RJ10374 (C0505-002) May 6, 2005
Computer Science

IBM Research Report

CPWCT: Making P2P Home Network Secure Virtual Multimedia Device

Yu Chen Zhou, Julian A Cerruti*, Lin Ma, Lei Ma
IBM China Software Development Laboratory
Beijing 100085
China

*IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

CPWCT: Making P2P Home Network Secure Virtual Multimedia Device

Yu Chen Zhou* Julian A Cerruti† Lin Ma* Lei Ma*
*IBM China Software Development Laboratory
100085, Beijing, China
{zhouyuc, mallin, malei}@cn.ibm.com
†IBM Almaden Research Center
95120, San Jose, CA, USA
jcerruti@us.ibm.com

Abstract

Content management and protection are important features of emerging home network technologies. In this paper, we present the design of Content Protection for Workplace Client Technology (CPWCT), a novel secure content management system based on the multiagent paradigm that transforms a P2P home network in a secure virtual multimedia device. In CPWCT, devices in a home network are organized in a secure P2P cluster leveraging the broadcast encryption based xCP Cluster Protocol. Based on this torus, a distributed content management mechanism for purely P2P home networks is implemented. This mechanism includes lightweight secure streaming and RDF based capability-aware content indexing technologies. We show how this approach can significantly enhance the ability to dynamically self-reorganize the underlying topology and deploy context sensitive functionality.

1. Introduction

The widespread availability of audio and video in digital form together with the increasing ability to easily share content among consumer electronics devices promises to create a new market for multimedia home networking. In this environment where content such as music and movies is no longer bound to the physical media that carries it, content protection is a critical feature. This presents new challenges to content owners, i.e. record labels, studios, distribution networks and artists, who want to protect their intellectual property from indiscriminate reproduction and distribution in order to extract economic benefit from it, while presenting a non-invasive user experience for the consumer. Legal actions against infringes may dissuade others from accessing or manipulating content illegally, but technical measures usually provide a more effective means for limiting abuse. The objective of a content protection scheme is to raise the barrier for casual violations and to require a concerted effort by attackers. Digital rights management allows the content owners to define and enforce restrictions on how the content is used. In order for a content protection scheme to be successful, it must be cost-effective to implement and run. At the same time, it must be unobtrusive to consumers, who do not want to be burdened with administrative tasks associated with protecting the interests of content owners.

The home network presents the following challenges in the area of distributed content management and protection: (1) decentralized P2P environment with devices connecting and disconnecting dynamically; (2) heterogeneous structure that consists of diversified devices (e.g., set-top box, PC, PDA) with different system and media capabilities and; (3) distributed storage of content.

In this paper, we present the design of Content Protection for Workplace Client Technology (CPWCT), which includes novel and comprehensive approaches for secure content dissemination and protection in a P2P home network, leveraging the eXtensible Content Protection (xCP) technology [1] based on broadcast encryption. Due to the Multiagent System (MAS) computing model's similarity to the P2P computing model in terms of common use of a distributed environment, we applied the MAS paradigm to implement a CPWCT model with 3 layers: secure clustering, secure content dissemination and dynamic content indexing. We show that this approach can significantly enhance the ability to dynamically self-reorganize the underlying topology and deploy context sensitive functionality. The final target of CPWCT is to turn a P2P home network into a secure virtual multimedia device for the end user. The structure of CPWCT is shown in figure 1. In this environment, no matter which device and network connection is used or whether the user is located in her home or

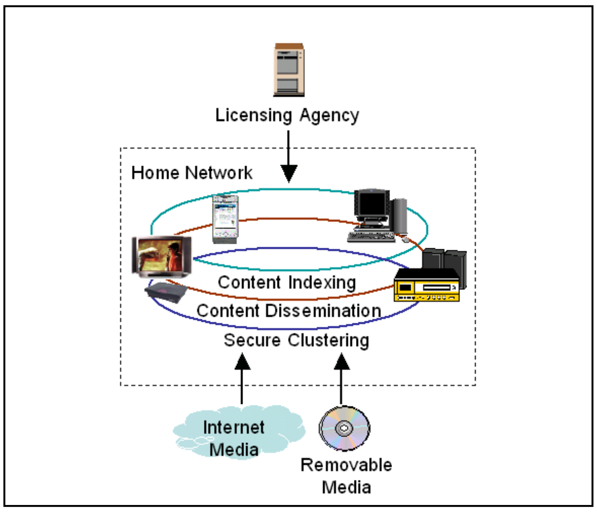

Figure 1. CPWCT system in home network

traveling, she is able to enjoy all the applicable multimedia content securely within the domain of her entertainment electronic devices.

The remainder of the paper is structured as follows: section 2 presents overview of the CPWCT architecture and CPWCT agents; section 3 introduces the xCP cluster protocol, the torus of CPWCT for P2P authorization and clustering based on broadcast encryption technology; section 4 discusses the distributed P2P content management and introduces a lightweight xCP based media streaming mechanism; section 5 describes a novel approach of capability-aware P2P indexing mechanism leveraging W3C RDF; and section 6 provides the main conclusions of this work.

