Key Devices group, create new devices for each of the key resources you need to monitor when the primary goes down.

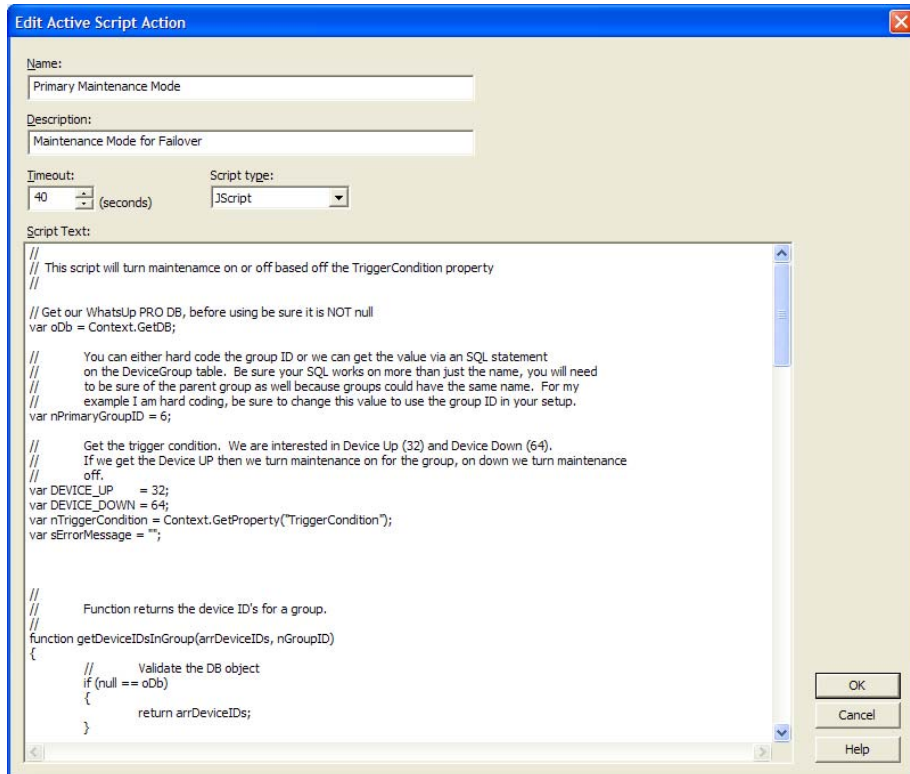
Note: If you have more than one device group you want to monitor during failover, these groups must be placed in the Key Devices group.

4. Configure those devices with the same monitors and actions you have in the primary.
5. In the top level of the device list, create a device for the primary system.
6. Make sure that Ping is the only monitor that is configured on the primary system device.
7. Run the following osql command from a cmd window:

```
osql -E -D WhatsUp -Q "SELECT nDeviceGroupID FROM DeviceGroup WHERE sGroupName = 'Key Devices' "
```

This number is used at line 13 in the Active Script shown below, to identify the device group.

8. In the Action Library, create a new Active Script action that places the devices in the Key Devices group in or out of maintenance mode when executed.



The complete syntax for this code can be cut and pasted from the following example. Once you have pasted it into the Active Script window, follow the directions commented in the code to make sure you are calling the correct device group name/ID.

