A Multi-Tenant RBAC Model For Collaborative Cloud Services

Birru Devender[1], Dr.Ch.V. Raghavendran[2], L. Manikanta Rahul[3]

[1]Associate professor,[2]Professor, [3]MTech Student, Department of CSE From Holy Mary Institute of Technology & Science Bogaram, Telangana.

Abstract

Most cloud services are built with multi-tenancy which enables data and configuration segregation upon shared infrastructures. In this setting, a tenant temporarily uses a piece of virtually dedicated software, platform, or infrastructure. To fully benefit from the cloud, tenants are seeking to build controlled and secure collaboration with each other. In this paper, we propose a Multi-Tenant Role-Based Access Control (MT-RBAC) model family which aims to provide fine-grained authorization in collaborative cloud environments by building trust relations among tenants. With an established trust relation in MT-RBAC, the trustee can precisely authorize cross-tenant accesses to the truster's resources consistent with constraints over the trust relation and other components designated by the truster. The users in the trustee may restrictively inherit permissions from the truster so that multi-tenant collaboration is securely enabled. Using SUN's XACML library, we prototype MT-RBAC models on a novel Authorization as a Service (AaaS) platform with the Joyent commercial cloud system. The performance and scalability metrics are evaluated with respect to an open source cloud storage system. The results show that our prototype incurs only 0.016 second authorization delay for end users on average and is scalable in cloud environments.

Keywords: cloud computing; multi-tenancy; trust; collaboration; fine-grained authorization

1. Introduction

The growing predominance of cloud computing impacts every aspect of the information technology (IT) industry [1]. It brings business agility and lower costs for information systems by using virtualization and shared infrastructures. Most cloud providers isolate tenants from each other using multi-tenancy to secure user data and configurations. However, the isolation strategy hampers multi-tenant collaborations which are also essential in the cloud [2]. In particular, fine-grained secure resource sharing among tenants is not fostered in today's commercial clouds.

Utilizing existing access control models, cloud providers are capable to control user activities within a single tenant. For example, NASA integrates Role-Based Access Control (RBAC) in the cloud to enforce fine-grained access control in their existing directories. Nevertheless, the traditional RBAC model [3] is not designed to enforce collaborative access control in decentralized environments. Currently, database schema is widely utilized to enable multi-tenant data sharing for Software as a Service (SaaS). However, this approach is confined to address multi-tenant access control in databases and cannot be directly extended to protect other vital

types of resources such as files and virtual machines. Consequently, in order to enable secure multi-tenant collaborations in the cloud, we need a general fine-grained access control model for this purpose.

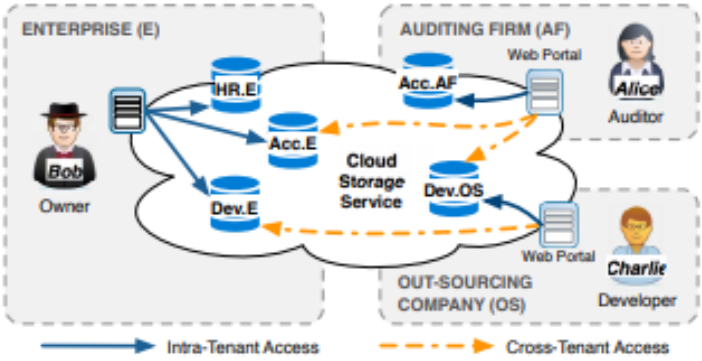


Fig. 1: Multi-tenant accesses in an out-sourcing case

To motivate the problem, we use an out-sourcing example illustrated in Figure 1, in which the Enterprise (E), the Outsourcing company (OS), and the Auditing Firm (AF) are three collaborating parties sharing a common cloud storage service. E out-sources part of its application development work to OS and external auditing tasks to AF. The cloud storage service provides storage services for the development department of OS, the accounting department of AF, and three of E's departments, development, accounting, and HR, as segregated tenants. Let "." denote the affiliation relation between the tenant and its organization (also called issuer) so that, for example, Dev.E represents the tenant corresponding to E's development department. The example cross-tenant accesses for collaborations are as follows.