2. Overview of CPWCT

The P2P architecture of a consumer electronics home network presented several challenges for the design of CPWCT. In the absence of a centralized mediator with global knowledge that directs requests to appropriate agents, CPWCT agents must cooperate to forward queries and merge and return results.

A group of CPWCT agents can form an xCP cluster, which defines the social context in which the agents interact. Each CPWCT agent joins an xCP cluster, acquires the commitments from other peers, and provides commitments according to its own capabilities. The behavior of CPWCT can be described as social behavior with both collaboration and competence.

The CPWCT agents collaborate in tasks that include clustering, content dissemination and indexing. A CPWCT agent is active rather than reactive: It actively contacts other agents to join the xCP cluster and share local content among trusted peers. An xCP cluster could be abstracted as a set *Cluster={A, S},* where *A* is a trusted agent set in the xCP cluster, and *S* is the cluster context. Extending the BDI model [3], a CPWCT agent could be defined as a set *A=(Bel, Int, Cap, Act)* which consists of believe, intention, capability and action set.

Generally, the intentions of CPWCT agents are to achieve better the goals of the whole cluster to protect content and that of themselves to share the content. We define *Int = {Ic, Is, Ii}* to be the set of intends of a CPWCT agent, including those for creating or joining xCP cluster, disseminating xCP content and sharing content information. *Int* will be dynamically updated according changes of *Bel, Cap*, and *S*.

$$P(Bel) \times P(Int) \times P(Cap) \times P(S) \xrightarrow{E(s)} P(Int)$$

There are the following decencies among the intention sets; this means that content could only be shared to trusted peers and only the index of shared content could be provided to peers.

$$index\ (a) \in I_i \mapsto share\ (a) \in I_d \mapsto trust\ (a) I_s$$

According to the tasks of CPWCT, $Bel = \{B_c, B_d, B_i\}$ are the knowledge and rules maintained by each CPWCT agent and will be updated dynamically according to changes of *S*. B_c is the believe for secure clustering which will be described in chapter 3. B_s is the knowledge of local media for content dissemination. B_i contains the knowledge and rules for content indexing that is described in Chapter 5

Being the core of *Bel,* B_c is represented by a set $\{B_{secinfo}, B_{cluster}, B_{authrule}, B_{comm}\}$. $B_{secinfo}$ is the secret information for P2P authorization. To monitor the whole xCP cluster, each CPWCT agent maintains $B_{cluster}$, which contains the dynamic cluster topology that is a cut of both logical and physical status of CPWCT agents in the cluster. $B_{authrule}$ is the set of rules for a CPWCT agent to authorize peers. The consistency of $B_{authrule}$ in each different CPWCT agent is critical to maintain a consistent state of the system against concurrent distributed transactions. B_{comm} is the knowledge of agent communications based on xCP cluster protocol.

In a heterogeneous home network, the capabilities of a CPWCT agent depend on those of device where it resides and dictates what this agent can commit to the cluster. These capabilities are represented by $Cap=\{C_{med}, C_{sys}\}$. The media capabilities C_{med} include media formats and input modes supported by media players plugged in the CPWCT agent, size of the available display, capability of the audio device, etc. The system capabilities B_{sys}, include the size of free local storage, network bandwidth, CPU capability, free memory size, etc.., *Cap* is a critical component for a CPWCT agent to choose which actions to take for content dissemination and indexing.

The cluster context is represented by a set $S = \{S_1, S_2, \ldots S_n\}$ of environment variables which represents the status of all the CPWCT agents in an xCP cluster. A change of state is triggered by a set of events, and it in turn causes the generation of another set of events. The sequence of changing states can be stated as follows.

$$S_1 \xrightarrow{E_1} S_2 \xrightarrow{E2} \ldots S_{n-1} \xrightarrow{E_{n-1}} S_n$$

The action set *Act* can be executed according to believes, intends and capabilities of each CPWCT agent and triggered by events representing changes in the cluster context

$$Execution\ : P(Bel) \times P(Int) \times P(Cap) \xrightarrow{E(s)} A$$

Coordination is also a main task for CPWCT which is forced to adopt distributed control and data in the P2P environment. This makes the agents have a higher degree of autonomy in generating new actions and in deciding which goals to pursue. The disadvantage is that the knowledge of the system's overall states is dispersed throughout the system and each agent has only a partial and imprecise perspective. There is an increased degree of uncertainty about each agent action, so it is more difficult to attain coherent global behavior. In CPWCT, great efforts have been made to synchronize the overall state information through perceptions (receiving message) and actions (sending message), especially for B_c, B_i and I_c, in the clustering and content indexing layer that is described in sections 3 and 5.

