PatchMaker: A Physical Network Patch Manager Tool

Joseph R. Crouthamel, James M. Roberts, Christopher M. Sanchez, and Christopher J. Tengi – Princeton University

ABSTRACT

PatchMaker is a network management tool that tracks and stores descriptions of physical network patches in a database. Physical patches are cables that connect switches, patch panels, and room ports. PatchMaker allows administrators to manage switch ports from multiple vendors giving them the ability to enable/disable ports, make VLAN changes, and perform other switch port functions through a common web front-end. Finally, PatchMaker aids in host management by tracing network connections, identifying what kinds of devices are connected to the network, and helping to troubleshoot physical network problems.

Introduction

Over the past year, the network/systems administrative group in the Department of Computer Science at Princeton University has been putting together a plan (along with a good bit of code) to ease the lives of both the end-users and administrative staff. The many goals laid out by the team revolve not only around ease-of-use and happy end-users, but also around issues such as security, resource accounting (and, indeed, accountability), and administrative overhead.

PatchMaker was the first application of a suite of software. Our goal in developing PatchMaker was to integrate all the tools needed to deal with end-user requests related to the network into a single web GUI front-end. We wanted a system that would track physical patches, manage switch ports, track hosts, and be extensible. Since the cost of buying enough switch ports to connect all available ports on every wall box would exceed our budget, we needed a simple way to track patches from the wall boxes all the way back to switch ports.

In the past, whenever a user changed locations we would be required to move the original patch physically to another patch panel and, at the same time, to update the cabling documentation. We wanted to improve our turn-around time and respond faster to user requests. PatchMaker allows an administrator to locate a host quickly, find out what switch port it is located on, and move the cable to its new location without needing to trace the connection by hand.

The Department of Computer Science at Princeton University has 1600+ CAT 5e RJ-45 ports located throughout the building. Each room contains multimode and single-mode fiber optic ports along with RG6 for CATV. We are using two Foundry FastIron 1500 switches and an Alcatel OmniCore 5052 switch to connect the bulk of the user ports throughout the building. While most of the network connections "home run" back to a central machine room, we are starting to maintain our networks in other neighboring buildings, as we have run out of space in the original five-floor building.

Keeping track of all the connections from the room jacks, patch panels, and switches is very important, since users and hardware are moving all the time. Network administrators are constantly dealing with end-user moves, adds, and changes. Having out-of-date cabling documentation was always a problem and it increased the time required for an administrator to deal with end-user requests.

PatchMaker was built out of a need for a more automated way of resolving problems at the physical network layer. Prior to PatchMaker, system administrators manually entered changes reflecting patch additions, changes, and deletions into a database. The system was often out-of-date. As a result, administrators would eventually abandon the database altogether, forcing us to perform audits twice a year to re-inventory the location of all physical patches. In developing a patch database system that was easy to keep updated, and by adding tools to administer switches and to trace switch connections, we have found that our administrators are now using the system more effectively.

Motivation For the Project

Documentation for physical network cabling is essential in managing and troubleshooting a network. Documentation done after the fact is often wrong or out-of-date. In our previous system we would do a full building audit of the cabling and hardware connected to the network twice a year. We then would try to keep the data current using a locally written PERL script. Audits were very time consuming and often data was out-of-date even before the audit was even completed.

Once the information was stored in a database, administrators did not keep the patch information updated. The system was used to track patches, but

administrators had little incentive to use the system, because in most cases it was faster to trace a patch by hand than it was to use the database. Over the course of several years we attempted to institute procedures with the aim of encouraging administrators to maintain the system. We still failed to gain total compliance, as administrators typically respond to emergencies and to frequent experimentation by creating ad-hoc patches that never get documented.

As the department grows, new technology becomes available, systems administration tasks also grow, so we needed to identify areas of our work that could be automated, freeing up time to focus on other tasks. The PatchMaker system was developed to solve the problems we had with the previous system and to add functionality that would make the system more useful for other tasks related to network administration. The previous system's principle failing was that the data was not up-to-date. PatchMaker addresses this problem by making data updates fast and easy, while removing the previous ineffective system. The previous system only stored physical patches. PatchMaker adds more tools and bundles the functionality needed to complete the entire process in a single system, thus making it more useful.

