Cover Page

15th ICCRTS

"The Evolution of C2"

Title of Paper

A Lightweight C2 Service Invocation Method Based on HTTP Proxy

Topics

Topic 3: Information Sharing and Collaboration Processes and Behaviors
Topic 5: Experimentation and Analysis
Topic 9: C2 Architectures and Technologies

Authors

Heng Wang
Guangxia Zhou
Shuanghua Zhu
Feng Ding
Weitai Liang

Point of Contact

Heng Wang

Name of Organization

National Key Laboratory of Science and Technology on C^4ISR

Address

Nanjing, Jiangsu Province, 210014, P.R. China

Telephone

(86)0139-1298-1215

Email

puchengew@yahoo.com.cn

A Lightweight C2 Service Invocation Method Based on HTTP Proxy

Heng Wang, Guangxia Zhou, Shuanghua Zhu, Feng Ding, Weitai Liang

National Key Laboratory of Science and Technology on C4ISR, Nanjing, Jiangsu 210014, P.R.China

Abstract

SOA and Web Services technologies have been increasingly applied to network centric C2. In this paper, a service-oriented C2 software architecture is firstly proposed based on SOA ideas and Web services technology to provide framework and guidance to service integration in future service-oriented C2 systems, in which software and capability of C2 systems often are wrapped web services so as to implement C2 function and information exchanging through service invocation. Traditional WS invocation is implemented through SOAP via HTTP. Due to security and efficiency problems about SOAP/XML/HTTP, how to implement service invocation satisfying military requirement has become a hotspot problem concerning future C2.

Usual methods solving the above invocation problems often have higher complexity, more enormous load and longer develop cycle. Considering security requirement in military WAN environment, a lightweight service invocation method based on HTTP proxy is proposed in this paper. A HTTP proxy is deployed in border of LAN system, in which web services messages are intercepted and SOAP is transported through transport protocol accorded with military standard instead of HTTP in WAN. The experiments demonstrate that proposed method which can meet security and timeliness requirements in military environment is effective and feasible.

Keywords: C2 service invocation; HTTP proxy; Service-Oriented Architecture (SOA); Web Services; C2 software architecture; Simple Object Access Protocol (SOAP); Lightweight; Network centric C2.

1. Introduction

With the developments in networking and network centric, more and more applications need more flexible, agile and standard pattern to implement sharing and collaboration among them. From software level, Service-Oriented Architecture (SOA) [1] and Web Services (WS) [1,2] are such excellent technologies or architectures that help people to realize above application needs. SOA is an architectural approach and methodology that builds on the concept of services, and is an architectural style that represents business functionality as implementation-neutral, standards-based shared services. Web Services are a set of open standards that implement the architecture of SOA and the idea of service.

Characteristics of loose coupling, location transparency, reusability for SOA and characteristics of standardizing, opening for Web Services are consistent with software integration and sharing demands for future network centric command and control (C2) [4]. As a result, SOA and web services technologies have been increasingly applied to network centric C2, for example, Joint Command and Control (JC2) and Net-Enabled Command Capability (NECC). Moreover, U.S. DoD has taken SOA and WS as technological standard or criterion in several major programs, such

as Global Information Grid (GIG) [5], Net-Centric Enterprise Services (NCES) [6], and so on. Therefore, a service-oriented C2 software architecture is firstly proposed based on SOA ideas and Web services technology, which can be used to provide framework and guidance to service integration in future C2 systems.

In the above service-oriented C2 systems, i.e., on basis of the proposed C2 software architecture, software and capability of C2 systems often are wrapped web services (or other style services) so as to implement C2 function and information exchanging through service invocation. Traditional web services invocation is implemented through SOAP transported via HTTP. Due to some problems, such as unauthorized access, interpolated and intercepted caused by SOAP message, XML efficiency and security of HTTP and so on, these problems make web services invocation unfit for operational requirements in next generation C2, especially concerning timeliness and security. As a result, how to implement service invocation satisfying military requirement has become a hotspot problem concerning web services in future C2.

Usual methods include modifying or rewriting SOAP, or amending based on open-source web services productions. However, such methods have higher complexity, more enormous load and longer development cycle. Considering security requirement in military Wide Area Network (WAN) environment, a lightweight service invocation method based on HTTP proxy is proposed in this paper. A HTTP proxy is deployed in border of Local Area Network (LAN) system, in which web services messages are intercepted and SOAP is delivered through transport protocol accorded with military standard instead of HTTP in WAN. The invocation efficiency experiments demonstrate that proposed method is effective and feasible, and can meet security and timeliness requirements in military environment.