The CPWCT Agent is implemented as an OSGi compliant middleware component in Java with a footprint of 372 KB. It is easy to deploy on both legacy and future devices in the home network. It implements a flexible structure shown in figure 2 with following layers: (1) Communication Service, which is responsible for inter-agent communication and provide SPI to integrate multicast UDP, HTTP, Web Services, UPnP or other

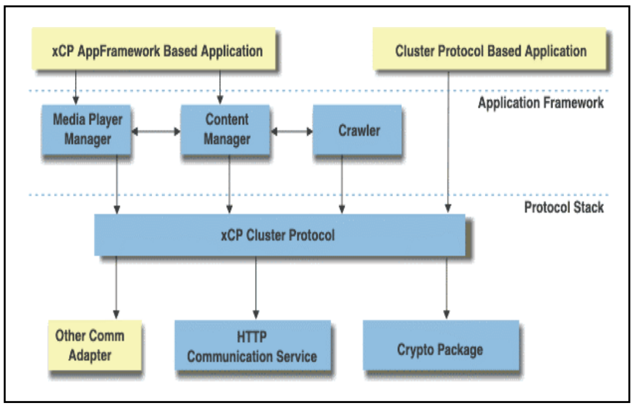

Figure 2. Structure of CPWCT agent

proprietary network protocols for device discovery, messaging and transportation of content data; (2) xCP Cluster Protocol Stack, which implements the core logic for secure clustering, and provides functions for up-level components to encrypt and decrypt xCP content and monitor the status of the xCP cluster and; (3) Application Framework, which consists of Content Manager, Media Player Manager and Crawler that allow various media players, repository services and customized UI components to be integrated into CPWCT.

3. Broadcast Encryption Based Secure Clustering

The secure clustering layer of CPWCT is an implementation of the xCP Cluster Protocol, introduced and presented in detail in [1], [3], [4] and [5]. In a nutshell, this is a broadcast encryption application that allows a group of devices in a P2P network to form an "authorized domain", agreeing in a common, unique cryptographic key. This key is used to protect all the content stored in this network of devices thus binding it to the group of devices instead of the single physical container that carries it. In this chapter, a high-level overview of this technology with particular emphasis on the CPWCT implementation is presented.

3.1 Broadcast Encryption

Broadcast Encryption (BE) is a key management technique introduced by Fiat and Naor in [7] which can be used as an alternative to traditional PKI. It was originally designed for applications where only a unidirectional broadcast channel is available and there is no chance for two way authentication for key distribution, hence its name.

BE is particularly appealing for applications where limited processing power is available because it can be fully implemented using symmetric key encryption algorithms, which is orders of magnitude more efficient than asymmetric ciphers such as RSA. With the latest advances in this area, the overhead in message length has became roughly the same size of that of a PKI exchange with a CRL, making this technology the best choice for consumer electronics applications.

3.1.1 Main components of a BE scheme

At system creation time, a group of long-lived keys are generated. These keys, which will be referred as device keys are grouped into unique groupings called device key sets (DKS) and assigned to each device that will participate in the system. In other words, each device instance is assigned a unique long-lived DKS at manufacturing time.

When a message needs to be transmitted – or a piece of content recorded – a random management key[1] (K_m) is chosen by the sender and a block of data called key management block[2] (KMB) is generated. In a first approach, the KMB can be though of as the encryption of the K_m multiple times using all the originally generated device keys. When a device receives a KMB, it will try to find an instance of the K_m that has been encrypted using a device key it has in its DKS and then it will be able to decrypt it. In BE terminology, this is to say that the receiver "processes" the KMB using its own DKS, getting a K_m as a result.

If the sender wants to exclude a group of devices from receiving the message, a new KMB is generated but this time the encryption of K_m with the device keys of the excluded devices is replaced with garbage information. Now, when a receiver processes the KMB using its DKS the result will be either the K_m or an invalid value, depending on whether it has been included in the transmission or not.

The message in question is of course encrypted by the sender and decrypted by the receiver using K_m as a session key for this transmission. Figure 3 depicts this whole idea, where receiver 2 is excluded from a transmission.

3.1.2 Real-Life BE Schemes

At this point a reader might rightfully ask: "But how long does this KMB need to be in a real world system? How big is each receiver's DKS?" These are the very questions that the inventors of new broadcast encryption systems have in mind while working on a new scheme, making their objective to make the answer to both questions a number as small as possible. A better or worse result will be reached depending on how are the device keys first generated and how are they

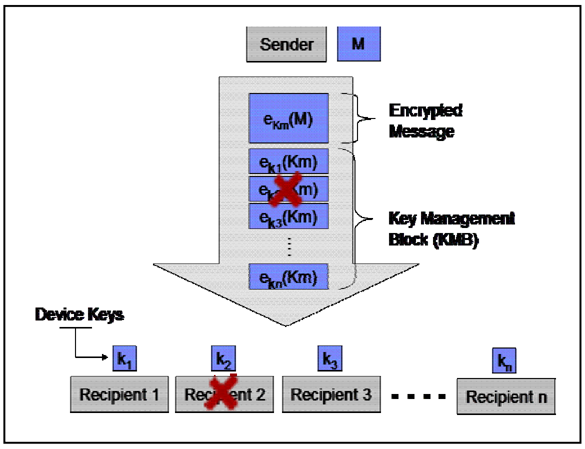

Figure 3. Trivial broadcast encryption scheme

[1] Some media-based applications of BE use the name Media Key (Km) for the same concept

[2] Some media-based applications of BE use the name Media Key Block (MKB) instead for this very same concept

grouped when assigning them to each receiver. Other topics that have to be taken into considerations are how the size of the KMB changes when more and more devices are to be excluded and what is the impact of a massive device exclusion or inclusion in the size of the KMB.

