

Optimizing Mainframe Applications for Reuse with Modern Technologies

Can you really gain flexibility without sacrificing security?

CONTENTS

Mainframe Modernization via SOA	1
Basic Security Differences: SOA vs. the Mainframe Application	1
Unlocking Mainframe Applications	1
Defining States	2
Five Options for Mainframe Access Using Services	2
Finding the Right Solution	6
Guidelines for Division of Labor	6
Mainframe Assets: Too Valuable to Extend?	7
Making it Easy with Verastream	7
About Attachmate	7

Optimizing Mainframe Applications for Reuse with Modern Technologies

Can you really gain flexibility without sacrificing security?

The first mainframes were built for deployment in trusted environments. Security concerns focused more on ensuring proper application access than on locking the network down. That's because the applications ran on platforms that were totally self-contained; the concept of a distributed environment was not even on the horizon.

But today's IT staffs are under increasing pressure to open System z and other host applications to a variety of settings outside that original realm. There are many options for doing this. However, you must address significant security implications when you extend your legacy assets beyond their original use. This is especially important when opening a legacy application to a distributed and more open network infrastructure.

Mainframe Modernization via SOA

In uncertain financial times many business decisions are determined by the cost factor. IT decisions are no exception and that's one reason IT organizations worldwide are tackling mainframe modernization with an increasingly popular method: SOA enablement. SOA enablement is directly cost-effective and yields a rapid ROI. It's also efficient and doesn't require rebuilding or replacing legacy applications—and it mitigates risk.

IT professionals considering SOA enablement of mainframe assets typically want to address concerns such as the following:

- How can I make mainframe data more useful without invasive changes to my legacy application inventory?
- How can I leverage and append the proven safeguards that are in place?
- How will I know if I've broken or invalidated the security that's in place?
- When I start distributing access, how can I maintain control, pass audit, and make sure security is still adequate?
- What level of risk is appropriate in getting the quality of service I need?
- Will I be able to limit levels of visibility once application x is talking to application y?

- Who will be able to gain access and what systems will ensure the proper access?

Forms of SOA enablement, specifically front-ending legacy applications with a "wrapped" service tier, allow a good level of control over these concerns. This paper does not focus on the service-wrapping technologies, but instead provides some guidelines on reconciling the use of services against your own IT environment with the right application-integration solution.

Basic Security Differences: SOA vs. the Mainframe Application

The power of the pure SOA approach lies in its flexibility through service composition and service reuse. Services in a formal SOA, by nature, do not know how they will be used as reuse is open-ended. This "ambiguity" is by design. Developers employ services as needed, in differing ways for differing purposes.

To provide oversight to critical resources, an SOA counts on forms of governance to track services and ensure proper usage. The oversight is external to the services and beyond the control of the services themselves. It is a trust relationship in the sense that the services "trust" some part of the infrastructure is protecting them.

This characteristic is fundamentally different from the traditional legacy security infrastructure, which knows directly who and what is using it, from end to end. Mainframe applications, and the systems they reside on, typically track and control the use of their data themselves. This dissimilar method for securing mainframe applications can complicate the building of an enterprise SOA.

Unlocking Mainframe Applications

When considering the creation of services that front-end legacy applications, you should assess the importance of guarding the use of the legacy components exposed as services. If that importance is high, then you have to determine the accepted method of tracking and protection.

From the perspective of the mainframe, all users of an application are at some level users of mainframe sessions. It's a relationship that lasts for the duration

of the legacy application's use. By having a strong authentication method to grant use of a session, the mainframe and the legacy application can track the exact who, what, and when of any legacy interaction. The business operations that are dependent on these applications typically count on this trust relationship and the guarded methods for use.

To allow use of these guarded mainframe applications within an SOA, the right level of preparation and understanding can be invaluable.

Defining States

This paper examines several ways to safely reuse mainframe application data and logic in an SOA. In that context, we'll look at best practices for ensuring proper host application involvement. But first let's establish working definitions of some key terms used in this paper:

- **State** conveys the session status of interactions between two computing devices, or (more specifically for this paper) between a computer application and a user ID, or a service and the service client. Simply put, state refers to the degree of information retained and available about ongoing communications for the duration of the session.
- **Stateless interactions** describe communications in which services do not retain information, such as stored state variables or their values, between one event and the next. Each user-initiated action is treated as an isolated incoming application request, with no extraneous information needed or kept.
- **Stateful interactions** describe communications in which services do retain certain information (or histories) between one event and the next, usually to track status and context within the application. For example, an application might allow an authorized user ID to perform multiple actions against it, while maintaining a single continuous session with the user ID for the duration of the activities.