C1. Charlie as a developer in OS has to access the source code stored in Dev.E to perform his out-sourcing job;

C2. Alice as an auditor in AF requires read-only access to financial reports stored in Acc.E; and

C3. Alice needs read-only accesses to Dev.E and Dev.OS in order to audit the out-sourcing project.

For simplicity, in our examples we assume all the tenants are created on a single cloud service, bringing homogeneous architecture which is often the case in cloud environments [4]. However, we do not exclude the potential of heterogeneous collaborations among multiple cloud services or even among multiple service models: SaaS, Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [5]. As a common collaboration need suggests, a user may test and deploy the source code stored in Dev.E directly on another tenant of a PaaS service. This function requires secure collaborative accesses between the two services. In fact, SaaS services are often hosted on PaaS clouds in which the SaaS services are regarded as tenants. The similar situation holds between PaaS and IaaS. Thus, we treat accesses among multiple tenants, clouds, or even service

models equivalently in the abstraction level as multi-tenant accesses upon which access control mechanisms should be enforced.

In order to achieve multi-tenant access control, Calero et al propose a multi-tenancy authorization system (MTAS) by extending RBAC with a coarse-grained trust relation. Multitenant collaborations are enabled in MTAS by bridging two tenants with a cross-tenant trust relation. The tenant establishing the trust is called the truster and the tenant being trusted is called the trustee. The essence of collaboration is resource sharing wherein a resource requester requests to share resources from a resource owner. The resources here represent the objects, such as virtual machine instances or stored files, in a tenant leased from the cloud service provider (CSP). For instance, in order to enable cross-tenant access C1 in the out-sourcing example above, an MTAS trust relation is established with Dev.OS as truster and Dev.E as trustee, allowing Dev.E as resource owner to issue suitable cross-tenant assignments to allow Dev.OS's users to access Dev.E's resources. By definition, the MTAS trust relation is always established by the resource requester. The granularity of MTAS trust is refined in [6] to reduce undue exposure of the truster's sensitive authorization information to the trustees. While Calero et al provide use cases for this resource-requester initiated cross-tenant trust relation, there are obviously other practically useful approaches that can be developed.

In this paper we develop a family of models for a cross tenant trust relation that is established by the resource owner. We demonstrate the utility of this approach by means of use cases. Later in the paper we formally compare the role-based trust models of MTAS, the current paper and RT [7]. There may be possibility of additional cross-tenant trust models, based on roles and perhaps on attributes. Eventually we believe there will be consolidation and unification of cross-tenant trust models but we are currently at early stages of developing and investigating alternate cross-tenant trust models.

Our central contribution is a novel family of Multi-Tenant RBAC (MT-RBAC) models where the cross-tenant trust relation is established by the resource owner rather than by the resource requester. MT-RBAC extends the traditional RBAC model [8] with two new built-in entity components: issuers and tenants.1 Each issuer can have multiple tenants in the cloud, whereas each tenant belongs to a single issuer. A cross-tenant trust relation is established and maintained by the issuer of the resource owner tenant (i.e., truster), as opposed to the resource requester tenant (i.e., trustee).

Three MT-RBAC models integrate three different trust relations with increasingly finer-grained constraints, respectively tenant trust (MT-RBAC0), trustee independent public roles (MT-RBAC1), and trustee dependent public roles (MTRBAC2). To allow the trustee to access the thruster's resources, the base model MT-RBAC0 requires the trustier to expose its entire role set and corresponding authorization assignments to the trustee. For example, to achieve the cross-tenant access C1 in the out-sourcing example, E will assign a trust relation from Dev.E to Dev.OS so that OS can use E's authorization information to issue appropriate cross-tenant

assignments enabling Dev.OS's users to access Dev.E's resources. To limit unnecessary exposure of the thruster's authorization information, MT-RBAC1 requires only exposing the thruster's public roles to all the trustees, these being the same for all trustees. Further, MT-RBAC2 enforces finer-grained constraints by exposing the thruster's public roles on a trustee by trustee basis. With these trust relations, a thruster's issuer can flexibly and efficiently constrain accesses from the corresponding trustees in a suitably fine-grained manner.