The scheme described and shown in Figure 3 is of course only provided as a means for depicting the idea behind the main concepts but it is never used in practice. There are several broadcast encryption schemes, some of which are currently being used in real life applications. Examples of this is the content protection for recordable media (CPRM) and content protection for pre-recorded media (CPPM) technologies [9], which use a two-dimensional matrix to layout the device keys and the logical tree hierarchy (LKH) [10], [11] which use a tree-based approach.

The BE scheme used in the xCP Cluster Protocol within CPWCT is a Naor-Naor-Lotspiech (NNL) scheme. This scheme named after their inventors is fully described in [7]. Some of its highlights are:

- It achieves an average efficiency of 1.38 messages per revocation, resulting in a KMB roughly the same size as a PKI certificate revocation list
- It allows very efficient group revocation of DKS (i.e.: revoke all the devices of a given model, increasing the KMB size only by one)
- Revocation is completely granular. This is to say, no matter how many revocations are there on a KMB, there will be no innocent devices revoked

All the basic cryptographic operations used for the xCP Cluster Protocol in this system are implemented using AES in CBC mode in most cases and ECB mode in a few exceptional ones.

3.2 Other Content Protection System Elements

A key concept when using BE to design a content protection system is the concept of device compliance. Along with the key management system described above goes a set of rules that dictate how devices should act under different conditions – i.e.: how to interpret the usage conditions associated to a given multimedia object. In other words, the system consists of a set of agents that have keys to access the content but are trusted to comply with the rules defined for the system. These trusted agents are said to be compliant with the system, because they play by the rules.

When the system is first launched all devices are assumed to be compliant and thus any KMB used to transmit a message should include all devices in the system. Nevertheless, it could happen that one of the devices is attacked, its DKS extracted and an application written to use that DKS and ignore the rules that dictate the protection of the content. Or, an originally compliant device could be found to be ill-behaved and allow a user to circumvent some of the restrictions associated with the content through careful device manipulation. In those cases, these non-compliant circumvention devices have to be excluded from further transmissions by excluding their DKS from calculating K_m in all future KMBs. This operation will be referred as device revocation. That is to say, the DKS of the circumvention devices have been revoked in the KMB.

This notion of device compliance is usually enough for the purposes of a content protection system in which licensors of the content that is distributed are not concerned with the particular identity of each receiver; it is enough for them to rest assured that only compliant devices can access the content.

Another concept that builds on top of compliance is the concept of content binding to a particular entity. This idea of making the cryptographic calculation used to access the content dependent on the unique identifier of an object has been used by content licensors to control how the content can be moved and copied between entities through compliant devices.

A common example of the use of this concept is CPRM and its ability to bind content to the media where it will be played back. When a compliant recorder makes a protected recording into a media, for example, a recordable DVD, the unique ID of that particular DVD is used as part of the cryptographic calculation performed to encrypt the content:

$$K_{mu} = [C2\_G(K_m, ID_{media})]$$ [3]

The media unique key (K_{mu}) will be used to encrypt the title key (K_t) used in turn to encrypt the content. ID_{media} is of course written in an area of media where it is not possible to be altered by the user.

This way, because only compliant devices can get to K_m, rules can be dictated by the content licensors as to when is it allowed to move or copy the content to another piece of media – which will require a re-encryption of K_t with a new K_{mu} . This gives the content owners the control on the copies of protected content that we were speaking about.

3.3 xCP Cluster Protocol

With the latest advances on consumer electronic technologies it is very natural that consumers want to take full advantage of the networking and storage capabilities of their devices to enjoy the entertainment content they have acquired in any of the devices they own.

In this environment, placing usage restrictions on where the user can store the content – i.e.: in a particular piece of media, or on a particular device – doesn't seem natural. With this concept in mind IBM has developed the xCP Cluster Protocol that provides a means for protecting content from indiscriminate redistribution without precluding the user from enjoying the content freely within the domain of her electronic devices.

The main idea behind this technology is that devices will use BE to form a cluster of a limited number (*N*) of compliant devices that share a common ID. This ID is used as the identifier for a particular household thus, when the content is bound to this cluster, it is equivalent to binding the content to the household instead of to a particular device or piece of media. The content will be playable in any of the devices owned by that household, but it will not play in other people's devices. By limiting the number of devices to a given finite number *N*, the forming of a single global million device (Inter)network is

[3] C2_G represents a C2-based cryptographic one-way function. See [9] for details.

avoided.

In CPWCT, a simplified version of the xCP Cluster Protocol technology has been implemented. Without getting into the details of the protocol – which can be found in [1], [3] and [4] – the particulars of the implementation used in CPWCT are outlined below.

3.3.1 xCP Cluster Components

Each xCP-capable device has a set of device keys and a unique device ID (ID_p) assigned by the licensing entity. Also, each device comes with its own singleton cluster already prepared so the first time a device is turned on it is already part of a new cluster consisting of a single device.

An xCP Cluster consists of a cluster KMB, unique cluster ID (ID_c) and a file that lists the devices that form part of the cluster – called authorization table (AT). This way, the default cluster that comes pre-made on each device has its ID_c, a default KMB and an AT consisting of the single device. This is depicted in Figure 4 below.