The rest of the paper is organized as follows. Section 2 gives the technologies that are related to this paper. A service-oriented C2 software architecture is proposed in Section 3. Section 4 describes our proposed service invocation method. Experiments are carried out in Section 5. Section 6 concludes the paper.

2. Related work

2.1 SOA and Web Services

Firstly, we consider the conception of service. By service-oriented idea, a service is a function that is well-defined, self-contained, and does not depend on the context or state of other services. SOA is an architectural approach and methodology that builds on the concept of services. Put simply, SOA spans both enterprise and application architecture domains. The benefit potential offered by SOA can only be truly realized when applied across multiple solution environments. This is where the investment in building reusable and interoperable services based on a vendor-neutral communications platform can fully be leveraged [1].

Web services (SOAP, WSDL, UDDI, and the extended Web services specifications) are a set of open standards that will lead to widespread adoption of SOAs and serve as the basis for a new generation of service oriented development. WS are also a set of operations, modular and independent applications that can be published,

discovered, and invoked by using industry standard protocols – Extensible Mark-up Language (XML), Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), and Universal Distribution Discovery and Interoperability (UDDI). It is a distributed computing model that represents the interaction between program and program, instead of the interaction between program and user. Web Services can also be defined as discrete Web-based applications that interact dynamically with other web services [3].

The architecture of WS is shown in Fig.1:

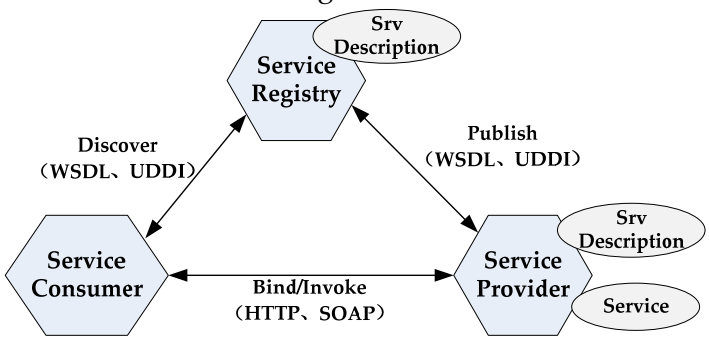

Fig.1 The WS architecture

SOA based on Web services has the following advantages:

1) It is standards-based, meaning that the organization no longer needs to invest in proprietary solutions, which create vendor lock-in.

2) It provides interoperability of solutions and allows you to mix and match best-of-breed products from several vendors, which can reduce costs significantly.

3) It supports intra-organization integration and can be extended to provide cross-organization and inter-organization integration.

2.2 Service Invocation for Web Services

In WS, service invocation refers to the whole process that web service requester invokes the service provider deployed in the service runtime environment through the web service client. Web service invocation can be used to support loosely coupled interaction between services, and it is one of the key elements for web services. The principle of service invocation for WS is shown in Fig.2.

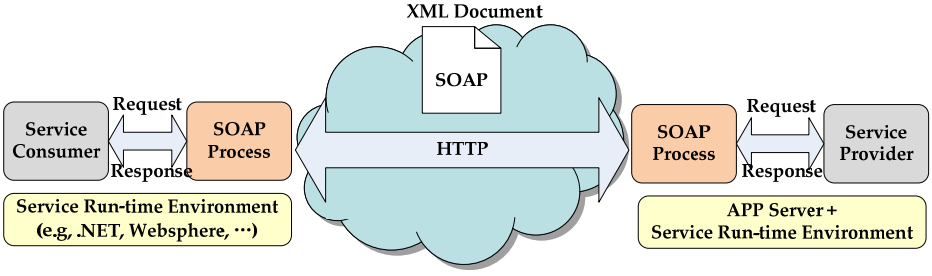

Fig. 2 The principle of service invocation for WS

Currently, commercial WS is often used to pass SOAP packets through HTTP protocol, and XML is data encoding method of SOAP. The SOAP defines a common format for XML messages over HTTP and other transports. SOAP is designed to be a

simple mechanism that can be extended to encompass additional features, functionalities, and technologies. SOAP consists of the three major blocks, or parts of SOAP messages: the envelope, the header, and the body. The envelope is required and marks the start and the end of the SOAP message. The header is optional, and can contain one or more header blocks carrying the attributes of the message or defining the qualities of service for the message. Headers are intended to carry contexts or any application-defined information associated with the message, such as security tokens, transaction identifiers, and message correlation mechanisms. The body is required and contains one or more body blocks comprising the message itself.

