'FUTURIZM-2014, (14TH & 15TH March 2014), 'P.D.E.As College of Engineering, Manjari[Bk]', 'IIP Cell,'Manjari Road, Hadapsar, Pune-412307. India.', 'Chief Editor:Dr.K.R.Harne', Editors: Prof R V Patil, Prof Niraja Jain

International Journal of Innovative Technology & Adaptive Management (IJITAM)
ISSN: 2347-3622, Volume-2, Issue-2 , November, 2014

Security of Web Mashups: a Survey

Ms.R.P.Jadhav
G.H.Raisoni Institute of Engg. And Technology,Pune

***Abstract*— Evolution of Web 2.0 applications has changed the outlook of business models and companies. Organizations need to rethink their communication, marketing and sale channels and how their employees and customers interact together internally and externally. Following this new trend, they also need to adopt their IT infrastructure and enhance their online presence and services in order to stay competitive in their businesses. Through this technological transition to Web 2.0 paradigm new security and privacy issues arise which should be taken into consideration to protect the whole Rich Internet Application (RIA) components. Web 2.0 has also introduced new possibilities for a better human computer interaction via rich internet applications such as Mashups that provide a user-driven micro-integration of web-accessible data. At the moment Mashups are mainly used for less important tasks such as customized queries and map-based visualizations; however they have the potential to be used for more fundamental, complex and sophisticated tasks in combination with business processes. In this paper, the security and privacy aspects of Mashup Architecture and some existing challenges will be discussed in more details.**

Keywords- Security; Privacy; Mashup; SOA; Web 2.0; Business Process; Business Intelligence

1. INTRODUCTION

The recent and rapid expansion of Web 2.0 has placed considerable pressure upon industry to institutionalize new technologies and conform them to emerging standards. While agreement on the scope of the term Web 2.0 does vary, O'Reilly provides a commonly accepted definition, noting this to include a range of enhanced services including web services, wikis, blogging, BitTorrents, and syndication [1]. The rapid growth of Web 2.0 has also introduced a number of new design patterns and architectural styles in web development. One of the notable techniques involves the mashing up of information from existing services to deliver value-added new services. This process effectively includes the drawing of content from several sources in order to create a new content or service. The resulting web page is finally referred to as mashup of the existing content. While mashup services bring flexibility and speed in delivering new valuable services to consumers, the legal implications of this technology are significant [2].

1.1.What is Web 2.0?

Web 2.0 is both a usage and a technology paradigm. It is a collection of technologies, business strategies and social trends. It is way more dynamic and interactive then Web 1.0 [3]. The offered contents are nowadays not only provided by a few companies. Many users can create and spread their custom information for example with Wikis, Blogs, Image- and Video portals like Flickr and YouTube. A Wiki for example, is multiauthored and dynamic instead of being monoauthored and static. Even applications have become dynamic in Web 2.0: Disparate components can combine from entirely new mashups, in contrast to the Web's static form-based applications [4]. Social networks, as for example MySpace, Facebook and studiVZ are very common Web 2.0 applications. These

International Journal of Innovative Technology & Adaptive Management (IJITAM)
ISSN: 2347-3622, Volume-2, Issue-2 , November, 2014

applications are highly appreciated by many people as a source of information and communication.

1.2 Understanding Mashups

A mashup is a technique for building applications that combine data from multiple sources to create an integrated experience. Many mashups available today are hosted as sites on the Internet, providing visual representations of publically available data. [5] Architecture of a Prototypical Mashup Although there is a great variation in the user interface and the sources of data for many mashups, we can still derive common architectural patterns that they all share. For example, all mashups are RESTful in nature (they conform to the Representational State Transfer principles). Figure 1 shows an architectural rendering of a typical mashup.

Data

The core element of any mashup is the data being aggregated and presented to the user.

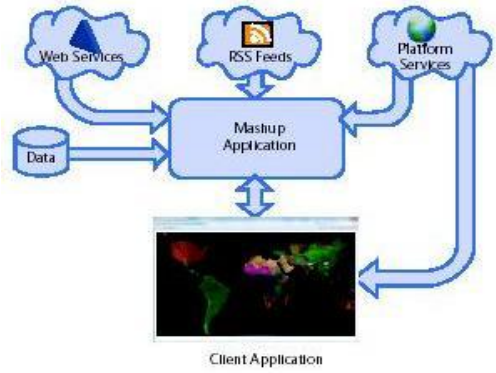

Figure 1: Mashup Architecture

Although the above diagram depicts the source of the data as a database, the concept of a mashup does not require a database that is local to the mashup software or the client. The data can strictly come from Web services where data is serialized to XML or JSON (this is the most common pattern in Internet-based mashups). There are architectural trade-offs to be made from storing the primary data in a local data store and accessing the data with every request. As mashups move from being Internet-based applications to internal to the enterprise, they tend to depend less on external data stores.

