

Authentication and authorization in microservice oriented application design

https://doi.org/10.31713/MCIT.2024.035

Yurii Kovalov

Department of Computer Engineering Faculty of Radiophysics, Electronics and Computer Systems Taras Shevchenko National University of Kyiv

Kyiv, Ukraine,

yuk123@meta.ua

Abstract – In the past years microservice oriented design patterns obtained much popularity among the system architects and programmers. One of the obstacles in this design is authentication and authorization issues. This article presents one of the possible solutions for this problem. The essence of this methodology is a single point of authentication and many points of authorization spread across all the microservices.

Keywords – microservice; authorization; authentication; monolith; application design;

I. INTRODUCTION

Recently microservice oriented application design obtained much popularity among the software architectures and engineers. It introduces a lot of advantages compared to monolith application design but has drawbacks too. The main advantages are the possibility to break down business processes to parts that can be implemented by different teams and reducing the complexity of every part of application. One of the drawbacks is an increased complexity of authentication and authorization procedures.

The point is that with monolith design developers have to protect only the periphery of application while with microservices there are many peripheries to protect. Authorization in monolith applications is usually centralized, while with microservices design every microservice must authorize user's queries by itself.

Scientists tried different approaches in their studies. Two separate microservices for authentication and authorization were proposed, while investigating performance with SQL and No-SQL databases [1]. Another study investigated a common approach to access control authorization. Authorization rules were shared between different microservices [2]. In the next study an attribute based JSON access control policy language was proposed [3]. As concluded in the next study, there are not many studies in the subject and practical demonstrating studies much less implementation of authentication and authorization with microservice architecture [4].

II. APPLICATION DESIGN

A. Interaction inside application

In this study some methodological recommendations in the authentication and authorization when dealing with microservice oriented application design were outlined. These recommendations are based on practical implementation of microservice systems for finance and accountancy.

As usual business processes in the realm of accountancy and finance are implemented very well with monolith application design. The benefits we would possibly receive from microservice design are easier support and maintenance. So let's try to implement microservice design including authentication and authorization to take advantage of both approaches simultaneously.

The idea is to implement a microservice application in such a way that it should look like a monolith for customers but behave like a microservice for programmers. In order to fulfill this the proposed methodology will construct a microservice application with one single entrance point – let's call it ApiGateway (Fig.1). All microservices are interacting with each other and users within a private secure network by sending events through EventBroker. ApiGateway authenticates users and passes out events to and from microservice applications and uses SSL (Secure Socket Layer) to



communicate with end users.



Modeling, control and information technologies - 2024

It is crucial for systems to operate as quickly as possible inside the enclosed private network. One user's query may pass through multiple microservices before the answer is ready. In our methodology we don't recommend using encryption between microservices because it will inevitably slow down events flow. The private network is protected by firewalls and VPN (Virtual Private Network) gateways. The same mechanisms were used for monolith applications and they demonstrated effectiveness and robustness. Just like in monoliths we protect only the periphery and gates of microservice application.

B. Authorization

As already mentioned above one of the main advantages in microservice application design is the possibility of implementing microservice functions by different teams. Hence every microservice should be as independent from others in design as possible. The central part of any application is authorization. In this methodology every microservice is responsible to keep authorization information for every user in the system. Of course in this situation the default (no information about authorization) is denying access to resources.

The first authorization rule every microservice should implement is authorizing other authorization rules change. Let's look at the example with menu