2.3 HTTP Proxy

In HTTP proxy, as an application-level gateway, the proxy server plays the role of the bridge between the client and the server. The clients or customers (such as IE, Netscape) send all of their requests to the proxy server, and the proxy server can listen to and receive the connection requests from clients. In the proxy server, a customer's identity information is verified firstly, then the customer's request data is received, and the desired IP address and port of the server, the requested documentation are parsed out. Next, the proxy server redirects to connect to the destination server through their own Socket, delivers the customer's information to the Web server, and returns the response coming from the Web server to the clients. HTTP proxy server works as shown in Fig.3:

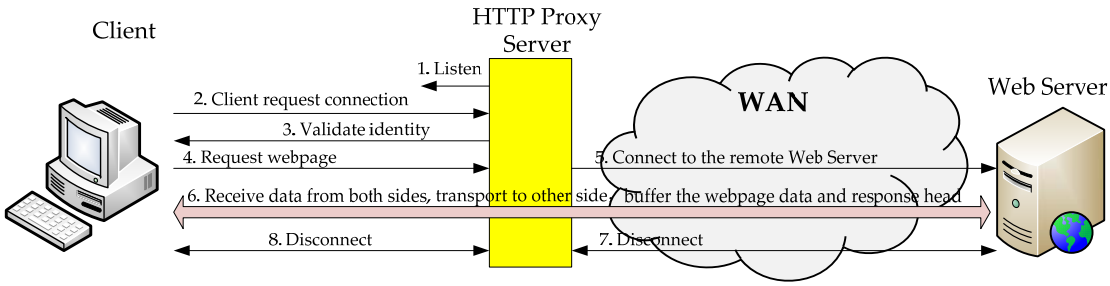

Fig. 3 HTTP Proxy Server

3. Service-oriented C2 software architecture

As we know, the increasing use of commercially supported open standards pushes the information technology (IT) infrastructure from proprietary military solutions towards SOA and Web Services [3]. As a result, the ideas and technologies SOA and web services have been increasingly applied to network centric C2 systems, such as JC2, NECC. In such service-oriented C2 systems, software and capability of C2 systems often are wrapped web services so as to achieve C2 function and information collaboration through service invocation.

To meet the above needs and further be supported to network centric warfare (NCW), combining with SOA ideas and Web services technology, we propose a service-oriented C2 software architecture, called SO-C2SA. In detail, we take

"service" as the granularity of the C2 software architecture, and use a unified description mechanism to achieve wrap of all kinds of C2 software services. Then, exploit the concept of "service" to summarize elements of domain software and the correlation between the elements, and achieve service combination and orchestration based on process approach. Taking the mission capability packages (MCP) [7] for the carrier of service capability, dynamically redistribute and reallocate operational resource capabilities, so as to improve largely mission-oriented flexibility in NCW. Service-oriented C2 system software architecture is shown in Figure 4:

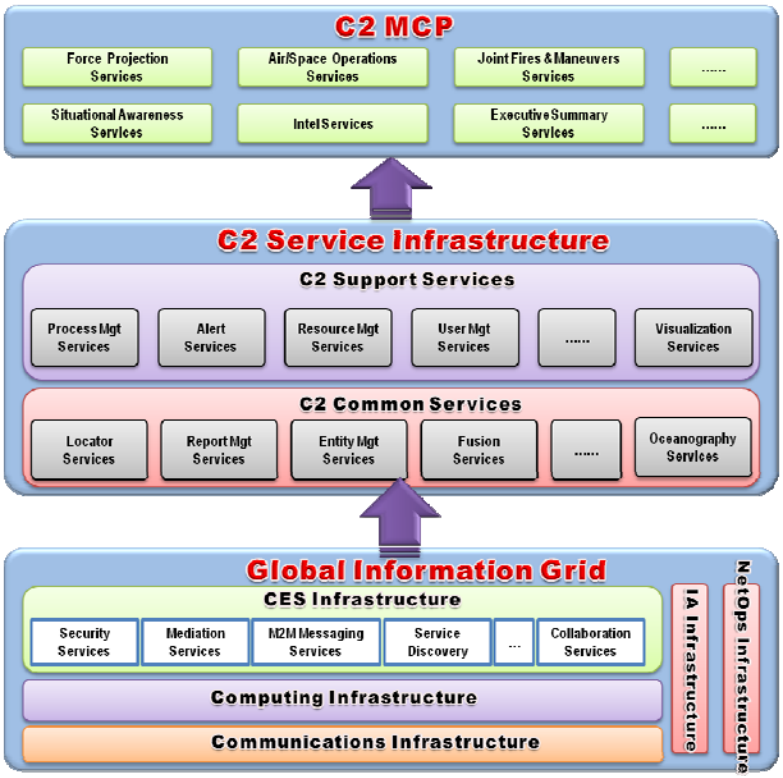

Fig. 4 Service-oriented C2 software architecture (SO-C2SA)