RSS feeds

Use of RSS (Really Simple Syndication) feeds is a common source of primary or supplemental data for mashups. RSS feeds are easy to consume as they are XML documents, and many libraries exist to manipulate the feeds. The format and specification for RSS is well documented and understood with only a few variations from version to version. The extensibility of RSS is also well known, as demonstrated by the number of extensions in use today, such as adding attachments to the feeds, creative commons licensing information and location information.

Web services

It is also common to include calls to Web services within mashups. It is common to see both WSDL-based Web services and REST-based Web services, with some services exposing both styles. Web services can be used to provide additional data or used to transform the data being mashed up. For a map-based mashup, the data may only contain street addresses and a call to a WSDL or REST-based Web service may be required to translate the street address to a Longitude / Latitude coordinate for the map. In case of the world wide web, mashups are websites, web - pages, web - services or applications which combine data, information, music, geotracks from more than one source into one application, service or website. This is generally achieved by using third party application programming interfaces (API's) or open technologies such as Ajax, PHP or syndicated feeds like RSS or ATOM. Based on service composition in Service Oriented Architecture (SOA) concepts, mashups are flexible and dynamic services. This technology also enables a dynamic way of service reusability unlike static "cut & paste" re - usability. Web 2.0 takes us to a deeply service- oriented world, where we can exploit everyday services such as news, instant messaging and blogging via our desktops, mobile phones, PDA, BlackBerry and so on. Most of the time people are not aware of the

International Journal of Innovative Technology & Adaptive Management (IJITAM)
ISSN: 2347-3622, Volume-2, Issue-2 , November, 2014

actions "behind the scenes" in terms of the massive network of service interactions. Such services occasionally seem as if they are re - writing the existing services. In fact they are smart compositions (mashups) of the existing ones. Known as "Web application hybrid" this smart way of combining the content from more than one source into an integrated experience is called "mashup" technology [6]

Web browsers are becoming the single stop for everyone's computing needs including information access, personal communications, office tasks, and e-commerce. Today's Web applications synthesize the world of data and code, offering rich services through Web browsers and rivaling those of desktop PCs. Browsers have evolved to be a multi-principal operating environment where mutually distrusting Web sites (as principals) interact programmatically in a single page on the client side, sharing the underlying browser resources.

Consider a scenario in above figure wherein an HTML file (possibly including scripts) sent from webmail.com and an HTML file sent from stocks.com run on the client browser. These HTML files are really delegates on be-half of webmail.com and stocks.com, respectively, using resources on the client to improve the interactivity of the services.

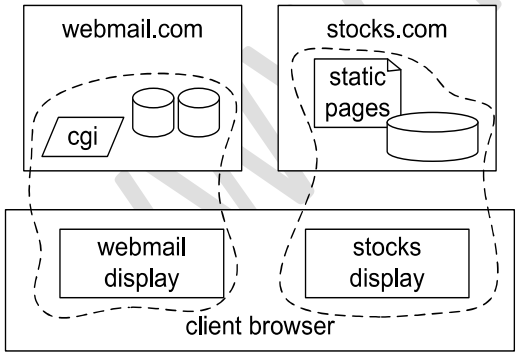

Figure 2. A javascript code running o client browser is really just distributed component of the web service that provided the code.

In this scenario, the sites are mutually distrusting principals sharing the browser's resources (display, memory, CPU, network access). This resembles the PC operating environment where mutually distrusting users share host resources. However, unlike PCs that utilize multi-user operating systems for resource sharing, protection, and management, today's browsers do not employ any operating system abstractions, but provide just a limited binary trust model and protection abstractions suitable only for a single principal system: There is either no trust across principals through complete isolation or full trust through incorporating third party code as libraries. The browser abstraction for the former is FRAME; frames enable interactive (script-enhanced) Web services to occupy neighboring display real estate, but the component cannot interact.