We could not find existing tools that incorporated all the things we wanted. One open source tool, LANdb, did not fit our needs, as it only maintained a database of patches, and did not do switch management. The few commercial products that do exist are focused on switch management and tend to be very expensive. Other people we spoke with used a simple database, but they experienced similar problems keeping the data current.

While intelligent patch panels that totally automate the record keeping process exist, this was not a viable solution for us since we already had an infrastructure in place. It would have been expensive and time consuming to swap out patch panels. Intelligent patch panel systems usually log to a proprietary database. Proprietary databases are often inflexible, and using them makes it difficult to retrieve data for other purposes. Also, at the time we were considering them, intelligent patch panel manufacturers did not sell panels rated for Category 6 cable. This would have been a problem for us in the future.

What It Can Do

PatchMaker is a visual web GUI front-end to our network. Incorporating three main tools needed to make network changes, we created a tool that administrator's find useful and allows us to keep our cabling documentation up-to-date. The first part of the system, the network cabling documentation database, keeps track of where all the cables are located end-to-end in the building. The next part of the system is a switch management interface that allows an administrator to make changes on switch ports from multiple switch vendors. The last part of the system is host and device tracking and management.

PatchMaker allows an administrator to search for a host or device in multiple ways to find its location in the building. Using the built-in search tools, administrators can locate hosts by MAC addresses, hostnames, or physical locations such as room numbers or patch panels. Using a web interface an administrator can trace a patch, receive information about what is connected to a particular port, and make changes to whichever switch port the host is using.

When debugging a network problem, there are occasions when one suspects that the physical cabling or switch ports are not functioning properly. To help with this, PatchMaker will map the user's wall box location straight to the switch port and perform diagnostics remotely. PatchMaker is able to do this as the patch panel database contains mappings that go from the point the user plugs into the wall through intermediate patch panels to the end-point represented by a port on a switch. A user who is not connected is not generally happy. While simple solutions can sometimes be identified by the clients themselves, there are other times when cabling, or even a switch port can be faulty. In these cases, having the ability to map the user's wall box location straight to the switch port, and then remotely to perform diagnostics, is invaluable. PatchMaker does rely on a host being connected in order to locate it and gather information about the host. If a host moves to a port that is not physically wired to a switch port, there is no way of obtaining host information or doing configuration on the port. In this case, we would create a patch by hand.

In the course of normal network management, it is sometimes necessary to make small changes or to monitor ports: VLAN membership, link status, resource allocation (protocol, TCP/UDP usage, IP source and destination, bandwidth utilization). PatchMaker has the ability to configure ports from the switch-side and allows an administrator to enable/disable ports, change a port VLAN, or check the link status and speed of a port. This allows the administrator to quickly locate problems, identify hosts that may have a security problem, and disable the port to which a host is connected.

PatchMaker has a port-monitoring and alerts interface with links to graphs showing MRTG and Sflow (RFC 3176) data for the particular port. Using PatchMaker, we can access multiple tools from a single interface. For example, using SNMP and Sflow, we graph port traffic, maintain a log of traffic protocol data, detect sources of congestion, and manage quality of service on the switches.

Implementation

PatchMaker was developed using open source tools. It is written in PHP and DHTML. It uses PHP's database abstraction layer, courtesy of PEAR, and

allows for a variety of database types to be used on the backend. The database stores information about patches, buildings, racks, and rooms. It contains mappings that stretch from the point where the user plugs into the wall, through any intermediate patch-panels, to an end point represented by a port on a device. Switch information is stored in the database and includes attributes such as port count, device name, SNMP community strings, and switch slots. Image information for devices, patch panels, port pictures, and coordinates are stored in the database. An image map for a new patch panel or switch can be added and integrated into the new system in a matter of minutes.

The application runs very fast, querying the device, mostly via SNMP. The returned information is saved temporarily so other sections or panels with ports that patch to the same device can avoid re-querying the same information. It is written to work with various switches from different vendors, and plug-ins can be easily written for nonstandard devices.