We demonstrate the feasibility of MT-RBAC by prototyping it with SUN's XACML library. We develop and deploy a novel Authorization as a Service (AaaS) platform applying MT-RBAC models in a Joyent cloud. We systematically evaluate the performance of MT-RBAC with an open source cloud storage service. The experimental results show the MT-RBAC policies with different expressive power in the AaaS service incur various policy evaluation delays. Evaluation of MT-RBAC policies takes on average no more than 0.016 second overhead in downloading files. Therefore, we believe that the performance of the prototype system is acceptable for cloud services. Further, we observe that the prototype system is also scalable in cloud settings.

2. Background Work

In the creator clarifies Cross Tenant Trust Models bolstered and upheld by the cloud specialist coop. Considering the On-request Self-Service include characteristic for distributed computing. [9] Creator propose a formal cross occupant trust display (CTTM) and its part based augmentation (RB-CTTM) coordinating different kinds of trust relations into cross-inhabitant get to control models which can be upheld by the multi-occupant approval as an administration (MTAaaS) stage in the cloud.

In the creator examines Control Cloud Data Access Privilege and Anonymity With Fully Anonymous Attribute-Based Encryption which introduces a semi-unknown benefit control conspireAnonyControl to address the information protection as well as the client personality security in existing access control plans. AnonyControl decentralizes the focal expert to confine the character spillage and accordingly accomplishes semi-obscurity. In addition, it likewise sums up the record get to control to the benefit control, by which benefits of all tasks on the cloud information can be overseen in a fine-grained way. Hence, creator introduces the AnonyControl which completely keeps the character spillage and accomplish the full namelessness. Security examination demonstrates that both AnonyControl and AnonyControl-F are secure under the DBDH supposition, and execution assessment displays the plausibility of plans.

In the creator proposes Fine-Grained Two-Factor Access Control for Web-Based Cloud Computing Services proposed 2FA access control framework, a property based access control system is actualized with the need of both a client mystery key and a lightweight security gadget. As a client can't get to the framework in the event that they don't hold both, the component can upgrade the security of the framework, particularly in those situations where numerous clients share a similar PC for electronic cloud administrations. Furthermore, property based control in

the framework likewise empowers the cloud server to limit the entrance to those clients with a similar arrangement of characteristics while saving client protection, i.e., the cloud server just realizes that the client satisfies the required predicate, yet has no clue on the correct personality of the client. At long last, creator additionally does a reproduction to exhibit the practicability of proposed 2FA framework [10].

In the creator talks about the Jobber: Automating between inhabitant trust in the cloud that present Jobber: a very self-sufficient multi-occupant arrange security system intended to deal with both the dynamic idea of cloud datacenters and the craving for enhanced between occupant correspondence. Middleman model use principals from Software Defined Networking and Introduction Based Routing to fabricate a between occupant organize strategy arrangement prepared to do naturally permitting improved correspondence between confided in inhabitants while additionally blocking or rerouting movement from unfrosted inhabitants. Middleman is prepared to do naturally reacting to the continuous changes in virtualized server farm topologies and, not at all like conventional security arrangements, requires insignificant manual setup, eliminating design mistakes.

In creator proposes Toward Fine-grained Data-level Access Control Model for Multi-occupant Applications, where part based and information based access control are both bolstered. Lightweight articulations are proposed to introduce confused approach runs in arrangement. In addition creator likewise talk about the design and approval technique which executes these two models. Some specialized execution points of interest together with the execution result from the model are given.

In the creator proposes Data Security for Cloud Environment with Semi-Trusted Third Party (DaSCE) that clarifies the information security framework that gives (a) key administration (b) get to control, and (c) document guaranteed erasure. The DaSCE uses Shamir's (k, n) edge plan to deal with the keys, where k out of n shares are required to create the key. The creator utilize numerous key directors, each facilitating one offer of key. Numerous key supervisors maintain a strategic distance from single purpose of disappointment for the cryptographic keys. (an) actualize a working model of DaSCE and assess its execution in light of the time expended amid different tasks, (b) formally display and break down the working of DaSCE utilizing High Level Petri nets (HLPN), and (c) check the working of DaSCE utilizing Satisfiability Modulo Theories Library (SMTLib) and Z3 solver. The outcomes uncover that DaSCE can be viably utilized for security of outsourced information by utilizing key administration, get to control, and record guaranteed cancellation.