The framework of our proposed software architecture is divided into three levels from the bottom, the global information grid (GIG) [8] layer, C2 service infrastructure layer, as well as C2 MCP layer. GIG provides the underlying infrastructures, such as communication, computing, Core Enterprise Services (CES), information assurance (IA) and network operations (NetOps) infrastructure, and so on. The most important one is CES infrastructure which provides global, universal network centric service capabilities, such as security services, mediation services, M2M messaging services, service discovery, collaboration services, and so on.

C2 service infrastructure provides basic command and control service capabilities, which is further subdivided into two layers: C2 common services layer and C2 support services layer. The lower layer (i.e., C2 common services layer) can provide foundational and common C2 services, including locator services, report management services, entity management services, fusion services, etc; the higher support services layer provides support to the forming of C2 Communities of Interest (CoI), such as process management services, alert services, resources

management services, visualization services, and so on.

The top one is C2 mission capability package (MCP) layer, which is used to form different applications-oriented or missions-oriented CoIs according to different operational missions, equaling the Internet Service Provider (ISP). Through the development, wrapping and deployment of business services coming from all Services and different domains, the services can be published to public as C2 MCPs manner, so as to agilely respond to changing battlefield environment.

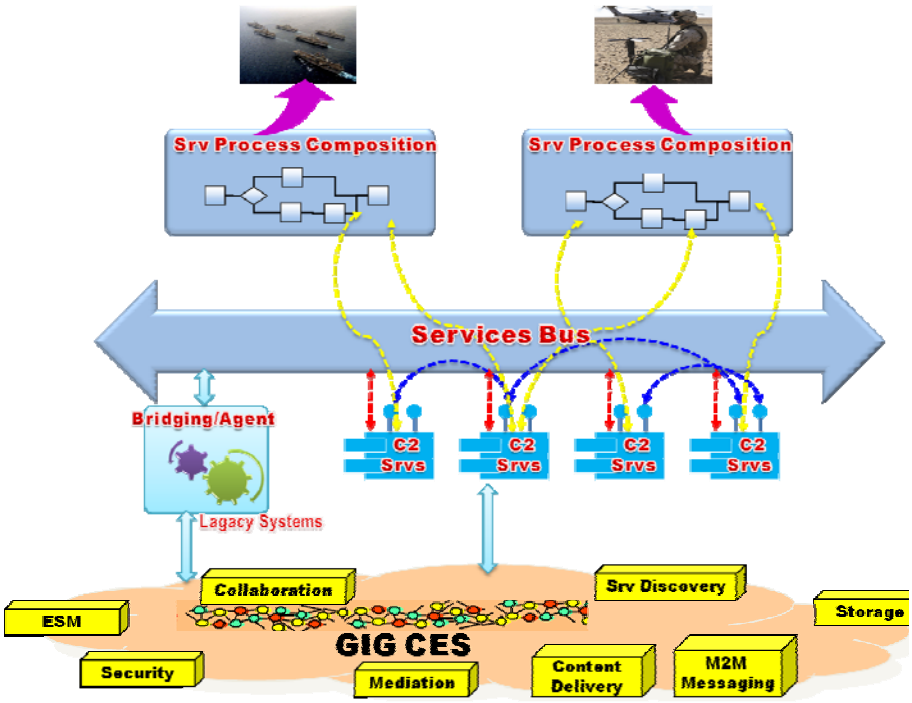

Fig. 5 Mission-oriented C2 architecture application pattern

The above figure shows application pattern of the C2 architecture facing different operational missions using our proposed architecture. Each layer in this architecture may be dependent on different SOA technologies, and this architecture itself does not restrict the use of specific technologies, such as Web Services, SCA Services [9], Restful Web services [10], etc. Our proposed service invocation method in following Section 4 is targeted at the popular Web Services.

