Collecting Router Information for Error Diagnosis and Troubleshooting in Home Networks

Andreas Müller, Gerhard Münz and Georg Carle
Chair for Network Architectures and Services
Technische Universität München, Germany
{mueller, muenz, carle}@net.in.tum.de
http://www.net.in.tum.de

Abstract—**With the increasing number of heterogeneous devices that are connected to an average home network, troubleshooting in case of network problems becomes more and more complicated. This is not only because most home users are consumers and not experts in the field of networking, but also because it is usually hard to find the relevant information useful for a systematic error diagnosis. Potential sources of information, such as the web interface of a home router, are not standardized and usually not available to people, e.g. friends or help-lines, that are willing to help from outside of the home. However, many home routers run a Linux driven operating system and are therefore capable of providing a manifold amount of interesting data for debugging network related errors. This paper presents a tool which allows gathering highly customizable data on Linux based home routers using arbitrary textual sources, such as the proc-filesystem, command-line output and configuration files. Additionally, we describe the necessary steps for securely transferring the information to an external helper.**

I. Introduction

Today's home networks usually consist of an embedded DSL or cable home router which provides Internet access to desktop and mobile computers, smart phones, Internet radios and televisions sets. Self-management functions ideally provide the necessary connectivity to new devices while protecting the home network against illegitimate users. If manual intervention is necessary, it must be user-friendly and foolproof. In practice, however, the setup and configuration of a home network often causes problems which cannot be easily fixed by inexperienced home users. Even if the network has worked fine for a long time, a problem may suddenly occur due to software and hardware failures, software updates requiring some reconfiguration, or an unintended configuration change caused by the user himself. With more and more domestic appliances being connected to the home network, the risk of malfunctions and failures increases. Hence, although self-management capabilities of future home networks will likely improve, we expect that home users will still be faced with certain network problems which they cannot solve without help.

If a computer or network problem occurs, many home users rely on remote assistance provided by family members and friends with advanced computer and networking expertise, or by professional help-lines. First aid is typically given over the phone or via e-mail. However, such remote error diagnosis is often difficult and sometimes impossible, because the helper does not have direct access to any of the devices in the home network. The helper needs to rely on the error description given by the affected user, which is often imprecise and not sufficiently detailed. Under these circumstances it is no surprise that the customer satisfaction of general help-lines is rather low [2].

Aggarwal et al. [6] determined common causes of problems in home networks, reaching from DHCP and NAT misconfiguration to wrong SMTP and wireless security settings. An external helper could easily provide a solution to most of these problems if he had knowledge about the network status and configuration. Such information is available at the home router since it acts as the central access point and gateway to the Internet. All or parts of this information can be retrieved from a web interface of the router. However, opening a web interface for remote access from the Internet represents a potential security threat. Furthermore, if the Internet connection itself does not work correctly, accessing the router from outside the home network may not be possible.

In order to enable the home user to provide detailed network information to a trusted remote helper, we propose to install a *data collection agent* on the home router, which stores the available status and configuration data in a log file. If a problem occurs, the user discloses the log file to document the current network status. Error diagnosis is facilitated by a comparison with log files which were generated earlier during normal operation of the home network. In fact, these older log files can be used as a reference.

As a proof-of-concept, we implemented the proposed data collection agent as a C program which can be compiled for and run on embedded Linux routers. Relevant network information is retrieved from system configuration files, values read from the proc-filesystem, and the output of command-line tools. The generated log file can be sent via e-mail, uploaded to a file server or into a peer-to-peer (P2P) network, or saved e.g. on an external USB memory stick. If the collected data is considered sensitive, file encryption can be used to prevent unauthorized access.

In the next section, we give an overview of related work in the area of home network troubleshooting. Section III shows which kinds of network information can be obtained on an embedded Linux router. In Section IV we present the concept and implementation of our data collection agent in more detail and show how the collected information can be transferred to external helpers. Section V concludes this paper with some final remarks.

II. Related Work

Studies have shown that home users are not satisfied with current network management and diagnosis tools that are available to them [7]. Also, the frustration associated with current products for home networks and the high number of problems that occur with these products are widely known [14].