3. Problem Statement

The main objective of this research work is achieving access control and efficient revocation in multi-tenancy cloud storage. For this proposing two different access models one is R-RBAC

model and RW-Access control [11].TSP using R-RBAC (Revocable-Role based access control) model can allocate roles to different tenants and whenever required he can revoke also.Tenant can enable security for his data using RW (Read Write)- Access control.

- Using single tenant resource utilization is less when compared to multi-tenant.
- Using single tenant more expensive.
- Difficult to define access control over multi-tenant
- Revocation of particular tenant is difficult process

Multi tenant is a shared storage server paradigm where multiple tenants are sharing single storage server in order to avoid cost and it avoid local storage maintenance, in multi tenancy achieving high scalability and effective access control is defined. In this implementation Tenant service provider (TSP), Tenant and Cloud service provider (CSP) are involved. From CSP storage server can accessed by TSP after TSP will share resource among multiple tenants.

In cloud condition multi-inhabitant stockpiling server is gotten to by different clients called occupants, so multi-inclination enhance asset sharing and it lessens cost. [12] In any case, giving security between multi-inhabitants is real test so in this work to conquer challenges in multi-inclination proposing two levels of security. First level security for TSP, utilizing R-RBAC the TSP can give set of benefits to set of occupants over capacity server. At whatever point inhabitant asking for capacity in light of occupant signature the TSP will allot Specific Square, and he can likewise repudiate specific inhabitant and reassign stockpiling to another occupant. Second level security for Tenant, utilizing RW-Access control, an inhabitant can characterize set polices over his stockpiling like who can have perused get to control and compose get to control.

4. Experimental Results

To demonstrate the feasibility of our approach, we implement a prototype to achieve multi-tenant authorization for collaborative cloud services. We also evaluate the performance and scalability of our prototype with Cloud Storage, an open source cloud storage system, deployed as a cloud service.

1) Implementation Overview: In order to foster fine-grained access control in collaborative cloud environments, we propose Authorization as a Service (AaaS) as a novel service model providing an independent authorization infrastructure in a multi-tenancy manner. This service can be integrated with the existing cloud services to manage and process authorizations for them. AaaS supports multi-tenant access control by applying suitable access control models, such as MT-RBAC.

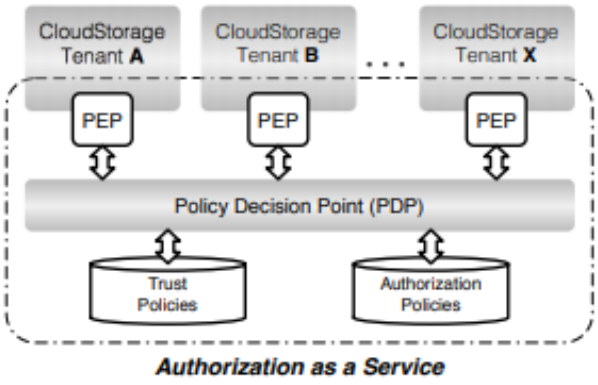


Fig. 2: The prototype architecture

Figure 2 shows the prototype architecture in which the tenants of the cloud storage service use a common authorization service (i.e., AaaS) by associating each with a policy enforcement point (PEP) module. The PEPs parse the requests from end users and generate normalized (XACML format) requests to the centralized policy decision point (PDP) module which refers to MT-RBAC policies stored in the centralized policy repository. The MT-RBAC policy specification is presented in Appendix A. After the decision is made, an XACML format response will be sent back to the requesting PEP to take effect with respect to the requested access. The integrity and authenticity of communication messages should be guaranteed, say via long-lived TLS connections between PEPs and the central PDP. For simplicity, they are not included in the prototype implementation. The prototype can be extended to a cloud authorization service with distributed PDPs.

The MT-RBAC policy engine is developed using SUN's XACML library [13]. The prototype is compiled and deployed on a Joyent cloud system. In the performance evaluation, the PDP module runs on a Smart Machine image with SmartOS Version 1.6.3 and 1 GB RAM. The PEP modules and the cloud storage service are deployed on another SmartMachine image with SmartOS Plus 64 Version 3.2.0 with 256 MB RAM. The CPU caps of both images are set to 350 meaning each can use at most 3.5 CPUs. The PDP and PEP modules are created on different physical machines, so that they don't affect each other in the performance evaluation results. All the machines in the prototype are connected through data center networks. End users connect to the cloud storage service through wireless local area network (WLAN) which is a common connection type for cloud service users. Note that, in the following experiments, the policy files are stored on the PDP server and an experiment request set consists of 10 independent sample multi-tenant access requests.