The application displays a graphical representation of the patch-panels and devices to a web browser. The GUI lets the user search by building, rack, room, Ethernet address, hostname, etc. The user can select multiple patches and devices to view at the same time. At quick glance, the user can determine patch and link status by the color of the port image. Statistical information about each port, such as patch point, port speed, and VLAN can be viewed in the side-panel when the cursor is placed over it. Clicking on a port gives the user a context styled menu for enabling and disabling, changing VLANs, adding and deleting patches, changing rooms, as well as other commands.

PatchMaker incorporates links to other network monitoring tools which use Sflow (RFC 3176) and NetFlow to monitor our Foundry and Cisco switches, respectively, and gather port statistics. We also use SNMP to monitor traffic load on the switches and include links to MRTG (Multi Router Traffic Grapher) to display graphical images of the network traffic.

Even with some automation efforts, it is still necessary to enter patch information into the database. We have, nevertheless, built-in checks that watch for patches that have not been properly entered in the database. Figure 1 shows a device view in PatchMaker. In the figure, unidentified ports are marked with a black 'X' across the port. This allows an administrator to quickly identify ports that have patches, but are missing an end-point record. The system allows an administrator to know if a patch they are making is already defined in the database, and then it prompts them to update the database without having to remove the old patch. Figure 3 shows all the options that can be performed on a port. PatchMaker allows an administrator to perform switch port tasks like enable/disable ports, change VLAN, or port speed no

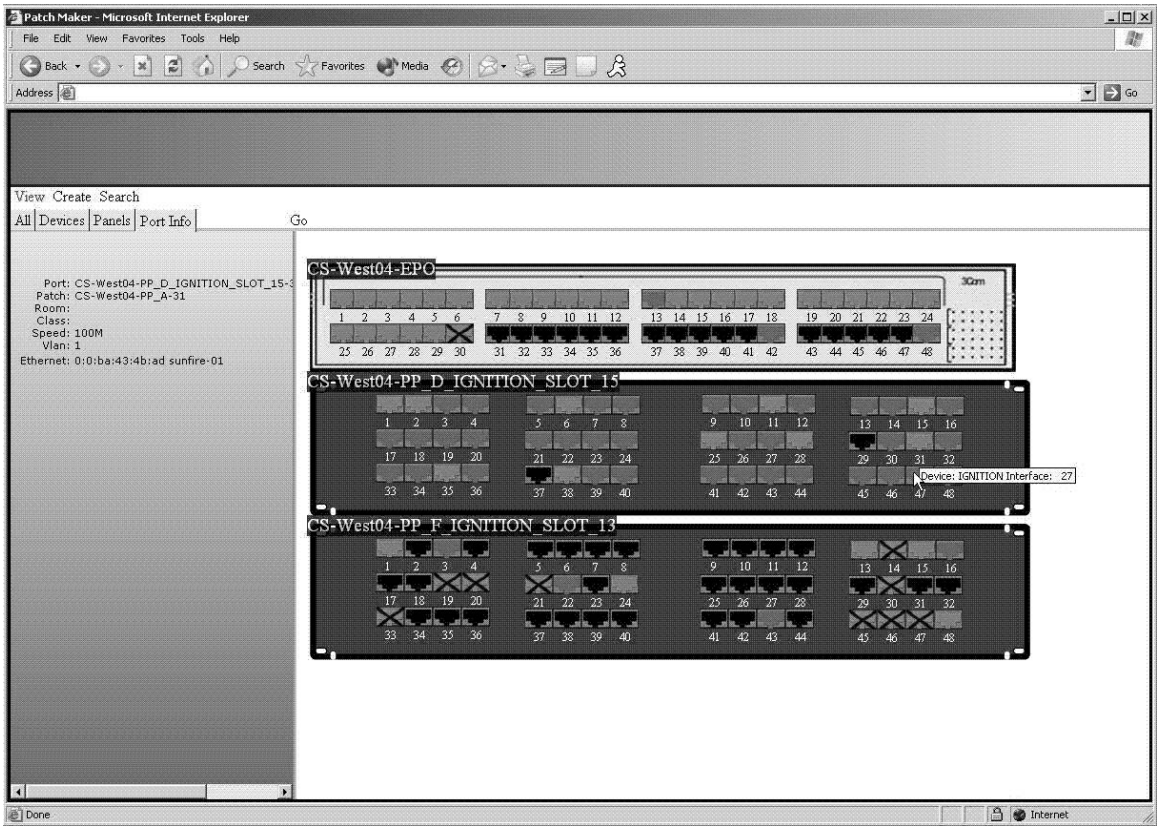

Figure 1: Devices view of PatchMaker.