A number of proposals have been made to improve the situation. Aggarwal et al. present NetPrints, a system that sends logging data to a remote server in order to associate the current problem with similar problems reported by other users in the past [6]. This is done by comparing the locally gathered data with a database running on the server. If a fix for the earlier problems is available, it will be found in the database and provided to the inquiring user. The logging data is collected by a client program which is executed on a regular PC. If the PC is connected to the home router, router information can be obtained via Universal Plug and Play (UPnP).

In contrast to NetPrint's client program, our data collection agent is installed on the router directly, which allows us to retrieve all available internal network state and configuration data. Thus, we do not depend on a PC with a working network connection to the router and on UPnP, which is often disabled (e.g. 70% of all routers we tested in a field study [11]). Moreover, we are able to gather more information than is possible with UPnP.

Calvert et al. describe the idea of turning the home router into a "general-purpose logging facility for home networks", which they call Home Network Data Recorder (HNDR) [4]. The presented HNDR prototype captures traffic on all interfaces using `tcpdump` [16]. The authors describe different scenarios in which the collected data might be useful, one of which is troubleshooting. The big challenge is to keep up processing and storing large amounts of traffic data, which is very difficult to achieve with small home routers having much less resources than the HNDR prototype. Our approach is similar to HNDR as we also collect data on the home router. Instead of capturing traffic, however, we gather network state and configuration data from files and command-line outputs.

For error diagnosis and troubleshooting, DSL Internet Service Providers (ISPs) may leverage functions related to the TR-069 standard [3] if the home router supports it. With TR-069, the home router can establish a connection to a predefined auto-configuration server maintained by the ISP in order to download firmware updates and configuration data, such as IP telephony and IPTV settings. Hence, the ISP is able to remotely correct invalid configurations of these services. On the other hand, TR-069 is not conceived to monitor any internals of the home network, such as DHCP or WLAN settings. Furthermore, it is not possible to let the home router connect to an alternative server, which means that the usage of TR-069 is restricted to the ISP.

There are a number of other tools providing information about the network. For example, Home Watcher [8] measures and illustrates the available bandwidth and currently used bandwidth of each device in the network. Tools such as Windows Remote Desktop or VNC (Virtual Network Computing) allow connecting to an end system from outside the home network. However, these tools require a working Internet connection and some effort when setting up user credentials.

III. Data Sources on the Router

Being the central access point and gateway to the Internet, the home router represents a very valuable source of information about the status and configuration of the home network. We are particularly interested in parameters which help to diagnose common network problems, such as those determined by Aggarwal et al. [6]. Examples of relevant information are the current status of the network interfaces, the wireless configuration, static routing table entries, port forwarding settings, current DHCP leases, and DSL/cable modem statistics.

The firmware of many consumer routers is based on embedded Linux. Similar to Linux PCs, a lot of configuration and status data on home routers is stored in regular text files or is accessible from files in the proc-filesystem; some information may also be queried with help of command-line tools. The exact names and locations of specific files and commands may vary depending on the router firmware. The availability of specific information also depends on the hardware and the device drivers in use. Hence, in order to obtain a specific network parameter from the home router, it is necessary to know the name and location of the corresponding file or the name and arguments of the command-line tool.

In the following, we describe which network information can be typically found in the proc-filesystem, in configuration and status files and in the output of command-line tools. Table I gives concrete examples for OpenWRT 8.09 [13] installed on a Linksys WRT54G router.

A. *proc-Filesystem*

The virtual Linux processing filesystem (`/proc`) holds process information generated by the kernel, such as status parameters which are related to the entire system (e.g. memory information in `/proc/meminfo`) necessary to detect a possible overload, to specific devices (`/proc/devices`), or to running processes (`/proc/PID`). All entries can be easily extracted using standard file operations (e.g. using `fopen` or the command-line tool `cat`).