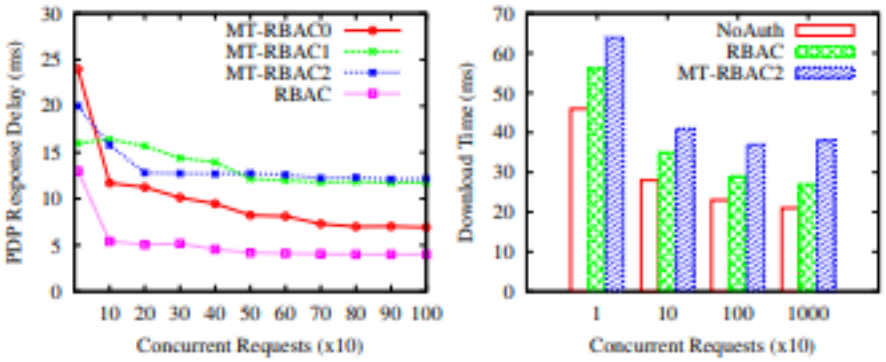
2) Authorization Delay: An MT-RBAC decision making process includes verifying subjects, resources and actions in the request, searching attributes and linking referred policy files. All these operations increase the overhead of access as authorization is involved. Authorization overhead is inevitable in MT-RBAC as well as other authorization models. Figure 3a compares

the policy evaluation delay in RBAC and the three MT-RBAC models. Note that in RBAC cross-tenant access requests are not supported. Due to the caching mechanisms of the operating system, as the number of concurrent requests increases, the average policy decision delay decreases dramatically until it reaches a stable state. RBAC has the least delay of 4.01 ms, while MT-RBAC0 has 6.92 ms delay. The evaluation of T P policies contributes to the extra delay of MT-RBAC0, compared to RBAC. Since IP S and DP S evaluations incur the similar I/O operations to T P evaluations, the authorization delay for MT-RBAC1 and MT-RBAC2 are similar. MT-RBAC1 and MT-RBAC2 have the most delay of 11.75 ms and 12.18 ms, respectively. MT-RBAC models introduce acceptable evaluation overheads compared to RBAC.

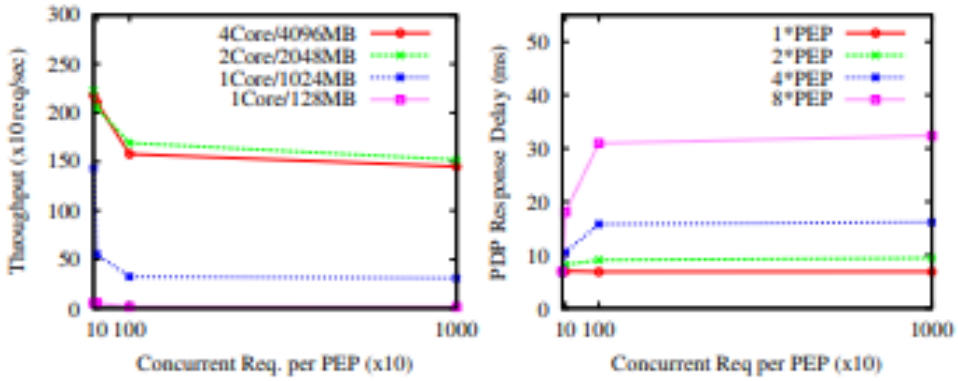
Figure 3b shows a comparison of delays at the client-end of the Cloud Storage service. The delays are observed upon a 1KB file downloading task with or without authorization through RBAC or MT-RBAC2. According to the experiment result, we observe that MT-RBAC2 introduces 15.50 ms (≈ 0.016 second) authorization delay on average which we believe is acceptable for file downloading tasks in cloud storage services.

3) Scalability: Scalability is also a critical criterion to judge the feasibility of a cloud application. Thus, we evaluate our prototype with various cloud images for the PDP module and numbers of engaged PEPs to see whether the system performance improvement is proportional to the increase of the hardware capacity which is represented by image size in the cloud.