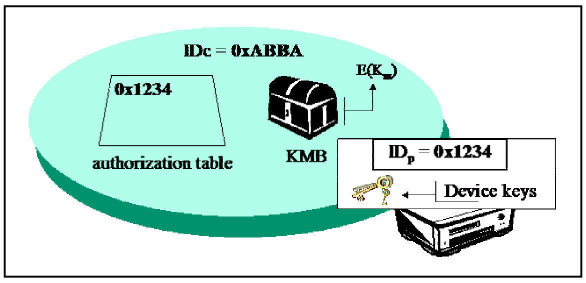

Figure 4: xCP Cluster Components

In order to bind the content to the cluster, when a piece of content is introduced in a cluster, its *Kt* is encrypted using a cluster unique key or binding key (K_b), calculated as

$$K_b = AES\_G(K_m, ID_c / Hash(AT))$$ [4]

As the cluster evolves and devices join the cluster, the contents of the AT will change. Also, if a new KMB is introduced into the system, devices have the obligation of adopting it, changing the value of K_m. In any of those cases, the value of K_b changes, requiring a re-encryption of the title keys of content stored in the cluster. In order to keep track of which are the AT and KMB that have been used to encrypt a group of title keys, a reference to them are kept in the header of each xCP file.

3.3.2 xCP Device Communication

During the implementation of CPWCT the communication of messages between devices has been moved to a separate communication layer, as show in figure 2 above, allowing the main logic of which messages need to be transferred be agnostic of the particular transport protocol being used. This communication layer provides two main services to the upper layer: (1) a device discovery service, which notifies the device whenever another xCP compliant device has been discovered in the network and; (2) a standard point-to-point communication service, addressable by device ID

The first service maps well with the "Who's there" message mentioned in [1], [3] and [4], while the second one can service all the rest of the xCP Cluster Protocol messages ("I'm here", "Authorize me", etc.)

In this first approach, CPWCT implements a communication service based on HTTP. The discovery service is implemented by listening for multicast messages on a defined port. When a device appears in the network, the device discovery service recognizes this multicast message and informs the device that a "Who's there?" message has been received.

In a future release, this same service can be provided using the underlying device discovery mechanisms built-in the physical transports, such as plug-n-play (i.e.: a portable player is connected via USB to a home media server), uPnP, etc.

3.3.3 Messages and Device Behavior

Whenever a device receives a "Who's there?" message indicating that there is a new device in the network, it must reply with a point-to-point "I'm here" message. This message contains ID_c and also the hash of the cluster's current KMB and AT.

If the discovered device is a new device the user has just connected into the network, it will not have any information about the cluster mentioned in the "I'm here" message. In that case, the receiving device will try to get authorized in the existing cluster ID_c, using the authorization procedure exactly as described in [1], [3] and [4]. At the end of this procedure, if the predefined maximum cluster size has not been reached, the cluster will consist of an additional device.

Each device is also responsible for sending "I'm here" periodically to all the rest of the devices that are already in the cluster in order to propagate updates in the AT and/or KMB, which will take place as follows:

If the device receiving the "I'm here" message is already part of the cluster, it must check whether the hashes of the AT and KMB match the ones it is currently using. If any of them are newer, it must update its information by downloading the AT and/or KMB from the device that sent the "I'm here" message. Of course, it should also re-encrypt all the title keys of the files that are stored in this device to the new value of K_b. If on the other hand, the AT or KMB of the device sending the "I'm here" is older than the ones it has, the roles must be

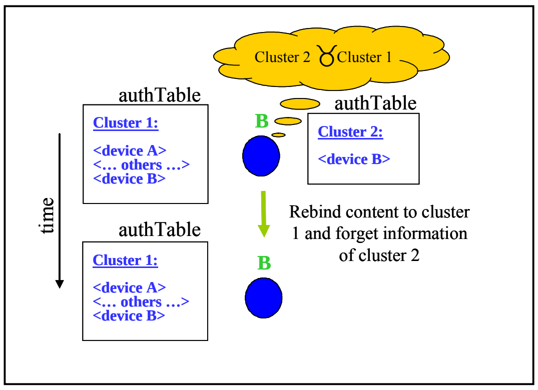

Figure 5: Merging clusters

[4] AES_G represents an AES-based cryptographic one-way function.

reverted: the device with the newer information must send an "I'm here" to the sending device to let it know there is newer information available.

Finally, if after processing an "I'm here" message a device finds itself belonging to more than one cluster it must try to merge them into a single one. This is done by checking if the AT of one of the clusters is a proper subset of the AT of the other cluster. When that happens, the device must try to abandon the smaller cluster and later move all the content it has stored for the smaller cluster into the bigger one.

As devices interact and share content within the user's network, information about the cluster is propagated, resulting in a distributed though uniform view of the user's cluster that allow all devices to share the same content. Figure 5 above depicts this merge procedure in the most common and simple context of a new device B with a singleton cluster 2 that has been introduced into an existing cluster 1.

4. Content Dissemination

The main target of CPWCT is to share content securely. Once a content file is ingested and bound to an xCP cluster, it will be shared within the whole cluster according to commitment of CPWCT agents.