Low-level information about network interfaces can be found in `/proc/net` and `/proc/sys/net` although it is usually more convenient to execute the command `ifconfig` which provides a summary of the relevant information. Many home routers have an integrated Ethernet switch. If VLANs are configured on this switch, the settings can be found in the `/proc/switch/` subtree of the proc-filesystem. This information is useful to diagnose connectivity problems at the LAN interfaces.

The connection tracking (conntrack) function of the netfilter kernel module tracks TCP connections and other IP packet flows traversing the router. A list of currently active connections and flows can be obtained from `/proc/net/nf_conntrack`. As an advantage over pcap-based tools such as `tcpdump`, conntrack-based traffic monitoring requires little computational resources.

B. *Configuration and Status Files*

Just like on Linux PCs, OpenWRT status and configuration files are usually located in the `/etc` directory while files with status information can be found in `/var`. The availability depends on the installed services and applications. The name and location of a specific file may vary between different router firmwares.

On OpenWRT, configuration files are located in the `/etc/config/` directory. As an example, the DHCP configuration is stored in `/etc/config/dhcp`. This file contains the interfaces for which DHCP is enabled. The list of active DHCP leases in the file `/var/dhcp.leases` gives an overview on currently connected devices, including their IP and MAC addresses. Interface and routing parameters are stored in `/etc/config/network`; firewall settings are located in `/etc/config/firewall`. WLAN settings, including possible passwords, are saved in `/etc/config/wireless`. This information is needed to configure new devices in order to connect them to the wireless network.

C. *Output of Command-line Tools*

Some volatile information, such as interface statistics and the list of connected WLAN clients, cannot be obtained from a file but requires the execution of a command-line tool. In the given examples, appropriate tools are `ifconfig` and `wl`[1], respectively. The error statistics returned by `ifconfig` allow detecting hardware problems, e.g. broken Ethernet cables.

Command-line tools can also be used to actively scan the network. For example, we can test the reachability of a certain host by examining the output of the `ping` command.

IV. Approach and Implementation

As we have seen in the preceding section, useful configuration and status data about the network is spread over various sources on the home router. Our goal is to summarize the relevant information in a single log file which, in the case of network problems, can be given to a trusted remote helper, such as an experienced friend or a professional help desk.

The data collection and log file generation is performed by a data collection agent installed on the home router. This step is shown in the upper box of Figure 1. More information about the log file generation is given below in Section IV-A.

There are various options how the log file can be transferred to a potential helper. In all cases, unauthorized access should be prevented because the file may contain personal and sensitive data, such as public IP addresses and pre-shared keys for WLANs. As shown in Figure 1, we protect the log file by encryption. Thus, access to the content of the log file is restricted to those persons who possess the valid credentials for decrypting it. Section IV-B describes different transport options and related security issues in more detail.

The remote helper needs to analyze the decrypted log file in order to identify the causes of the network problem and provide adequate advise. Error diagnosis based on the log files can be partly automated using appropriate tools. Some ideas on this topic are presented in Section IV-C.

Finally, Section IV-D presents details about our prototype implementation.

A. *Log File Generation*

All data sources identified in Section III consist of human-readable text. The simplest way to generate a log file is to just copy text from different sources one behind the other. A more

[1]Note that the availability of the `wl` tool depends on the router hardware.

TABLE I
Examples of useful network information (Linksys router running OpenWRT 8.09)

Wireless information	Data source
Nearby APs	cmd: `iwlist wlan0 scan`
Connected clients	cmd: `wl assoclist`
Encryption	cmd: `iwlist wlan0 encryption`
Frequency	cmd: `iwlist wlan0 frequency`
SSID and password	file: `/etc/config/wireless`
Network information	**Data source**
DSL status	file: `/proc/avalanche/avsar_modem_stats`
Interfaces, routing	file: `/etc/config/network`
Active hosts	file: `/proc/net/arp`
DHCP	file: `/tmp/dhcp.leases`
Active TCP/UDP connections	file: `/proc/net/nf_conntrack`
Packet counter	file: `/proc/net/dev`
Sockets	file: `/proc/net/sockstat`