ingoas nonoonen									
FreeBSD NySel ×									
ile Edit Yew Query Databa	se j	Server	Tools S	ionipting	Help				
			1.2						
Navigator	use	rdeta	evenices	teiled	uservali	dtokens claimslist 🛃	irrolist ×		
ICHEMAS 0-	6		19 9	Q ()	19510	C C C Limit to 100 row		4 Q N N	
R Filter objects			SELECT .	FROM	enutith	orization claimslist.			
efmigrationshistay cleinslist eventachelist groodedata userolata userolata									
Stored Procedures	<				_				
Views Stored Procedures Punctions eventbroker	< Re	sult Cri	e 🖪 📢	Filter Nov	•	tele 🚅 🖶 (😓 Exporti Trapo	te 🙀 🔯 Wrap Cell Contante 🖽	
Views Scored Procedures Punctions overStroker menualtarization	< Re	suit Cris	d 🔝 📢 InDeleted	Ner for Clarid	or Userid	tde 🚅 🖶 I	biportilingo UserGranter3d	t: 🙀 🔞 Wing Cel Content: 🔣 Clarificale	
Views Stored Procedures Practices eventbroker Remusetharization Tables	< Re	solt Gri Id	e i i ta Celetad	Filer Rov Clairedd	er Userid 1	tdt: 🚅 🖶 1 DyredlatsTre 209-12-31 00:00:00.000000	biportilingo UserGranter3d	t: 🙀 🔯 Wrap Cel Contant: 🔀 Cantrolate ("Clantrolate" (1,2,3,4,5))	
Views Stored Procedures Stored Procedures Consistent Constitution Cons	< Re	ault Gri	e i i tableted	Filter Rove	er Userid 1 1	tdie 🚅 🖶 1 ExpiredLateTime 2009-12-31 00:00:00.000000 2009-12-31 00:00:00.000000	biportilmpo UserGranterid 1 616161	et 🙀 🌆 Wees Call Content: 📧 Calentiale (Calentiale) (Calentiale) (1,2,3,4,4)) (Calentiale) trans)	
Views Stored Procedures Stored Procedures Processes eventbroker Tablesfrigrationshistoyclaimslat	<	ault Gri 1d 2 3	e i i ta Celetard D D D D	Pher Nov Clandd 1 4 2	uberîd 1 1 1	tde: 2009-12-31 00:00:00 000000 2009-12-31 00:00:00 0000000 2009-12-31 00:00:00 0000000 2009-12-31 00:00:00 000000	bport/Impo UserGranter3d 1 616161 616161	t: Wes Coll Content: IS ClamMake (ClamMake) (ClamMak) (12,2,3,43) (ClamMak) (14,4) (ClamMak) (14,4)	
Viena Stored Procedures Practices resultations resultations - cerestroker -	< Re	ault Gri 1d 1 2 3 4	e I II 🚯 📢 IzDeleted D D 0	Filer for Clarid 1 4 2 3	uertd Leertd 1 1 1	ToprediateTime 2009-12-31 00:00:00,000000 2009-12-31 00:00:00,000000 2009-12-31 00:00:00,000000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:00000 2009-12-30 00:000000000 2009-12-30 00:0000000000000 2009-12-30 00:000000000000000000000000000000000	b Deporti Treport User Granter Id 616161 616161 1	te See Twee Cell Content: E Carrindae (Carrindae (1,2,3,4,3)) (Carated): Tax) (Carated): Tax)	
Viena Viena Viena Viena Cond Procedures Punctices orestarcher researcher(acion Viena) -cefreignationshistay -cefreignationshistay Calamatist Viena Vi	<	sult Cri Id I 2 3 4 5 6	d i i i i i i i i i i i i i i i i i i i	Piter Nov Clairedd 1 4 2 3 2 1	er Userid 1 1 1 1 2 2	Edit gal lbb ExpiredDateTime 2009-12-31 00:00:00.000000 2009-12-31 00:00:00.000000 2009-12-31 00:00:00.000000 2009-12-30 00:00:00.000000 2009-12-30 00:00:00.000000 2009-12-30 00:00:00.000000 2005-07-31 00:00:00.000000 2005-07-31 00:00:00.000000 2005-07-31 00:00:00.000000 2005-07-31 00:00:00.000000 2005-07-31 00:00:00.000000	 DeportiTepor UserGranterId 616161 616161 1 1 1 	t () () () () () () () () () () () () ()	
Viena Stored Procedures Punctices contentions Contenti	< Re 	soft Cri 1d 2 3 4 5 6 7	d i i i i i i i i i i i i i i i i i i i	Piter Nov Clairedd 1 4 2 3 2 1 4	er Usertd 1 1 1 2 2 2	tale al 2000 2000 2000 2000 2000 2000 2000 2	bport/Impo UserGranter2d 616161 616161 1 1	t (2) (2) (2) (2) (2) (2) (2) (2) (2) (2)	
Viena Stored Procedures Practices exertification remultification Ching attorn for Ching attorn f	< Re 	sult Gri 1d 2 3 4 5 6 7 8	a III 🔹	Piter Row Claimid 1 4 2 3 2 1 4 3 3 2 1 4 3	e Userid 1 1 1 2 2 2 2	Extre all Extrema ExpresedData Time 2009-12-31 00:05:00.000000 2009-12-31 00:05:00.000000 2009-12-30 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.0000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000 2005-07-31 00:05:00.000000	b bpotilepo UserSceter3d 666361 616361 1 1 1	t Carendar Control (12,2,4,4) (Control (12,2,4,4) (Control (12,2,4,4) (Control (12,2,4,4) (Control (12,2,4,4)) (Control (12,2,	