matter what view you are in. Most network management tools give an administrator a switch view and allows for management of the switch from only this view. In PatchMaker if you view only the patch panels you can still change the switch port to which the patch panels connect to. Figure 4 shows a building room search and displays both the switch and patch panel information with an option to select a patch to view more information such as connected hosts. Figure 5 shows how you can display multiple devices and patch panels at the same time.

PatchMaker can handle more complex cabling infrastructures, including network closets and multiple cross connects, but Figure 2 illustrates a basic cabling diagram. In this diagram, a switch port is connected to a patch panel, then to a wall box, and finally to a host or other device. On the one hand, it is relatively simple to discover what host or device is on a particular switch port using switch management tools. On the other hand, information about which patch panel is used, and what wall box connections exist between the switch and the host, can only be discovered by hand tracing the connection or by keeping the patch documentation current. As long as patch panels and wall boxes are passive devices and do not signal their presence, this limitation is inherent.

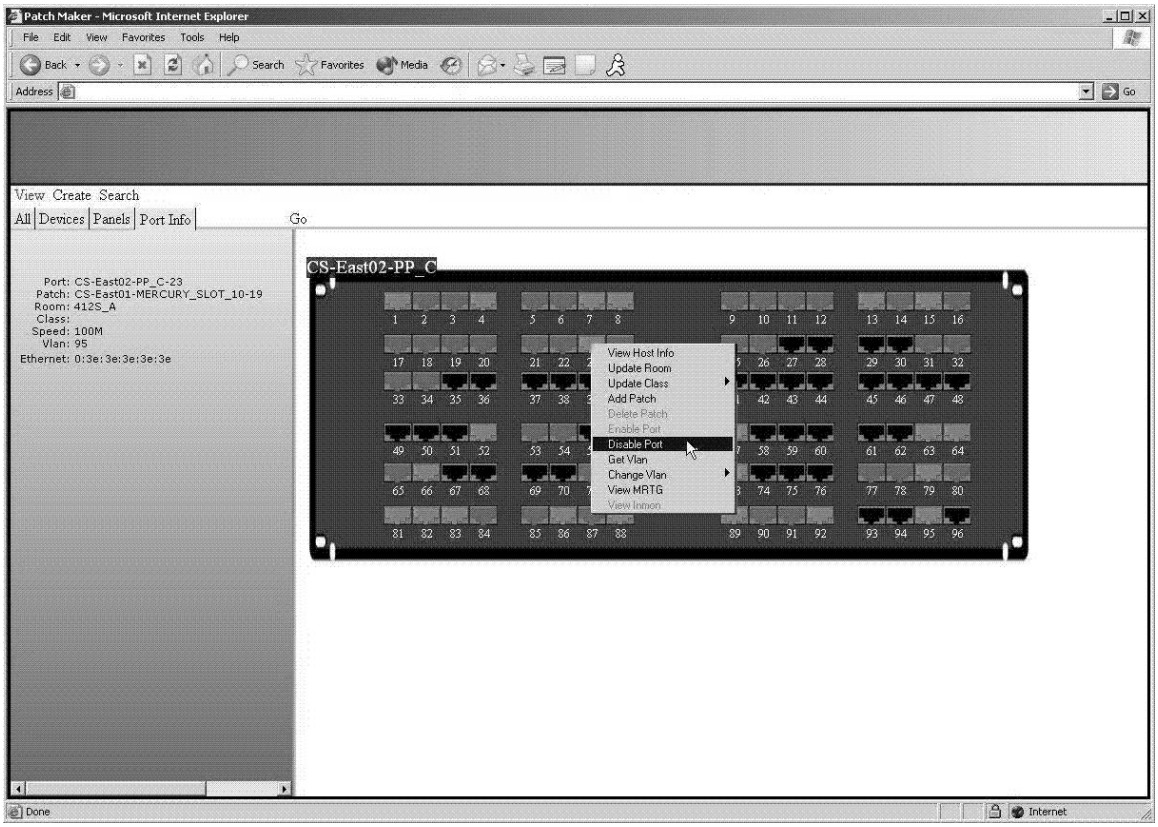

Figure 3: Patch Panel view with a drop down menu that shows all the options that can be performed on a port.