We compare the throughput of PDPs with various capabilities in terms of image size. The result is shown in Figure 3c. The speedup of the system is in positive correlation with the increase of CPU cores and memory size. The throughput decreases when the number of concurrent requests increases, until it reaches a stable position. Compare the authorization throughput of the PDP with 1Core/128MB and the one with 1Core/1024MB. The approximately ten-time increase in memory size results in constantly ten-time enhancement in the throughput. When the physical resource assigned to the PDP increases to 2Core/2048MB, the throughput increases around five times. But the throughput has no obvious increase when the hardware becomes 4Core/4096MB because the amount of requests does not fulfill the utilization of the system with increased capacity. According to the results, we conclude that



(a) PDP performance (b) Client-end performance



(c) PDP scalability (d) PEP scalability

the system throughput is proportional to the hardware capacity of the PDP module. Figure 3d illustrates the authorization delay in a 1Core/1024MB PDP with different amounts of engaged PEPs. More PEPs means more concurrent requests generated and more connections which consume the capacity of the PDP. When the number of PEPs increases exponentially, the average evaluation delay also increases exponentially. Hence, if all the engaged PEPs send equal amount of requests, the load of PDP can be roughly determined by the number of PEPs. As PEP is assigned per tenant, the required capacity of the system is proportional to the amount of tenants. The results show that AaaS with MT-RBAC is scalable in the cloud environment.

5. Conclusion

In this paper, we propose a family of MT-RBAC models by extending the well-known and widely used RBAC model with the components of tenants and issuers to address multi-tenant authorization for collaborative cloud services. MT-RBAC aims to enable fine-grained cross-tenant resource access by building tenant-level granularity of trust relations. We prototype MTRBAC using SUN's XACML library to implement an Authorization as a Service (AaaS) in cloud. To demonstrate the viability of the prototype system, we evaluate its performance and scalability in a Joyent cloud. The results show that the AaaS platform with MT-RBAC incur acceptable overhead and is scalable for the cloud storage service.

Reference

1. "Gartner outlines five cloud computing trends that will affect cloud strategy through 2015," Gartner Press Release, 2012.

2. P. Mell and T. Grance, "The NIST definition of cloud computing," Special Publication 800-145, 2011.

3. J. Judkowitz, "Taking advantage of multi-tenancy to build collaborative clouds," http://communities.intel.com/community/datastack/cloud builder/blog/2011/04/29/taking-advantage-of-multi-tenancy-to-buildcollaborative-clouds, 2011.

4. D. Jermyn, "Health care not yet ready to share," https://secure.globeadvisor.com/servlet/ArticleNews/story/gam/20110610/SRCLOUDHEALTH0610ATL, 2011.

5. A. Kurmus, M. Gupta, R. Pletka, C. Cachin, and R. Haas, "A comparison of secure multi-tenancy architectures for filesystem storage clouds," in Middleware, 2011, pp. 471–490.

6. J. McKenty, "Nebula's implementation of role based access control (RBAC)," http://nebula.nasa.gov/blog/2010/06/03/nebulasimplementation-role-based-access-control-rbac/, 2010.

7. D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control," ACM Trans. Inf. Syst. Secur., vol. 4, no. 3, pp. 224–274, Aug. 2001.

8. F. Chong, G. Carraro, and R. Wolter, "Multi-tenant data architecture," http://msdn.microsoft.com/en-us/library/aa479086.aspx, 2006.

9. Prasadu Peddi (2016), Experimental Study on Cloud Resource Prediction and Allocation using Bat algorithm, ISSN: 2455-6300, volume 1, issue 2, pp: 88-94.

10. I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in GCE, 2008, pp. 1–10.

11. Prasadu Peddi (2016), Comparative study on cloud optimized resource and prediction using machine learning algorithm, ISSN: 2455-6300, volume 1, issue 3, pp: 88-94.

12. B. Tang, R. Sandhu, and Q. Li, "Multi-tenancy authorization models for collaborative cloud services," in IEEE International Conference on Collaboration Technologies and Systems, 2013.

13. N. Li, J. C. Mitchell, and W. H. Winsborough, "Design of a role-based trust-management framework," in IEEE Symp. on Sec. and Priv., 2002, pp. 114–130.