The abstraction for the latter is SCRIPT which allows third-party scripts to be included as library code; the embedded cross-domain scripts enjoy full trust from its includer and can access the includer's data, display, and access to back-end server resources. With these limited existing browser abstractions, Web programmers are forced to make tradeoffs between security and functionality, and often times sacrifice security for functionality. In the scenario above, a web program either segregates HTML content from webmail.com and stocks.com into separate frames denying any communications or embed their scripts as libraries into a containing page allowing intimate interactions. As we can see, controlled interactions may be desired: If the stocks.com server offers a limited Web interface that other servers such as webmail.com may access, then the browser should allow similar communication between the corresponding components running on the client. This controlled communication among otherwise isolated client components is not attainable in today's browsers.

2. RELATED WORK

Recently, a new wave of "web operating systems" [7] (e.g., YouOS [8]) have emerged. These sites present a traditional desktop user interface, complete with a window manager. The applications run natively in JavaScript. All are hosted on the same domain as the web desktop, and thus have unlimited access to one another. This lack of isolation, comparable to the 1995 PC

International Journal of Innovative Technology & Adaptive Management (IJITAM)
ISSN: 2347-3622, Volume-2, Issue-2 , November, 2014

desktop, requires the user to completely trust every application that is run.

Subspace [9] provided a cross- domain communication mechanism that is designed to run on current browsers without any additional plug-ins or client-side changes. Subspace uses the browser's existing document .domain property to communicate with isolated subdomains, which are in turn used to draw in scripts from other domains. However, Subspace requires significant work on the part of the web developer to use correctly, and does not provide the resource constraint options of FRIV. We believe browsers should provide built-in cross-domain interaction primitives.

Crockford recently proposed the JSONRequest [10] communication mechanism for asynchronous cross- domain data retrieval from a remote server. The proposal was motivated by scenarios where the cross-domain SCRIPT tag was being used to execute code when only data was really required. The JSONRequest's usage is similar to XMLHttpRequest, but it is not constrained by the Same-Origin Policy. Lifting these same-origin restrictions is safe because cookies are not sent, because the request includes a header indicating the source of the request, and because the server's reply must indicate the server is aware of the protocol and hence its security implications. JSONRequest transmits data in the JSON format, but its security applies equally to XML or other formats. A cross-domain JSONRequest is the client- side equivalent to a cross-server TCP request, and thus JSONRequest complements MashupOS well. Alternatively, one can simulate JSONRequest in MashupOS by creating a FRIV that communicates with its home domain and then passes the received data back to the outer SERVICEINSTANCE. Crockford also identified the security limitations that affect today's cross-domain mashups. In response, he proposed a new HTML tag, the MODULE tag, to partition a page into a collection of modules [10]. A module groups DOM elements and scripts into an isolated environment; socket-like communications are allowed between the inner module and the outer module. To isolate the module from the origin server, modules may not make network requests. Thus, modules are equivalent to null-principal FRIVs.

Cross-document messages [11] are a proposed browser standard that would allow cross-domain frames to send string messages to each other on the client side. Cross-document messages are thus similar to the messaging capabilities of full-principal FRIVs. They are implemented in the Opera browser. We expect that with better data type support, automatic layout capabilities, and other syntactic sugar provided by FRIV, wider deployment and use of this messaging paradigm can be achieved.

3. NEED

In Web 2.0 there are many risks if all users are able to publish their custom informations. Just take a look at Wikipedia. It is well - known and every user is able to add and edit the content. There is no need to say, that it is impossible to guarantee that Wikipedia provides only valid informations. No one can proof the whole Wikipedia for correctness or can control that the users validate their informations. Portals like YouTube allow the registered users to upload files like images, videos, archives, programs and many more. These uploaded contents can contain malicious code which could be propagated by people opening these contents. The Yamanner worm was spammed to Yahoo Mail users. When the attachment was opened, the worm sent a copy, outside the browser window, to everyone in their contact lists [12]. There are numerous ways to attack browsers or computers with Web 2.0 vulnerabilities. The most common attack strategies are, the both well known, cross- site-scripting (XSS) and cross-site-request-forgery (CSRF) [13].