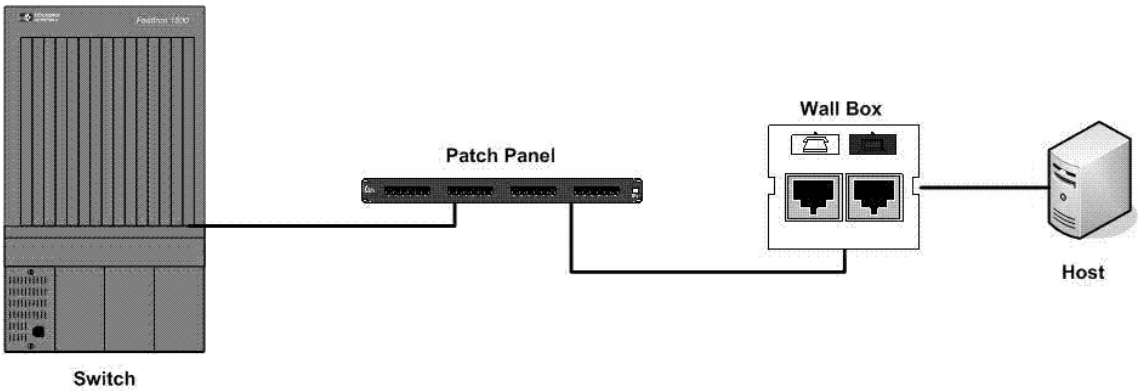

Figure 2: Basic network cabling diagram.

Future Enhancements

While PatchMaker is very useful to us, there are a few features we would like to add over time. In order for us to release this as a package so others can use it, we plan to write a GUI creation tool that would be used to set up patch panels and switch images in a drag and drop style with a small set of available images. We also would like to enhance user authentication to allow finer-grained control of user privilege levels. Other features we are working on include switch configuration file management, quality of service or rate limiting management, security access control list management, and change logging on the system and user levels.

Conclusions

The system was built out of a need for a more automated way of resolving problems at the physical network layer. Prior to PatchMaker, system administrators manually entered changes reflecting patch additions, changes, and deletions into a database. The system was always out-of-date. As a result, administrators would eventually abandon the database altogether, forcing us to perform audits twice a year to re-inventory the location of all physical patches. In developing a patch database system that was easy to keep updated, and by adding tools to administer switches and to trace switch connections, we found our administrators are using the system.

Since the system went into production, we have reduced user downtime and also reduced the time spent by administrators in dealing with network-related problems. We also decreased our response time in dealing with hosts that become compromised by a virus or trojan. We are able to disable the port and to take corrective actions on the hosts before the situation worsens. The search functionality also allows us to track hardware as it moves from room to room around the department. Our goal in creating the system was to build a tool that others could use and quickly deploy in their own environments.

Availability

PatchMaker will be released under GNU Public License (GPL) and will be made available at: http://www.cs.princeton.edu/patchmaker. To request more information about PatchMaker please send email to patchmaker@lists.cs.princeton.edu. We expect to release a public version of PatchMaker in November, 2004.

Acknowledgments

The authors would like to thank Chris M. Miller, Paul Lawson, and Craig Haley for their assistance in