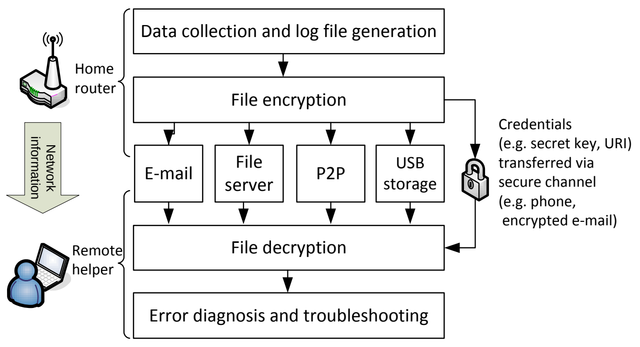
Home router
Data collection and log file generation
File encryption
Network information
E-mail
File server
P2P
USB storage
Credentials (e.g. secret key, URI) transferred via secure channel (e.g. phone, encrypted e-mail)
Remote helper
File decryption
Error diagnosis and troubleshooting

Fig. 1. Steps showing the approach of our system

Listing 1. Example log file generated by the data collection agent

```
<interfaces>
  <if-name>eth0.0</if-name>
  <mac-addr>00:1E:E5:84:E5:D0</mac-addr>
  <if-name>eth0.1</if-name>
  <mac-addr>00:1E:E5:84:E5:D0</mac-addr>
</interfaces>
<wifi>
  <interface>wlan0</interface>
  <channel>5</channel>
  <mode>ap</mode>
  <ssid>monitor-ap</ssid>
  <encryption>psk</encryption>
  <key>mykey</key>
</wifi>
<dhcp>
  <lease1>00:16:D3:29:63:99 192.168.1.132</lease1>
  <lease2>00:16:D3:12:44:63 192.168.1.133</lease2>
</dhcp>
```

sophisticated mechanism parses the data sources, extracts the relevant pieces of text, and uses them to compose a log file in a unified format. We decided to implement the second solution and to use an XML-based log file format.

In order to be flexible, the parsing is performed with configurable regular expressions and capturing groups. The deployed regular expressions need to be well adapted to the text format of the data sources. As an example, the following listing shows the output of the ifconfig command issued on OpenWRT:

```
eth0.0    Link encap:Ethernet  HWaddr 00:1E:E5:84:E5:D0
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask
               :255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:169 errors:0 dropped:0 overruns:0 frame:0
          TX packets:178 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:17330 (16.9 KiB)  TX bytes:144891 (141.4 KiB)

eth0.1    Link encap:Ethernet  HWaddr 00:1E:E5:84:E5:D0
          inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1346 errors:0 dropped:0 overruns:0 frame:0
          TX packets:120 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:214525 (209.4 KiB)  TX bytes:14481 (14.1 KiB)
```

In order to extract the interface names together with their hardware addresses, the following regular needs to be configured:

```
(\w+)[ \t]+Link[ \t]+encap:\w+[ \t]+HWaddr[ \t]+([a-eA-E0-9:]+)
```

The contents of the capturing groups are sequentially written to the log file between the start and end tags of XML elements. An example log file is shown in Listing 1. The child nodes of `<interfaces>` are the result of the regular expression described above.

The configuration of the data collection agent encompasses the data sources to be considered, the regular expressions applied to them, and the names of the resulting XML elements. Hence, our data collection agent can be easily customized for different home routers and different firmware version. Furthermore, we can configure the agent to use the same log file format for different router models, which facilitates tool-based processing and analysis of the files (see Section IV-C).

We do not expect the average inexperienced home user to configure the data collection agent from scratch. Predefined configuration templates for different routers and firmwares can be made available for download from a server or provided as part of the router software packages (e.g. in the OpenWRT repository). Control of the data collection agent should be integrated into the web interface of the router, offering the possibility for the user to apply configuration changes, for example with respect to the information included in the log file.

The data collection agent periodically updates the log file at a configurable time interval. After each update, it triggers an external process which takes care of the encryption and file transfer as described in the next section.