4.1 Distributed Content Management

Each CPWCT agent maintains knowledge about local xCP content $B_s = \{cf_1, \ldots cf_m\}$, where cf_m is an xCP content file that could be identified uniquely in an xCP cluster by a triple *(id, clusterinfo, agent)*. *id* is the identifier of an xCP content file and contains a general description of the content file including attributes of the media format. *clusteinfo* is a set of information about the cluster to which the xCP content file is bound to, and will be used by CPWCT agent to decrypt the content. *agent* represents the location where the xCP content file is located. CPWCT supports both P2P file sharing and P2P streaming. There may exist more than one copy of an xCP content file within a xCP cluster. The P2P streaming mechanism in CPWCT will be described in section 4.2.

Based on B_s, a CPWCT agent performs actions triggered by events $e \in \{E_u, E_a\}$, where E_u is the event set generated by user involvement and E_a is the event set generated during communication with other CPWCT agents. As described in chapter 3, since each content file is tightly bound to a cluster, a particular action performed by CPWCT agents is content rebinding. Once the cluster information is changed (e.g. new device joining or new KMB introduced), each CPWCT agent will decrypt the encrypted title key (*eKt*) located in header of each content file using the old binding key (K_b), and encrypted it using the new binding key (K'_b).

$$rebind : e_{cl} \rightarrow encrypt\ (decrypt\ (eKt_i, K_b), K_b')$$

4.2 Secure P2P Streaming

Because of the asymmetry in device capability, low-end devices such as PDAs and smart phones don't have enough storage for large content files and need streaming functions to render content that resides on other high-end devices such as PCs and home media servers. Even for more powerful devices, streaming can enhance the efficiency of rending by removing the time consuming replication phase, and decreasing the complexity of content management with fewer redundant copies of each content file on different devices in the same cluster.

Different from P2P file sharing, P2P streaming in CPWCT poses more stringent requirements: (1) a secure mechanism to protect the media from being accessed illegally during transmission; (2) independence from specific media players, media formats and network protocols for diversity of devices and media technologies used in heterogeneous networks and; (3) symmetric functions applicable to both low-end and high-end devices according to P2P nature.

Conventional streaming solutions based on client/server architecture and open standards such as Internet Media Streaming Alliance (ISMA) are not designed for this purpose. DTCP provides device authentication and data encryption for devices connected with a digital interface in a home network. Nevertheless, it is based on PKI that is different from broadcast encryption mechanism implemented by CPWCT [2]. In CPWCT, an xCP-based lightweight secure streaming mechanism is designed to avoid conflicts between streaming and content protection in the P2P environment, especially when a low-end device such as a PDA plays the role of a streaming server, with balance made between applicability and functionality.

The structure of the CPWCT streaming mechanism and the streaming process from agent a_1 to a_2 are shown in figure 6. The following components are involved:

Trusted Player – The player plugged in the CPWCT agent for rendering content, which should support HTTP streaming that is widely supported and has no complex control mechanism

Content Binding Service – The service provided by xCP Cluster Protocol Stack for the encryption and decryption of content files.

Controller – It handles user requests and coordinates the streaming process. Once an event $e_{md}(a_1, c_i)$ is received to request a_1 to render content c_i on a_2 , the controller starts the trusted player $url_p(c_i)$ with pseudo media URL $url_p(c_i)$ pointing

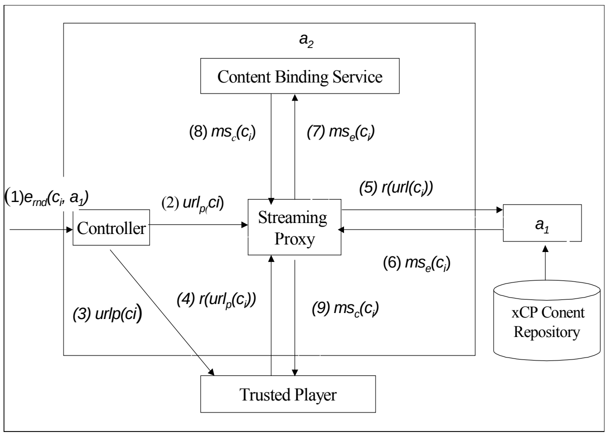

Figure 6. Process of xCP based secure P2P streaming

to a local Streaming Proxy, and passes <$url_p(c_i)$, $url(c_i)$>, the mapping set between $url_p(c_i)$ and the real media URL $url(c_i)$ to the Streaming Proxy.

Streaming Proxy – Core component that acts both as a client to the requesting agent and as a streaming server to the local Trusted Player. When the Streaming Proxy receives a streaming request $r(url_p(c_i))$ from the local Trusted Player, it checks the map <$url_p(c_i)$, $url(c_i)$> and sends the real request $r(url_p(c_i))$ to a_2. The encrypted media stream $ms_e(c_i)$ responded from a_2 will first be decrypted by invoking the Content Binding Service of xCP Cluster Protocol stack with the cluster information described in chapter 3, and then the clear media stream $ms_c(c_i)$ will be passed to the local Trusted Player. The operations from (5) to (8) are opaque to the Trusted Player.