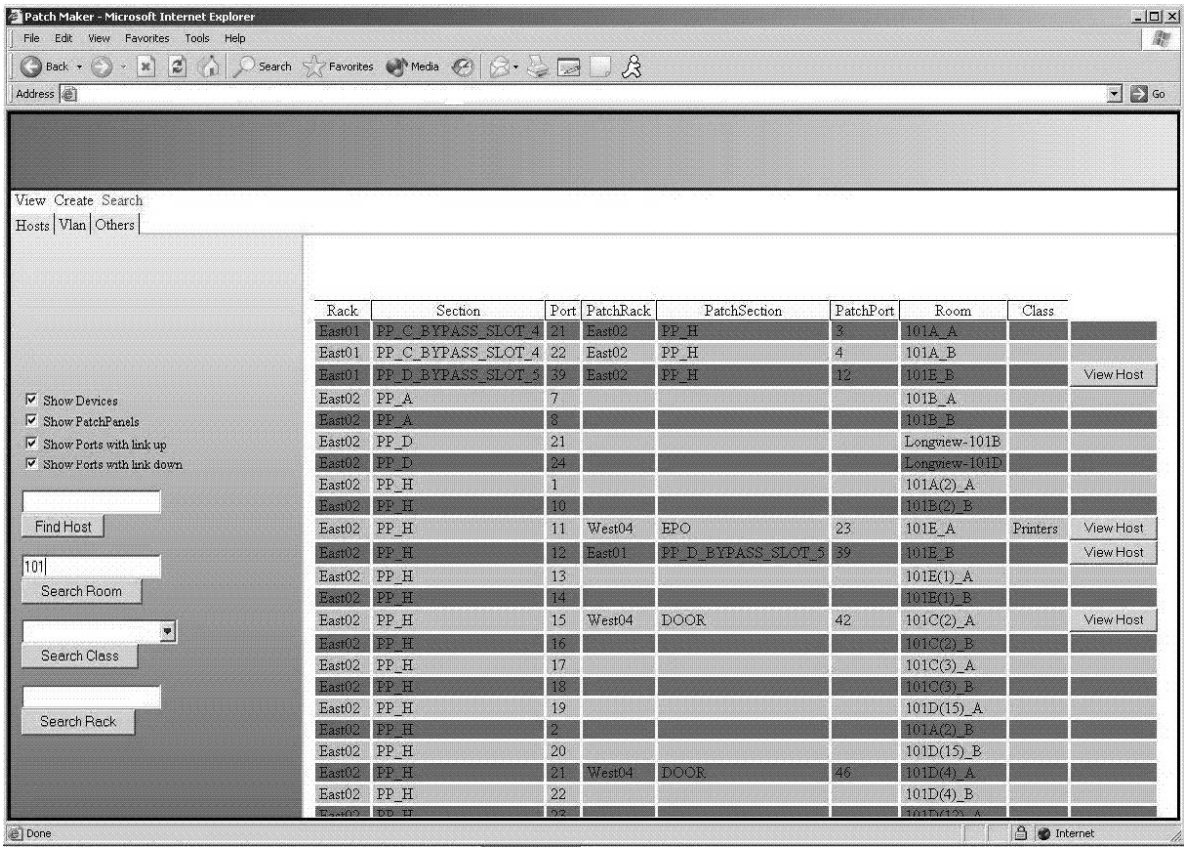

Figure 4: This image shows a search by room number. This shows the switch slot, port, and also the patch panel. You also can see more information about what is connected to a port by clicking the View Host button.

reviewing this paper. We would also like to express our sincere thanks to our shepherd Mario Obejas for his valuable feedback.

Author Biographies

All of the authors are members of the technical staff in the Computer Science Department at Princeton University.

Joseph R. Crouthamel is a System and Network Administrator in the Computer Science Department. He can be reached at jrc@CS.Princeton.EDU .

James M. Roberts is the Director of Computing School of Engineering at Tufts University. He can be reached at jmr@cs.tufts.edu .

Christopher M. Sanchez is a System and Network Administrator in the Computer Science Department. He can be reached at cmsanche@CS.Princeton.EDU .

Christopher J. Tengi is a System and Network Administrator in the Computer Science Department. He can be reached at tengi@CS.Princeton.EDU .

References

[1] http://www.usenix.org/publications/library/proceedings/lisa2000/full_papers/mitchell/mitchell_html/ .

[2] http://www.usenix.org/events/lisa2001/tech/full_papers/poynor/poynor_html/ .

[3] http://ee-staff.ethz.ch/~oetiker/webtools/mrtg/mrtg.html .

[4] http://www.sflow.org/SamplingforSecurity.pdf .

[5] http://ucd-snmp.ucdavis.edu/ .

[6] Galvin, J. M., K. McCloghrie, and J. R. Davin, "Secure Management of SNMP Networks," *Proceedings of the IFIP TC6/WG 6.6 Second International Symposium on Integrated Network Management*, pp. 703-714, North-Holland, 1991.