Script Start

```
//
// This script will turn maintenance on or off based off the TriggerCondition property
//

// Get our WhatsUp PRO DB, before using be sure it is NOT null
var oDb = Context.GetDB;

//      You can either hard code the group ID or get the value via an SQL statement
//      on the DeviceGroup table. The ID number is hard coded in this example. Be sure to
//      change this value to use the group ID in your setup. See Solution 2 - step #6 in the White Paper for
//      more information on how to do this.
var nPrimaryGroupID = 6;

//      Get the trigger condition. We are interested in Device Up (32) and Device Down (64).
//      If we get the Device UP then we turn maintenance on for the group, on down we turn
//      maintenance
//      off.
var DEVICE_UP = 32;
var DEVICE_DOWN = 64;
var nTriggerCondition = Context.GetProperty("TriggerCondition");
var sErrorMessage = "";

//      Function returns the device ID's for a group.
//
//
function getDeviceIDsInGroup(arrDeviceIDs, nGroupID)
{
    //      Validate the DB object
    if (null == oDb)
    {
        return arrDeviceIDs;
    }
}
```

```

function getDeviceIDsInGroup(arrDeviceIDs, nGroupID)
{
    //      Validate the DB object
    if (null == oDb)
    {
        return arrDeviceIDs;
    }

    var sSql = "SELECT bDynamicGroup FROM DeviceGroup WHERE nDeviceGroupID = " +
nGroupID;
    var oRs = oDb.Execute(sSql);
    if ( !oRs.EOF )
    {
        if(oRs("bDynamicGroup") == 1)
            return arrDeviceIDs;
    }

    sSql = "SELECT DISTINCT Device.nDeviceID "
        + "FROM Device "
        + "INNER JOIN PivotDeviceToGroup ON Device.nDeviceID =
PivotDeviceToGroup.nDeviceID "
        + "WHERE PivotDeviceToGroup.nDeviceGroupID = " + nGroupID;

    oRs = oDb.Execute(sSql);
    while(!oRs.EOF)
    {
        arrDeviceIDs[arrDeviceIDs.length] = parseInt(oRs("nDeviceID"));
        oRs.MoveNext();
    }
    var oGroupRs = oDb.Execute(""
        + "SELECT DISTINCT nDeviceGroupID "
        + "FROM DeviceGroup "
        + "WHERE nParentGroupID = " + nGroupID
        );
    while(!oGroupRs.EOF)
    {
        arrDeviceIDs = getDeviceIDsInGroup(arrDeviceIDs,
parseInt(oGroupRs("nDeviceGroupID")));
        oGroupRs.MoveNext();
    }
    oGroupRs.Close();
    oRs.Close();
    return arrDeviceIDs;
}

function StartProcess()
{
    Context.NotifyProgress( "The trigger state is " + nTriggerCondition);
    //      Only work on these two trigger states.
    if (nTriggerCondition == DEVICE_UP || nTriggerCondition == DEVICE_DOWN)
    {
        var bMaintenance = true; // assume maintenance should be turned on
        if (nTriggerCondition == DEVICE_DOWN)
        {
            bMaintenance = false;
        }
        var arrDeviceIDs = new Array()
        arrDeviceIDs = getDeviceIDsInGroup(arrDeviceIDs, nPrimaryGroupID);

        sErrorMessage = "Processing " + arrDeviceIDs.length + " devices";
        if (arrDeviceIDs.length > 0)

```

```

        {
            //      Use the WhatsUp PRO helpers
            var oUtility = new ActiveXObject("CoreAsp.Utility");
            var oEventHelper = new ActiveXObject("CoreAsp.EventHelper");
            for(var i = 0; i < arrDeviceIDs.length; i++)
            {
                oUtility.SetDeviceMaintenanceMode(bMaintenance,
                    arrDeviceIDs[i]);
                oEventHelper.SendChangeEvent(2, arrDeviceIDs[i], 1);
            }
        }
    }
    else
    {
        sErrorMessage = "Trigger state " + nTriggerCondition + " was not processed.";
    }
}

return 0;
}

//
//      Start of the process.
//

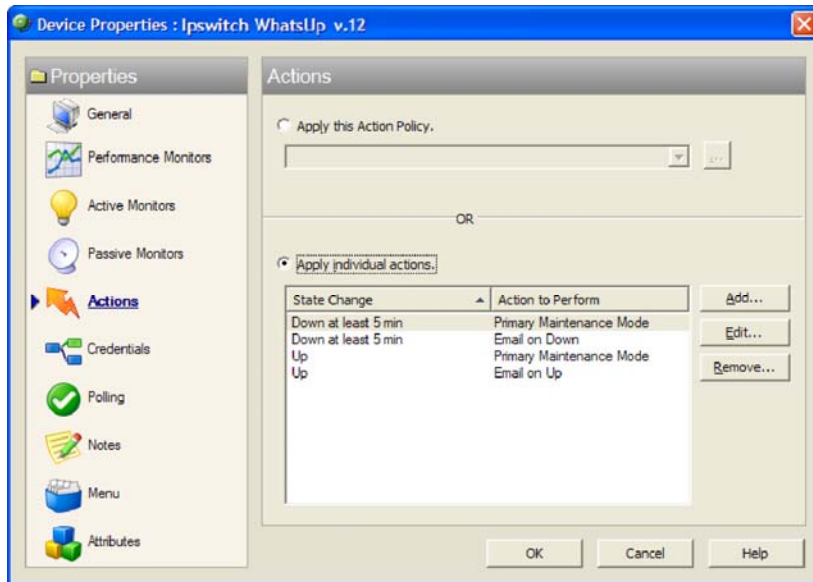
//      Test the trigger to be sure if we are in the right state.
var nReturn = StartProcess();

Context.NotifyProgress( sErrorMessage);
Context.SetResult(nReturn, sErrorMessage);

```

Script End

9. Place an action on this device that alerts you when the primary goes down, and the failover state begins.
10. Place an action on this device that alerts you when the primary system is up again, and the failover state ends.
11. Assign the Primary Maintenance Mode action to the primary device, associating it with an up state.
12. Assign the Primary Maintenance Mode action to the primary device, associating it with a down state. You may want to use the Down for 5 minutes state to give you time between when you receive the down alert and when the failover state begins.



Other Solutions?

These solutions have been validated by Ipswitch and will provide a failover capability if the above directions are followed.

Copyright © 2008, Ipswitch, Inc. All rights reserved. WhatsUp is a registered trademarks of Ipswitch, Inc. Other products or company names are or may be trademarks or registered trademarks and are the property of their respective holders.