B. Transport Options

There are several ways to get the log file out of the home router and to transfer it to a place where a remote helper can access it in the case of a problem. If the Internet connection is functional, we can send the file in an e-mail, upload it on a file server, or push it in a P2P network. We do not need to transmit an updated log file if the content has not changed compared to the last transmitted version of the file. On the other hand, the transmission of new log files shows that the data collection and transmission process is working correctly. As long as there are no resource limitations, we suggest to periodically send or upload new log files.

If the primary Internet connection fails, a backup channel may be provided by a mobile data connection. For example, some routers already support 3G USB modems out of the box. As a last resort, the log file can be saved on a USB memory stick which is plugged into the home router. In the case of a problem, the memory stick can be physically provided to the helper, or the saved log files are electronically transfered using any computer with a working Internet connection. Saving the current log file on the stick can be triggered by pressing a "panic button" on the router.

Since a log file may contain sensitive and security related data, it must be protected against unauthorized access. To which extent additional data protection measures are needed to achieve this goal depends on the transport mechanism and the trustworthiness of any involved external server. Effective protection can be achieved by encrypting the file using a symmetric key which is shared between the home user and the remote helper. The key can be shared in advance or after a network problem has occurred. As shown in Figure 1, the key should be given to the trusted helper via a second (secure) channel, such as by phone or in an encrypted e-mail. For this purpose it would be very useful to have an infrastructure in place which allows establishing and maintaining trust relationship between different home networks, such as the one presented in [1] and [12].

In the following we discuss some of the transport options in more detail. All of them require a working Internet connection.

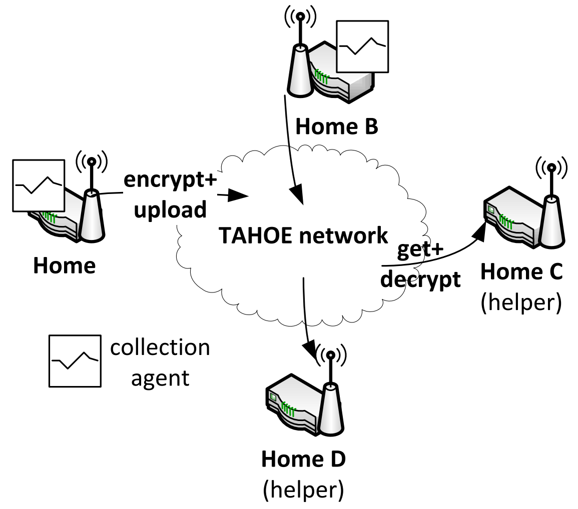

Fig. 2. How we use the TAHOE network to store and retrieve logging information

E-mail: An easy way to share the log files is to send an e-mail to a mailbox. It is recommended to encrypt the files because e-mails are not always transfered in an encrypted way.

File Server: The log files can be uploaded to a file server located in the Internet, for example using FTP or SCP. With appropriate access control mechanisms on the server, it can be ensured that access to the file is restricted to the trusted helper. An effective solution is to use a file repository supporting version control, such as subversion [15]. This allows tracking changes between subsequent versions of the log file. If transport layer security is enabled during file transfers, additional file encryption can be omitted.

P2P: The log files can also be uploaded into a P2P network. For example, TAHOE [18] is an open-source implementation of a distributed P2P file system. Uploaded files are split into chunks which are stored on a large number of participating nodes. The chunks contain redundant information to ensure that the original files can be retrieved even if a certain proportion of the nodes is not available. In order to protect the files against interception, all data chunks are encrypted before being uploaded. Once a file is uploaded to the TAHOE network, the owner gets a unique URI that consists of the ID and the key for the file. Without this URI, it is not possible to retrieve and restore the file.

Figure 2 shows an integrated troubleshooting solution based on TAHOE and our data collection agent. Home routers run TAHOE clients in order to upload the generated log files into the P2P network. Since the TAHOE client uploads encrypted data chunks, no additional file encryption is needed. The home user must keep the corresponding TAHOE URI a secret and only reveal it to the trusted helper.