Viena Stored Procedures Practices events roker Practices Processor Proc	<	soft Gri 1d 2 3 4 5 6 7 8 9	a III 🔹	Fiber Rove Claimid 1 4 2 3 2 1 4 3 2 2 1 3 2	tberid 1 1 1 2 2 2 2 3	tate al base a	b Doport/Impo UnerGranter3d 66666 1 1 1 1 1 1 1	Image Of Content E Control of (1,2,3,40) (Content 1, 10) (Content 1, 10) (Content 1, 10)	
Viena Stored Procedures Tradises eventurker eventurker menusuktarization * Configurationshistary * Configurationships * Configuratio	<	soft Gri 1d 2 3 4 5 6 7 8 9 10	e 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Filer flow Claim2d 1 4 2 3 2 1 4 3 2 5	tberid 1 1 1 2 2 2 2 3 1	Edm (2010) 2009-2010 2009 2000 2009 2000 2000 2000 2000	 Export/Import User/Franterid 616161 616161 1 	Constant (12,3,4,4) Constant (12,3,4,4) Constant (12,3,4,4) Constant (12,3,4,4) Constant (12,4,4) Constant	
Viena	<	soft Cri 11 2 3 4 5 6 7 8 9 10 11	al () IaDeletad D D D D D D D D D D D D D D D D D D D	Filter Rom Claim3d 4 2 3 2 1 4 3 2 5 5 5	e Usertd 1 1 1 2 2 2 2 3 1 2	Intel Intel Intel Diper-ED-411 (00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-12-31.00:00:00.000000 2000-02-31.00:00:00.000000 200-02-31.00:00:00.000000 200-02-31.00:00:00.000000 200-02-31.00:00:0000 200-02-31.00:00:0000 200-02-31.00:0000 200-02-31.00:0000 200-02-31.00:0000 200-02-31.00:0000 200-02-31.00:0000 200-02-31.00:0000 200-02-31.00:0000 <td> Export/Impo User/Franter3d 666661 666661 1 </td> <td>Constant activity Constant activity</td> <td></td>	 Export/Impo User/Franter3d 666661 666661 1 	Constant activity	
View Store Protockins Store Protockins Store Protockins Prutisas results and store results and store st	< Re	soft Get 1d 2 3 4 5 6 7 8 9 10 11 12	e 1 1 2000 D D D D D D D D D D D D D	Filter Rove Claimidd 4 2 3 2 1 4 3 2 5 5 5 5 2	1 Userid 1 1 2 2 2 2 2 3 1 2 5	Detr. (m) Detr. (m) Dependiant Tree 2000-12-31 (Dect0.30. Dect000) Dependiant Tree 2000-12-31 (Dect0.30. Dect000) Defendiant Tree 2000-12-30 (Dect0.30. Dect0.30. Dect	 Doport/Import User/Center3d G66561 G66661 1 <li1< li=""> 1 <li1< li=""></li1<></li1<>	Constantial	
Wires Store Protectives Store Protectives Practisat exteribitier non VBT Tables Image: Store Protectives Store Protectives Store Protectives Image: Store Protectives	< Re	sult Gri 1 2 3 4 5 6 7 8 9 10 11 12 13	d 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	Filer Rove Clandd 1 4 2 3 2 1 4 3 2 5 5 5 5 2 2 2	1 Userid 1 1 2 2 2 2 2 3 1 2 5 6	Intel Intel Intel DependbaseTime 2009-12-31 0000302 0000302 2009-12-31 0000302 0000302 0000302 2009-12-31 0000302 0000302 0000302 2009-12-31 0000302 0000302 0000302 2009-12-31 0000302 0000302 0000302 2009-12-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302 2029-07-31 0000302 0000302 0000302	Doport/Impo UmerGranter3d Si6161 616161 1	Contract (12.13.10) C	
Wren Store frontekres Store frontekre	< Re	sult Cri 1d 2 3 4 5 6 7 8 9 10 11 12 13 14	e 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Filer Row Clamb 4 2 3 2 1 4 3 2 5 5 5 5 2 2 2 5 5	e Userid 1 1 1 1 2 2 2 2 3 1 2 5 6 6	Detc. Bit DysecLiss Tre 2004-12-31 (Det0.50.0.00000) Detroit - 23.1 (Det0.50.0.00000) 2004-12-31 (Det0.50.0.00000) Detroit - 23.1 (Det0.50.0.00000) 2004-12-31 (Det0.50.0.00000) Detroit - 23.1 (Det0.50.0.00000) 2004-12-31 (Det0.50.0.00000) Detroit - 23.0 (Det0.50.0.00000) 2004-00.0.00000 Detroit - 23.0 (Det0.50.0.00000) 2004-00.0.00000 Detroit - 20.0 (Det0.50.00000) 2004-00.0.00000 Detroit - 20.0 (Det0.00000) 2004-00.0.00000	Deportingo UnerGranterid 66666 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Consent and Content E Consent A Consent A Cons	