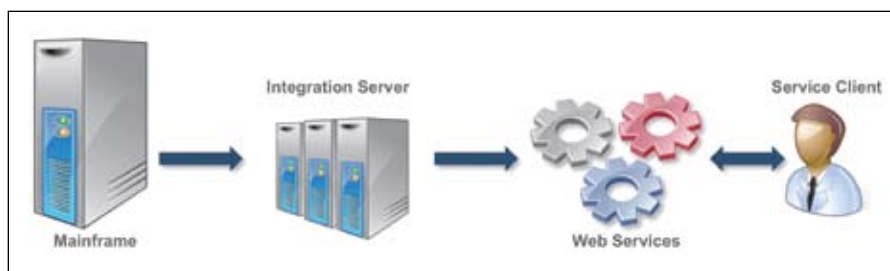
Five Options for Mainframe Access Using Services

With the above definitions in mind, we can discuss five general options for allowing services access to the mainframe. With each approach, you will see a technical description, accompanying diagram, advantages, disadvantages, and appropriate use.

1. Open/stateless with no locking

The most basic approach is to create stateless services that access and drive a legacy application for the client or user. These services create their own session to the host and then they make themselves available for use externally. The mainframe application talks indirectly to the client application via these stateless services over existing open lines without regard to the location of the client.

This approach is a simple extension of the legacy application, with no changes made to the legacy application or the network. It assumes the same safe network environment that most legacy applications were built with.



Stateless services create their own session for the mainframe application to talk indirectly to the client application over existing open lines.

Advantage: This method is noninvasive and lets you reuse valuable assets without rewriting or rebuilding host code, which can be risky. It has the lowest level of change to system and application access, so you can get a very rapid ROI.

Disadvantage: Because the lines are open and unencrypted, there is an assumption of a safe network. And because the legacy interactions are stateless with no tracking, the application-client communications are anonymous.

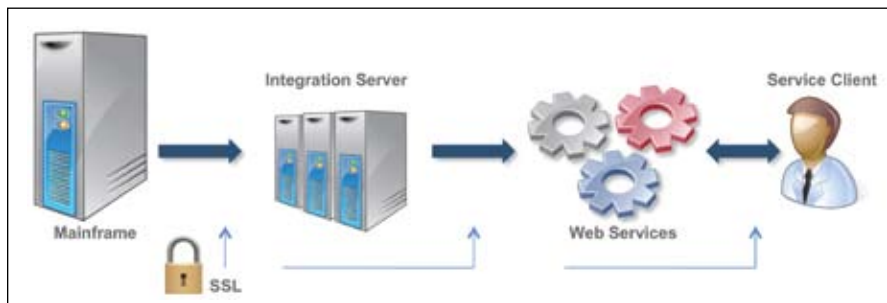
Appropriate use: While this method might seem nonsecure for legacy use, it has actually proved to be appropriate and successful in certain cases, most notably for tasks where security is implicit by the nature of the data and the current network layout. In fact, many organizations using Attachmate® Verastream® Host Integrator have implemented this indirect-communication method for quick access to legacy applications that do not need extended session state. While it's not typically taken beyond the existing safe network zone in IT, it ensures both good performance and low impact on the source host system. That's why it's often used in a protected, or "glass house" environment, where all users with network access are known and credentialed.

For example, a system administrator might want to track the health of various applications through a console dashboard. When mainframe applications are involved, this task might call for health-monitoring

services to populate the dashboard with data – an activity commonly employed in Verastream scenarios. Because you discretely query the monitoring services only to verify health, it's ideal to make them nonintrusive and stateless. Also, with service usage targeted solely to an IT audience within the IT-controlled environment, you can assume the security of both the users of the services and the network they traverse. That means you don't have to encrypt network conversations. So open/stateless access without locking allows a highly responsive yet low-overhead method that's appropriate for the needs of client and host application alike.