4. Our proposed C2 service invocation method

4.1 Basic ideas

Based on the principle of HTTP proxy, a lightweight web services invocation method is presented in this paper. The prerequisite of the proposed method is that military LAN environment is considered secure, i.e., it is allowed to use some open or standard transport protocols such as HTTP, whereas it must follow military security standards and regulations, and use military transport protocols for messaging in WAN environment. The basic principle of our proposed method is as follows: in LAN, standard Web Services are used to messaging, i.e., SOAP via HTTP. Then, we deploy HTTP proxy software at the boundary of the LAN to intercept

HTTP packets. The SOAP message in the HTTP packets is delivered in the WAN via the military transport protocol satisfying military security requirements instead. At the receiver end, the same HTTP proxy is used to convert and restore standard invocation way (i.e., SOAP via HTTP) in LAN, so as to achieve the services invocation. The proposed C2 service invocation method based on HTTP proxy is shown in Fig.6:

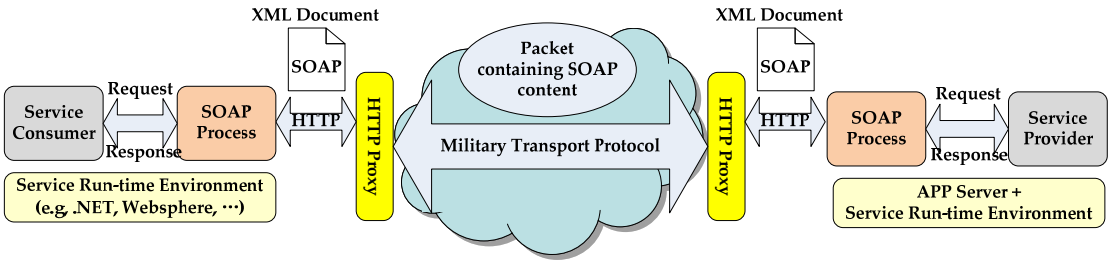

Fig. 6 Our proposed method

4.2 Detail implement

The core of our proposed C2 service invocation method is to develop the HTTP proxy (including client and server) software.

HTTP proxy client and the service requester are deployed in a same LAN. HTTP proxy client is mainly responsible for receiving the service request message (SOAP via HTTP) from the service requester, parsing the request message, and forming into the packets meeting military transport protocol. At the same time, it is used to receive service message packets (including results of service request processing) from HTTP proxy server end, and return the results to the service requester in standard service invocation way.

HTTP proxy server and the service provider are deployed in a same LAN. The process of HTTP proxy server is similar to HTTP proxy client. That is, it mainly responsible for receiving service invocation messages (including service request) from HTTP proxy client end, parsing the messages, and sending the service request packets formed the original WS messages to the service providers. And, HTTP proxy server returns the request results from the service provider to HTTP proxy client via the military transport protocol.

The procedure of HTTP proxy is shown in Fig.7:

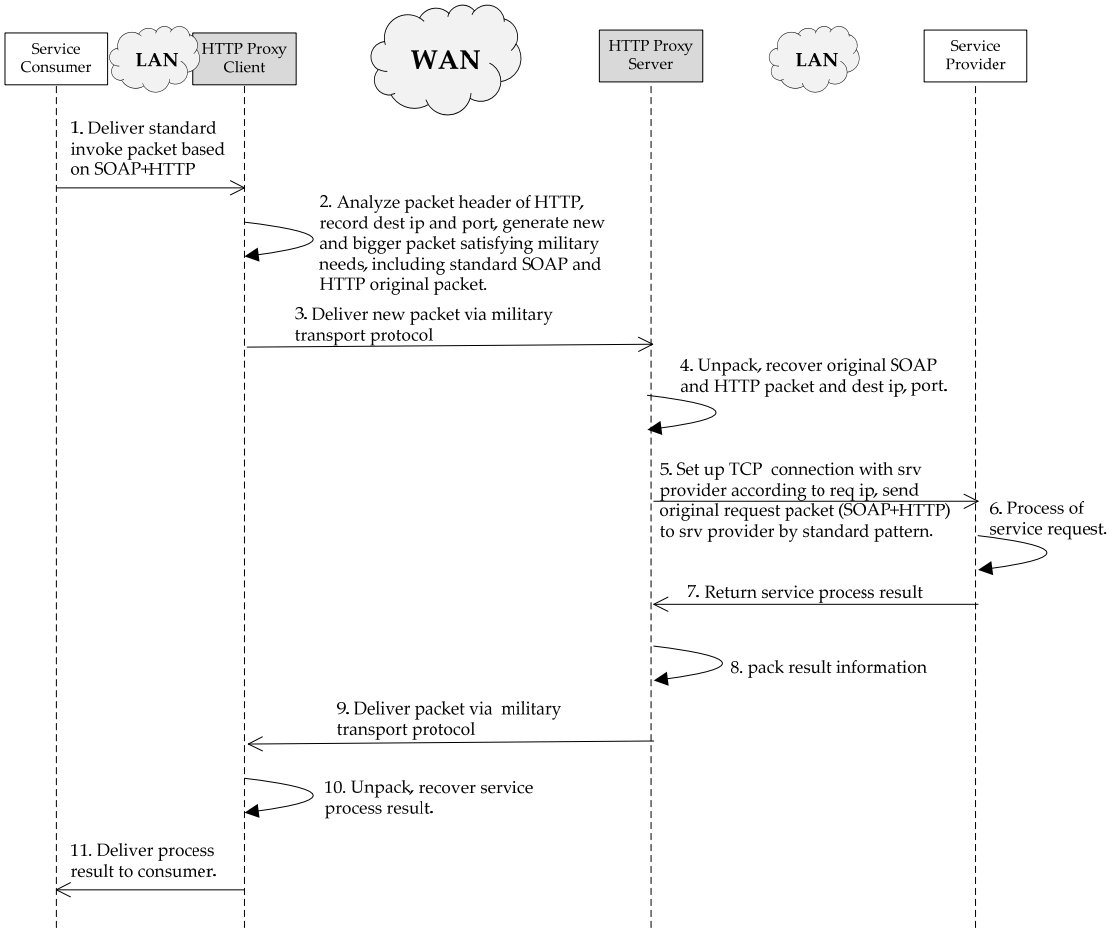

Fig. 7 The procedure of our proposed method

1). the standard invocation packets (SOAP) is delivered via HTTP;

2). at HTTP proxy client end, the packet header of HTTP is analyzed, and the destination IP address and port are recorded here. Then standard SOAP and HTTP original packet are packed to generate new packet satisfying military needs;

3). new packets are delivered via military transport protocol in WAN;

4). at HTTP proxy server end, the received packets are unpacked; the original SOAP, HTTP packets and the information about destination IP address/port are recovered;

5). HTTP proxy server is set up TCP connection with service provider according to request IP address, and sends original request packet (SOAP via HTTP) to service provider by standard WS pattern;

6). service request is processed at service provider end;

7). service process results are returned to HTTP proxy server;

8). HTTP proxy server packs result information to generate new packets;

9). the new packets are delivered via military transport protocol in WAN;

10). when HTTP proxy client receives the military packets, it unpacks these packets and recovers service process result;

11). at last, the process result is delivered to consumer. The whole procedure finishes.

5. Experimental Results

To verify the effectiveness and feasibility of the proposed method, we carry out experiments about the efficiency of C2 service invocation in a mixed WAN and LAN environments.

A C2 service named *computing service for parameters of target trajectory* is selected as use case, developed by using BEA WebLogic, Java language. At the client end, client program for service invocation has been developed by .NET tool, Microsoft Corp. Both HTTP proxy client and server software are developed by Visual C++ 6.0.

In order to verify the feasibility of HTTP proxy, a method in which Socket is used to transport message between two HTTP proxies is firstly implemented; then another method is that Socket is replaced with military transport protocol to achieve the messaging between two HTTP proxies. Therefore, there are three C2 service invocation methods compared in this paper:

1) C2 service invocation not using HTTP proxy, i.e., the standard Web Service invocation, called NO_PROXY;

2) Based on HTTP proxy, messaging between two HTTP proxies is implemented by using Socket, denoted SOCK_PROXY;

3) Based on HTTP proxy, messaging between two HTTP proxies is implemented by using military transport protocol, named MIL_PROXY.

The bandwidth of WAN and two LANs are 100Mbps. Hardware and software configurations of experimental machines are shown in Table 1:

Tabel.1 Experimental environment and configurations

	Client	HTTP Proxy		Application Server
		HTTP Proxy Client	HTTP Proxy Server	
Computer	DELL 9200	HP 4100	HP 4100	HP MS530
Hardware	CPU：3.0GHz Memory：2G	CPU：3.0GHz Memory：2G	CPU：3.0GHz Memory：2G	CPU：Intel Core2 E6420 Memory：2G
OS	Windows XP/SP2	Windows XP/SP2	Windows XP/SP2	Windows XP/SP2
IP Address	192.168.11.3	192.168.11.2	192.168.22.3	192.168.22.188
Software	Client invoking srv; .NET Container;	HTTP Proxy Client;	HTTP Proxy Server;	BEA WebLogic Server 9.0; BEA Web Services Container; C2 Computing Service;

(1) Experiments about the total time of C2 service invocation

The total time of C2 service invocation is firstly recorded by using three ways, respectively.