microservice authorization rules (Fig. 2).

Figure 2. Authorization rules example.

The first authorization rule in the table sets access to users in giving authorization access for other people. Other rules are authorizing access to different parts of menu in microservice application. All rules can be managed by authorized persons using a micro frontend interface that accumulates authorization rules from all microservices (Fig.3).

vient autionzation s	ettings		
Can give access for c	others		
Can give access Access payslip w Access payslip st Access user clair Access balances	for others iorkers menu tudents menu ns index		
Expire date: 10/09/2	2024		
Access payslip worke	Expire date: 10/09/2024	告	
Access payslip stude	nts menu		

Figure 3. Authorization rules managing microservice.

It is essential to build this micro frontend interface with respect to possibility of changing it by every team independently, thus implementing microservice logic they would not intertwine with each other.

REFERENCES

- Randa Ahmad Al-Wadi and Adi A. Maaita (2023) "Authentication and Role-Based Microservice Architecture: A Generic Performance-Centric Design", Journal of Advances in Information Technology, Vol. 14, No. 4, November 2023, pp. 758-768 doi: 10.12720/jait.14.4.758-768.J. Clerk Maxwell, "A Treatise on Electricity and Magnetism," 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
- [2] Algimantas Venčkauskas, Donatas Kukta, Šarūnas Grigaliūnas and Rasa Bruzgiene (2023) "Enhancing Microservices Security with Token-Based Access Control Method", Sensor, 23, 3363, pp. 1-21, doi: 10.3390/s23063363.
- [3] Davy Preuveneers and Wouter Joosen (2017) "Access Control with Delegated Authorization Policy Evaluation for Data-Driven Microservice Workflows", Future Internet, 2017, 9, 58, doi: 10.3390/fi9040058.
- [4] Murilo Góes de Almeida and Edna Dias Canedo (2022) "Authentication and Authorization in Microservices Architecture: A Systematic Literature Review", Applied Sciences, 2022, 12, 3023, pp. 1-20, doi: 10.3390/app12063023