Comparing this with a conventional streaming solution, the CPWCT secure streaming mechanism has prominent advantages. Unlike conventional secure streaming servers that dynamically encrypt content during the streaming process, the CPWCT agent decouples the processes of encryption and streaming, using the already encrypted content file generated during ingestion. Also, it moves most of the workload of streaming to requesting peers, and the requested peer becomes only responsible for sending the encrypted media stream, so that low-end devices can support concurrent streaming transactions. As a result, it provides a symmetric function by which even low-end devices such as PDAs can become streaming servers. On the other hand, transmission of the media stream is independent of the network protocol and security mechanism, so even proprietary protocols without security mechanisms can be used to transport content streams. Meanwhile, a unified user experience for the streaming of both local and remote content files can be implemented using the same mechanism.

5. Dynamic P2P Indexing

In P2P home networks, shareable content changes dynamically. Nevertheless, all devices in the P2P network should be aware these changes in real-time. Popular internet P2P content sharing systems usually involve a centralized directories or registry services which use technologies derived from the enterprise. Dedicated multimedia applications for the home network, e.g., Network-Integrated Multimedia Middleware [6], [12], also work in such mode. Nevertheless, the failure of a device with centralized content information may cause a critical problem in the home network.

Also, in most of these solutions it is difficult to accommodate heterogeneous structures because of the lack of device capability information. In such case, devices may get the information about all the content shared within the cluster, while only part of the content can be rendered. Then, an error occurs when this device tries to render this content that is beyond its capabilities, e.g., when a PDA tries copy a huge movie file to local storage, or a device with only mp3 support tries to render an MPEG4 file. Though standards for mobile web access such as W3C CC/PP and OMA UAProf use capability information, their main focus is on client/server architecture and provides no mechanism to dynamically notify clients about changes in the contents of the server.

In CPWCT, a Crawler which shares content information dynamically among CPWCT agents according to device capabilities described in W3C RDF format [10] was developed. Each CPWCT agent is responsible for maintaining local content and capability information. Each response containing content information from a requested peer has been filtered using the capabilities information of the requesting peer to avoid replicating and rendering useless content. This significantly decreases user involvement and makes the system more robust. The changes in content information are broadcasted actively. The update of content information is triggered by changes in the content, cluster topology, network connectivity, capabilities and status of each CPWCT agent. Fault tolerance is implemented for system level changes of device and cluster are monitored.

5.1 RDF Based Capability Description

One important characteristic of RDF metadata is the ability to use distributed annotations for the same resource. Furthermore, RDF schemas are flexible and extensible such that schemas can evolve over time, and RDF allows the easy extension of schemas with additional properties. As such RDF is capable of overcoming the problems of fixed and unchangeable metadata schemas that often occur in current P2P systems. Annotations about resources are based on various schemas that are defined using RDF and are transmitted in RDF based messages.

Using RDF, we can represent schemas based on device, capability and value, to define the vocabulary used for describing device capability. An RDF triple (*d, c, v*) represents specific annotations, where *d* identifies the device we want to abstract, *c* specifies what property the device has, and *v* specifies the value of this property. Figure 7 shows a sample of system capability used in CPWCT, which uses RDF.

```
<rdf:RDF >
  <rdf:Description about="http://www.ibm.com/DeviceX">
    <s:Assignment>
      <rdf:Description about="http:// www.ibm.com/DID">
        <v: FreeStorage >32M </v: FreeStorage >
        <v: CPUSpeed >1.5G Hz </v: CPUSpeed >
      </rdf:Description>
    </s: Assignment >
  </rdf:Description>
</rdf:RDF>
```

Figure 7. Sample of RDF based capability description

5.2 Capabilities-Aware P2P Indexing

In an xCP cluster, the global content snapshot is $B_{cs}= \cup B_{s,i}$, where $a_i \in A$ is the agent in the cluster, and $B_{s,i} =\{cf_1, \ldots cf_m\}$ is the set of content files managed by a_i. Then, the global content snapshot retrieved by a_i, $B_{cs,i}= \cup f(B_{s,i}, C_i)$, where $a_i \in A$ and $j \neq i$, is only a subset of B_{cs}. This result has been filtered using the capabilities of a_i. The basic assumption is that $B_{s,i}=\varphi$ if a_i is disconnected from the network, a_i is shutdown or a_i has no local content file.

One of the difficulties in an xCP cluster is that no agent has access to the global state of the system. In CPWCT, the content information can be updated in active and passive modes in various scenarios. A simple message set $M=\{m_{req}, m_{res}, m_{ref}\}$ is designed to simplify the distributed transaction among CPWCT agents. Here, m_{req} is the message sent to request shared content information from another agent, which contains the media capability, system capability and even user profile of requesting peer in RDF format; m_{res} is the response message which contains local content information filtered according to the capabilities of requesting device and; m_{ref} is the notification message which is sent to notify other CPWCT agents of local contents change.