In NO_PROXY mode, the total time of C2 service invocation is shown formula (1):

$$T=T_{c\to s}+T_s+T_{s\to c} \qquad (1)$$

In other two proxy mode, the total time of C2 service invocation is computed by using formula (2):

$$T=T_{c\to proxy\_c}+T_{proxy\_c\to proxy\_s}+T_{proxy\_s\to s}+T_s+T_{s\to proxy\_s}+T_{proxy\_s\to proxy\_c}+T_{proxy\_c\to c} \qquad (2)$$

where, c denotes client/consumer that needs to invoke service; s denotes service provider; *proxy_c* and *proxy_s* denote HTTP proxy client and server, respectively. T_s

denotes the time of service processing at service provider end; $T_{x\to y}$ denotes the time of transmission or processing from x to y, x or y may represents any of client, service provider, HTTP proxy client, or HTTP proxy server.

In each invocation method, we record the average value of the total time of invocation for 10 times, 50, 100, 200, 300, 500, 800, and 1000, respectively. The results are shown in Figure 9:

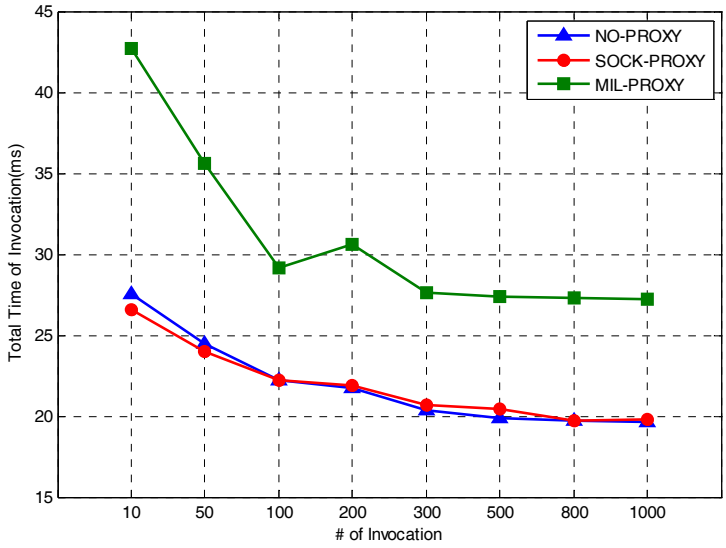

Fig. 8 A comparison on the total time of invocation of the different methods

It can be seen from the above figure that the performance of SOCK_PROXY is almost equivalent with NO_PROXY, which shows HTTP proxy itself does not consume too much time, and the approach based on HTTP proxy is feasible. Replaced with military transport protocol, we can see that the performance of service invocation gives slightly higher total time than other two methods, but within the scope of tolerance. Reasons for the performance decline is mainly due to increasing security process of military transport protocol.

(2) Experiment about the time in MIL_PROXY mode

In HTTP proxy using by military transport protocol, i.e., MIL_PROXY, we track the total time of service invocation and the total process time of HTTP proxy client for each invocation. Figure 6 shows the experiment results. We can see from the figure that the difference between the two curves keeps consistent nearly, which shows that the processing performance of the developed HTTP proxy is stable.

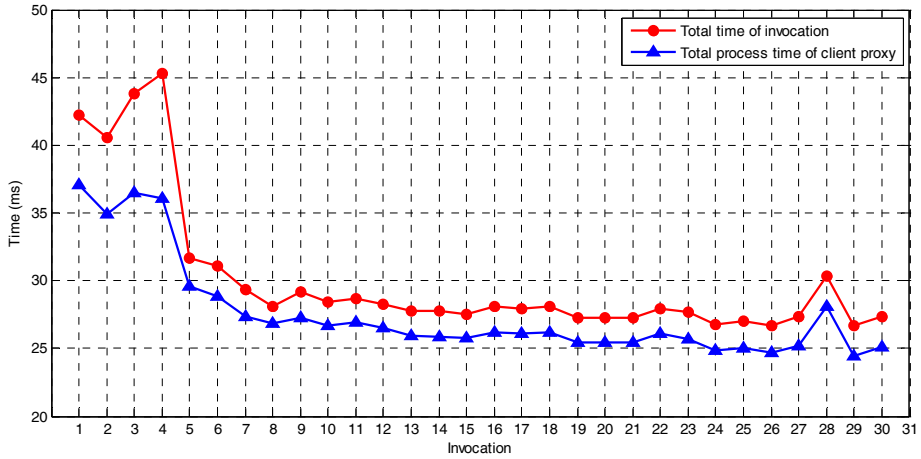

Fig. 9 The time results of MIL_PROXY

(3) Experiments about the quantity of data

In last experiment, we record the quantity of data (bytes) sent by the client/consumer and server, respectively. The result is shown in Table 2. Note, the results include the size of two packets (1st and 2nd) of one request or response.