C. Error Diagnosis and Troubleshooting

The information contained in the log file helps experienced friends or help-lines to identify the causes of network problems, such as misconfigurations or hardware failures. Besides checking the configuration and status parameters for plausibility, it is useful to compare to current log file with an earlier version generated before the problem occurred. Differences between two XML files can be efficiently identified by using tools like `xmldiff` [19]. The comparison to an earlier log file shows which parameters have changed and thus are potentially related to the problem. Depending on whether the Internet connection is working or not, troubleshooting can be started by providing instructions via phone or e-mail, or by using remote access to a computer in the home network.

Expertise of user communities often helps to find a solution to a problem. Community driven knowledge databases, such as NetPrints [6] could be established or extended to analyze log files generated by our data collection agent. Thus, if a problem occurs, it would be possible to find similar problems reported by other users in the past, together with valuable troubleshooting information. The lookup of earlier incidents could be assisted by data mining methods which identify characteristic patterns in the log files and associate them to specific types of network problems.

D. Prototype Implementation

We implemented the data collection agent as a small customizable C program and released it as open-source software. [10]. Data sources, regular expressions, XML elements written in the log file, and update period are specified in a configuration file. After each update of the log file, an external program can be called to take care of the file transfer.

Since the program depends on standard C libraries only, it can be easily compiled for different hardware architectures, such as standard Linux PCs and embedded Linux routers. We used the OpenWRT SDK to cross-compile the source code for a Linksys WRT54G router, which was patched with the OpenWRT firmware [13]. Alternatively, the data collection agent can be built for many other Linux based routers and therefore pushed to existing routers. Firmwares such as DD-WRT [5] and Tomato [17] support a large number of routers from different manufacturers and can be easily equipped with our tool. It also runs on the AVM Fritz Box, the most popular home router in Germany.

We assessed the impact of the data collection agent on the packet forwarding performance. Therefore, we connected two PCs to a Linksys WRT54G router and run `iperf` [9] to measure the maximum TCP transmission rate between the two PCs. The `iperf` server was connected to the WAN port of the router while the `iperf` client was connected to an internal LAN port. With this setup, all packets needed to be processed by the Linux kernel of the router. The router was configured to perform Network Address Translation (NAT) between the LAN and the WAN port.

With the data collection agent disabled, we obtained a throughput of 32.7 Mbps as an average of ten measurements (see Table II). Since the link speed was 100 Mbps, the measurement shows that the throughput is limited by the processing resources of the router.

TABLE II
AVERAGE IPERF THROUGHPUT FOR DIFFERENT CONFIGURATIONS

Update interval	disabled	10s	1s
sendmail export	32.7 Mbps	32.4 Mbps	32.2 Mbps
scp export		32.2 Mbps	32.1 Mbps

Thereafter, we started the data collection agent with a configuration to collect information about the current date and time (command `date`), the CPU load (file `/proc/loadavg`) and memory utilization (command `free`), the network interfaces (command `ifconfig` and file `/proc/net/dev`) and open sockets (file `/proc/net/sockstat`), and some wireless parameters (command `iwconfig`). The size of the generated log file was approximately 5.6 KByte. Using the `sendmail` command, the file was periodically sent by e-mail to an SMTP server located in the WAN port network. Table II shows the average `iperf` throughput measured for update intervals of 10 seconds and 1 second. As we can see, the throughput slightly decreases to 32.4 Mbps and 32.2 Mbps, respectively. This decrease can be partly explained by the fact that the log file is transmitted through the same interface as the `iperf` traffic, accounting for 49 Kbps of outgoing traffic at an update interval of 1 second. Further impact likely goes back to the CPU load caused by the execution of the data collection agent.

In a second test, we used `scp` to copy the log file in an encrypted way to a remote SSH server located in the WAN port network. The `scp` command installed on the router is part of the SSH suite dropbear. We configured `scp` to use public private key certificates (RSA) for authentication. Due to the establishment of an encrypted channel for each log file transfer, we expected a higher CPU load and consequently a significantly lower throughput than in the setup using `sendmail`. However, when exporting the log file every second, we still obtained an average `iperf` throughput of 32.1 Mbps. Hence, encryption has a very small additional impact on the router throughput, only.