[7] *Sflow*, http://www.sflow.org/ .

[8] Cisco, *Cisco Netflow Flowcollector*, http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/nfc .

[9] Cisco, *Netflow Services and Applications White Paper*, http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm .

[10] Dooley, Kevin, *Designing Large-Scale LANs*, O'Reilly, January, 2002.

[11] Mellquist, Peter Erik, "SNMP++ Specification," Hewlett-Packard Company, http://rosegarden.external.hp.com/snmp++/index.html .

[12] *LANdb: The Network Management Database*, http://landb.sourceforge.net/ .

[13] *MySQL*, http://www.mysql.com .

[14] *PHP*, http://www.php.net.

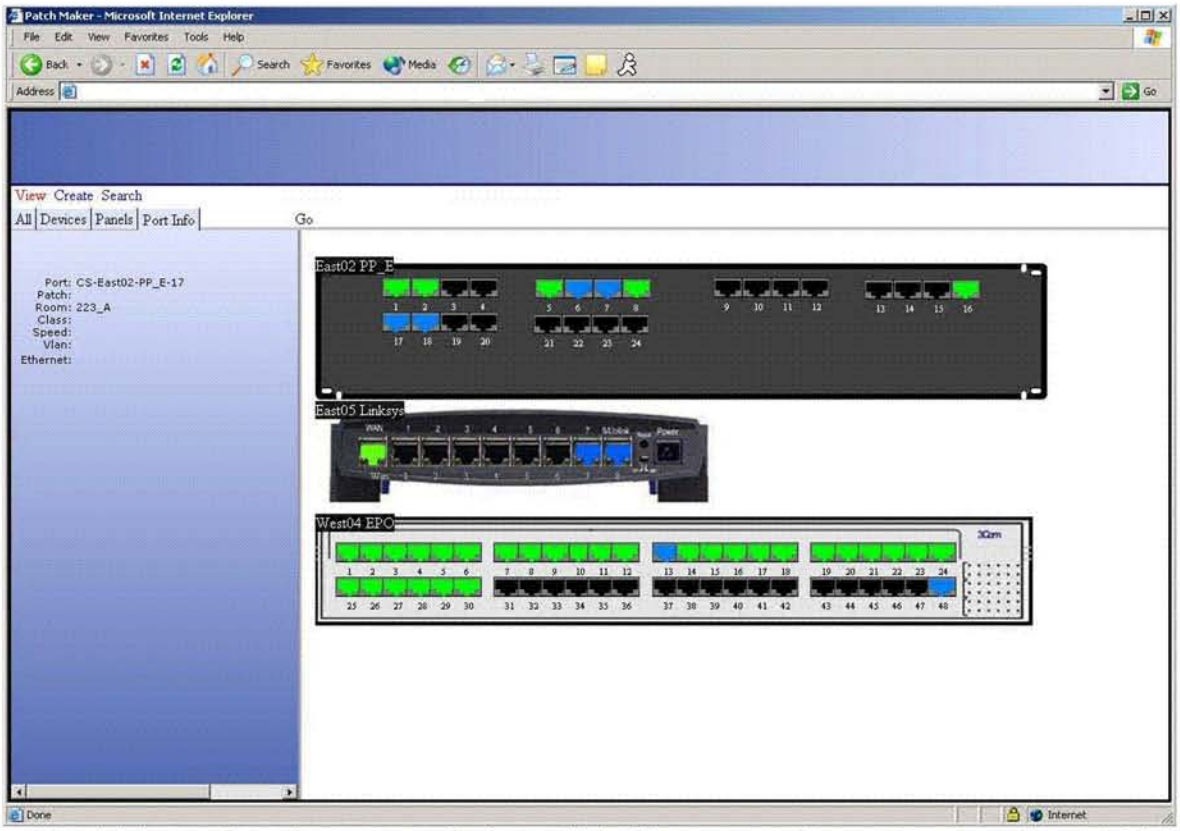

Figure 5: This image shows how you can display multiple things at once. Here we have a patch panel, LinkSys BEFSR81, and a 3Com 4300 switch.