In the CPWCT Crawler, an event set represented by $E = \{e_{mc}, e_{sc}, e_{lc}\ e_{as}, e_{ras}, e_{cl}, e_{req}, e_{res}, e_{ref}\}$, is defined to trigger actions for content indexing. e_{mc} and e_{sc} are the events generated when the local media capabilities change (e.g., a media player is added or removed) and the system capabilities change (e.g., memory, hard disk or network bandwidth is changed); a CPWCT agent generates e_{as} when its status changes, typically, when the agent is started; e_{lc} is used to notify about changes on local content, (e.g., content file is ingested or removed); e_{cl} is an event generated when the cluster topology is changed (e.g., an agent joins or leaves the cluster); e_{ras} is event generated when other agent is disconnected from the local agent; e_{res}, e_{req} and e_{ref} are generated when m_{res}, m_{req} and m_{ref} messages are received. Following are the actions triggered by the events.

$$requestall \quad : \left\{ \begin{matrix} e_{cl} = \text{CONN} \\ e_{as} = \text{START} \end{matrix} \right\} \rightarrow send\ (m_{req}, A)$$

$$request \quad : \left\{ \begin{matrix} e_{cl}(a_i) = \text{JOIN} \\ e_{ras}(a_i) = \text{START} \\ e_{ref}(a_i) \end{matrix} \right\} \rightarrow send\ (m_{req}, a_i)$$

$$notifiyall \quad : \left\{ \begin{matrix} e_{lc} \\ e_{mc} \\ e_{sc} \end{matrix} \right\} \rightarrow send\ (m_{req}, a_i)$$

$$response \quad : e_{req} \rightarrow send\ (m_{req}, a_i)$$

6. Conclusion

A novel approach for content management and protection in P2P home networks was introduced. A MAS model for secure clustering, content dissemination and indexing was proposed. In such a system, devices in a home network are organized in a trusted P2P cluster leveraging the broadcast encryption based xCP Cluster Protocol. Based on this torus, the distributed content dissemination mechanism for purely P2P home network including lightweight secure streaming mechanism and RDF based capability-aware content indexing mechanism were implemented. The resulting architecture poses significant enhancements in the ability to dynamically self-reorganize the underlying topology and deploy context sensitive functionality, while pervasively protecting the content from illegal redistribution. As a result, the entire P2P network of devices is turned into a secure virtual multimedia device from the user's perspective.

Acknowledgements

We would specially like to thank Amal Shaheen for conducting CPWCT project and providing valuable comments on earlier drafts of this paper. We would also want to warmly thank Florian Pestoni, Stefan Nusser and Jeff Lotspiech for their initial and continuing support, helping us learn about the exciting fundamentals of this technology.

Reference

[1] Jeffrey Lotspiech, Stefan Nusser, Florian Pestoni. "Anonymous Trust: Digital Rights Management Using Broadcast Encryption" PROCEEDINGS OF THE IEEE, VOL. 92, NO. 6, JUNE 2004

[2] A.S. Rao, M.P.Georgeff. "An abstract architecture for rational agents". Proceedings of Knowledge Respection and Reasoning, pages 439-449, 1992.

[3] Florian Pestoni, Jeffrey B. Lotspiech, Stefan Nusser. "xCP: Peer-to-Peer Content Protection". IEEE SIGNAL PROCESSING MAGAZINE, MARCH 2004.

[4] Jeffrey B. Lotspiech, Dalit Naor, Florian Pestoni. "Secure Local Agreement in a Peer-to-Peer network"

[5] Jeffrey B. Lotspiech, Stefan Nusser, Florian Pestoni. "Broadcast Encryption's Bright Future". Computer magazine (Vol. 35, No. 8), August 2002

[6] Marco Lohse, Philipp Slusallek. "Middleware Support for Seamless Multimedia Home Entertainment for Mobile Users and Heterogeneous Environments".Saarland University, 2003

[7] A. Fiat and M. Naor. "Broadcast Encryption. Advances in Cryptology". Crypto '93, Lecture Notes in Computer Science 773 (1994), 480-491.

[8] D. Naor, M. Naor, and J. Lotspiech. "Revocation and Tracing Schemes for Stateless Receivers". Advances in Cryptology (Crypto 2001), Lecture Notes in Computer Science 2139, Springer-Verlag, New York, 2001, pp. 41-62.

[9] 4C Entity. LLC. http://www.4centity.com

[10] D.M. Wallner, E.J. Harder, and R.C. Agee. "Key Management for Multicast: Issues and Architectures," RFC 2627 (informational), July 1999; ftp://ftp.isi.edu/in-notes/rfc2627.txt

[11] C.K.Wong, M.Gouda, and S.Lam. "Secure Group Communications Using Key Graphs". Proceedings SIG-COMM 1998, ACM Press, New York, pp. 68-79

[12] Patrick Becker, Patrick Cernko, Wolfgang Enderlein, Marc Klein, Markus Sand."The Multimedia-Box - Design and Development of a Multimedia Home Entertainment System for Linux".Advanced practical project, Universität des Saarlandes, 2002

[13] GartnerConsulting. "The Emergence of Distributed Content Management and Peer-to-Peer Content Networks". GartnerConsulting, January 2001.

[14] W3C. "Resource Description Framework (RDF) Model and Syntax Specifation". RFC-rdf-syntax-19990222, February 1999.