3.1 Combining Services

As already mentioned, Mashup Systems combine several services. For Example, a mashup can combine Google maps and a GPS System. The provider of this mashup could track you, wherever you are using this system. If you are using it in the car he always knows where you are driving, even how fast you are driving. If this system is combined with a list of restaurants, shopping

International Journal of Innovative Technology & Adaptive Management (IJITAM)
ISSN: 2347-3622, Volume-2, Issue-2 , November, 2014

centres, patrol stations, and so on the provider can guess what you are doing the whole day long!

3.2 Combining Information

Everyone of us already provided some private information in the internet. Amazon knows the name, age, address, bank data, what products we are interested in and how much money we spend every month of his users. Google knows which information we are looking for, is able to analyse our emails if we have a "gMail" Account. With "Google Docs" they can take a look at our work. You can create your personal Google Startpage. StudiVZ has very personal information about the most users,they also store uploaded pictures, a list of friends, the interests of the users and informations about their study. There are many services in the world wide web that save personal data. If just the three mentioned services like Amazon, Google and StudiVZ are combined to one single system, the provider would know rather everything about the users. He could offer very personal advertisement, influence your search results, offer personal information to companies, where you are applying to.

3.3 Examples

If a Mashup do more then only merge location - based information with other online sources, like combining data with search functions from another Mashup, then it is possible to create an application that amounts much more than the sum of its parts. A simple Example for instance, www.chicagocrime.org combines Google Local's maps with Chicago's crime database, pinpointing the city's crime hotspots. At www.housingmaps.com, houses for sale advertised on craigslist.org are injected into Google Maps. So users are able to see the location of properties they are interested in. Now a hacker could feed the location mashup with false data to a crime location, for example to help raise property prices in a particular area by making it crime free.[14]

To demonstrate how easily mashup Systems can combine information in a way that invades people's privacy, computer consultant Tom Owad mashed book wishlists posted by Amazon users with Google Maps. The wishlists often contained the user's first and last name, as well as the city and state in which they lived. Enough information to find their full street address from a search site such as Yahoo People Search. These data are enough to get a satellite image of their home from Google Maps. Thats an easy way to get your full name, your adress and a photo of your neighbourhood [14].

4. CONCLUSION

The shift away from traditional Web 1.0 is forced by the growing need for more efficient information sharing, collaboration and business processes. Mashup architecture is one of the outcomes of the Web 2.0 paradigm that has been widely accepted and used for user-centric information processing. At the moment Mashups are mainly used for less fundamental tasks such as customized queries and map based visualizations; however it has the potential to be used for more fundamental, complex and sophisticated tasks too. As more serious applications make use of Mashup architecture, there is a growing need to study security and trust issues in companies. Furthermore, it is important for enterprises to understand these concerns to develop a holistic security strategy. Through application of user-generated information resources such as Mashups, the possibility of confidential data leaks increases. The basis of every successful organization is its information resources. By employing the Mashup architecture, organizations enable their employees and trusted stakeholders to get access to this valuable data and information in an easy and appropriate way. As a result the Mashup governance and personal / organizational data sharing policies are of great importance for organizations. Existing security technologies and methodologies can only support the information resources against tradition security threats; however, for the emerging data sharing approaches such as Mashups new security models are required.

5. REFERENCES

[1] T. O'Reilly: "What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software", O'Reilly Media Inc., September 2005

International Journal of Innovative Technology & Adaptive Management (IJITAM)
ISSN: 2347-3622, Volume-2, Issue-2 , November, 2014

[2] Joe Zou, Christopher J. Pavlovski: Towards Accountable Enterprise Mashup Services, IBM Corporation, St. Leonard NSW Australia, e-Business Engineering, 2007. ICEBE 2007. IEEE International Conference .
[3]San Murugesan: Understanding Web 2.0, IT Pro – published by IEEE Computer Society,2007
[4] Mary Ann Davidson (Oracle), Elad Yoran (Security Growth Partners): Enterprise Security for Web 2.0, Computing, Science & Engineering USA, 2008
[5] Holmes, Josh. "Enterprise Mashups". MSDN Architecture Journal. MSDN Architecture Center.
[6] Semih Cetin, N. Ilker Altintas, Halit Oguztuzun, Ali H. Dogru, Ozgur Tufekci, Selma Suloglu: Legacy Migration to Service-Oriented Computing with Mashups, Department of Computer Engineering, Middle East Technical University, 2007
[7] Big WebOS roundup -10onlineop- erating systems reviewed. http:// franticindustries.com/blog/2006/12/21/ big-webos-roundup-10-online-operating-systems-reviewed/.