With these measurements, we can state that the influence of the collection client on the packet throughput is negligible. For practical deployment, we suggest that an interval of several seconds is sufficient, and that further decreasing the interval does not actually generate more useful information for error diagnosis and troubleshooting.

V. CONCLUSION

In this paper we identified the home router as a very valuable source of information for network error diagnosis and troubleshooting. Since the home router is the center of the home network, it holds a wide range of network state and configuration data which cannot be easily obtained from other devices. We developed a data collection agent which collects useful information from various data sources on the router and writes it into a single XML log file. Thanks to this log file, the home user is able to provide a detailed description of the home network to a remote helper in the case of network problems. The utilization of secure transport protocols and encryption techniques ensures that the log file is protected against illegitimate access.

As a main application area, we see communities of home users sharing questions and answers regarding computer and network problems. Today, these communities make use of web forums, mailing lists, and social networks. We expect that the information in the generated log file would enable experienced users to provide better support. In addition, we can store log files, problem descriptions, and possible solutions in a knowledge database, such as NetPrints [6]. If a network problem occurs, we can use the log file to search the database for similar problems and information about how these have been solved in the past.

ACKNOWLEDGMENT

The authors would like to thank Kameliya Terzieva who helped to develop and to implement the data collection agent.

REFERENCES

[1] A. Müller and H. Kinkelin and S.K. Ghai and G. Carle, “A secure service infrastructure for interconnecting future home networks based on DPWS and XACML,” in *Proc. of ACM SIGCOMM Workshop on Home Networks (HomeNets) 2010*, New Delhi, India, Sep. 2010.
[2] D. Ashby and R. Khasawneh, “An Analysis of Home Computer Customer Service Hotlines,” *International Management Review*, vol. 4, no. 2, 2008.
[3] Broadband Forum, Technical Reports, http://www.broadband-forum.org.
[4] K. Calvert, K. Edwards, N. Feamster, R. Grinter, Y. Deng, and X. Zhou, “Instrumenting Home Networks,” in *Proc. of ACM SIGCOMM Home Networking Workshop*, New Delhi, India, Aug. 2010.
[5] DD-WRT Homepage, http://www.dd-wrt.com.
[6] B. A. et.al, “NetPrints: Diagnosing Home Network Misconfigurations Using Shared Knowledge,” in *Proc. of USENIX Symposium on Networked Systems Design and Implementation (NSDI) 2009*, Bosten, MA, USA, Apr. 2009.
[7] E. S. P. et.al, “More Than Meets the Eye: Transforming the User Experience of Home Network Management,” in *Proc. of ACM Conference on Designing Interactive Systems (DIS) 2008*, Cape Town, South Africa, Feb. 2008.
[8] M. C. et.al, “Who’s hogging the bandwidth: the consequences of revealing the invisible in the home,” in *Proc. of International conference on Human factors in computing systems*, Atlanta, GA, USA, Apr. 2010.
[9] iperf Homepage, http://sourceforge.net/projects/iperf/.
[10] LInEx Homepage, http://vermont.berlios.de/linex.
[11] A. Mueller, “NAT-Tester,” http://nattest.net.in.tum.de.
[12] A. Müller, H. Kinkelin, S. Ghai, and G. Carle, “An Assisted Device Registration and Service Access System for future Home Networks,” in *Proc of IEEE IFIP Wireless Days 2009*, Paris, France, Dec. 2009.
[13] OpenWRT Homepage, http://openwrt.org.
[14] L. Socher, “Managing the digital home,” Accenture, Communications Industry Group, 2008.
[15] Subversion Homepage, http://subversion.apache.org/.
[16] Tcpdump Homepage, http://www.tcpdump.org/.
[17] Tomato Homepage, http://www.polarcloud.com/tomato.
[18] Z. Wilcox-O’Hearn and B. Warner, “Tahoe: the least-authority filesystem,” in *Proc. of ACM international workshop on Storage security and survivability (StorageSS) 2008*, Fairfax, VA, USA, Oct. 2008.
[19] XMLDIFF Homepage, http://www.logilab.org/project/xmldiff.