Table.2 The result of size of packet(Bytes)

		NO_PROXY	SOCK_PROXY	MIL_PROXY
Client	The size of request packet (1st)	407	407	474
	The size of request packet (2nd)	305	305	352
APP Server	The size of response packet (1st)	25	25	66
	The size of response packet (2nd)	572	572	617

It can be seen from the table that three methods have the same order of magnitude about the quantity of data. The size of packet for SOCK_PROXY is equal to NO_PROXY, and MIL_PROXY gives a slight more size of packet than other two methods. This phenomenon is normal, because the characteristics of military transport protocol decides that it will increase the quantity of transmission packet than using Socket.

From the entire experiments, we can see that our proposed C2 service invocation method based on HTTP proxy is simple, effective and feasible. Most importantly, it can meet the security and real-time requirements in military environment. Therefore, the proposed method is a lightweight approach implementing C2 service invocation in applications of service-oriented C2 software architecture.

6. Conclusion

SOA and Web Services are more and more widely applied in C4ISR, especially in network centric C2. So, how to achieve service invocation meeting the military requirements to gain service capabilities on demand in network centric environment has become a research hotspot concerning service technology. In this paper, a service-oriented C2 software architecture named SO-C2SA was put forward, and the C2 service invocation issue was discussed in detail. Taking into account the security requirements of military WAN environments and the current large-scale use of Web

Services, we presented a new lightweight C2 service invocation method based on HTTP proxy. We deployed HTTP proxy software at the boundary of the LAN system, and used it to intercept HTTP packets and SOAP messages. Then, the packets were delivered via military transport protocol in WAN to implement the service invocation. The experiments of invocation efficiency demonstrated that our proposed method is simple, effective and feasible, and can meet the requirements of security and timeliness in military environment.

References

[1] Thomas Erl, Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall PTR, ISBN: 0-13-185858-0, 2005.

[2] Eric Newcomer, Greg Lomow, Understanding SOA with Web Services, Addison Wesley, ISBN: 0-321-18086-0, 2004.

[3] Tolk, A.; Gaskins III, R.C. Challenges and Potential of Service-Oriented Architectures for Net-Centric Operations. Meeting Proceedings RTO-MP-HFM-136, Paper 5. Neuilly-sur-Seine, France: RTO. Available from: http://www.rto.nato.int/abstracts.asp. 2006.

[4] Wang, H., Ding F., Xu H. Feasibility Study on Building Military Grid Applications based on SOA/Web Services. Fire Control and Command Control, 2007, 32(7): 69-72.

[5] UNCLASSIFIED Capstone Requirements Document (CRD), Global Information Grid (GIG), U.S. Department Of Defense, 2001.

[6] Dawn Meyerriecks, Net-Centric Enterprise Services(NCES), The 3rd NCES Workshop, http://www.afei.org/brochure/4AF9/Meyerricks_Overview.pdf, May 2004.

[7] Shaffer G. R. Composing and Orchestrating Mission Capability Packages Through Business Process Execution Language (BPEL). Command and Control Research and Technology Symposium, 2004.

[8] Department of Defense Global Information Grid Architectural Vision, Vision for a Net-Centric, Service-Oriented DoD Enterprise, v1.0, U.S. Department of Defense, June 2007.

[9] Service Component Architecture. IBM developerWorks, http://www.ibm.com/developerworks/webservices/library/specification/ws-sca/?S_TACT=105AGX52&S_CMP=cn-a-ws, 2005.

[10] Fielding R T. Architectural Styles and the Design of Network-based Software Architecture. Doctorial Dissertation, Dept. of Computer Science, Univ. of California, Irvine, 2000.

Heng Wang received his B.E. degree in Computer Communication Engineering and Ph.D. degree in Computer Science from Nanjing University of Science and Technology, Nanjing, Jiangsu Province, China in 1999 and 2004, respectively. He is currently a senior engineer in the National Key Laboratory of Science and Technology on C⁴ISR, Nanjing, China. His research interests include C⁴ISR system software integration, information grid, information dissemination and QoS routing.