2. Open/stateless with lines locked down

When the use of services spans beyond the controlled network environment, you must add a layer of security by – at minimum – locking down your lines. You still access the mainframe indirectly through a set of stateless proxy services in the same way as above, with the services opening and controlling the sessions to the mainframe. The difference in this method is that line security, via SSL or equivalent, has been introduced.



Stateless services create a session for the mainframe application to talk indirectly to the client application, as in option 1, but the lines are locked down.

While this change seems simple, it contributes significantly to the usefulness of an SOA. The SOA by its nature is distributed; locking down line communications at the application-to-service and service-to-service client removes restrictions on the location of the service client or user.

Advantage: With this approach, you get noninvasive, flexible services that leverage needed parts of your legacy applications. As long as there is no reason to track client use from one query to the next, the services can stay stateless. Access is fast because the services can establish their underlying sessions to the mainframe application before they become available for use. The clients of the services can access mainframe assets without waiting for session creation. This is safe delivery for stateless use.

Disadvantage: Stateless application use implies that the mainframe or its applications don't directly know who their users are. Any authentication happens in the mid-tier infrastructure, or not at all. Although lines are locked down and the service might have authentication that points to Active Directory, for example, there could still be a possible security loophole; when legacy assets are unaware of their use, audit concerns can arise. This method requires a trust relationship with the mid-tier service clients and the infrastructure that governs them.

Appropriate use: When the application extends beyond the safe network zone, but the clients are trusted (either directly or through proxy), this approach works well. The assumption is that the services are visible and available only to those who have the right to use them. Even without explicit protection or tracking of service usage, the services are not expected to be able to perform harmful activities; their use is innocuous.

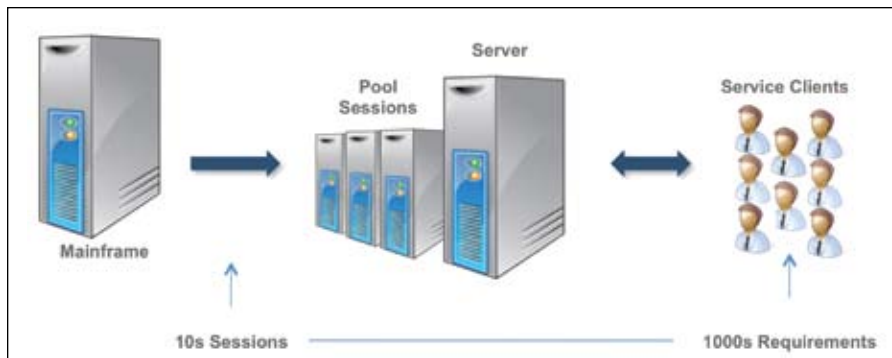
Verastream installations often contain services that aggregate non-proprietary data built statelessly. This situation allows the client to continuously query for updates, but it's not a danger because no updates to the legacy application are allowed. And because the

data are non proprietary, the need to match each client to each query is not of high importance. If you have complete control of business data and logic in your enterprise, you can implement stateless access and lock lines down as you see fit. If your usage extends beyond the "glass house," and you are opening mainframe applications to trusted session

pools or trusted external sources, this method might also be adequate.

3. Stateless with logging in place

Organizations needing more stringent security measures might consider adding a security log to methods 1 or 2 above. With this third approach you match service clients to service sessions for application access. Using Verastream Host Integrator, for example, you can create and store logs of the user identity for each use of a host application service. This helps in tracking host resource use and provides a good audit trail, without impacting some of the inherent performance benefits of stateless use. With services in established session pools, multiple users can call to get server-controlled dedicated sessions in a many-to one relationship.



In this many-to-one approach, service clients are matched to service sessions using logs stored on the mid-tier and controlled by service-wrapping technologies.

classified and business advertising system. Credentialed users interact with services that in turn interact with the designated mainframe application to initiate and pay for ads. Security is considered appropriate for the situation as the users are known, the lines are locked down with SSL, and all interactions are recorded at the service level.

The stateless model with logged service use works best for a true SOA, but still contains risk from a mainframe perspective. When you create wrapped services where use is controlled by the mid-tier, you need a trusted infrastructure in front of the mainframe. SOA governance, combined with service-use logging, helps reduce the risk. In this scenario, the mainframe itself isn't required to know who is gaining access, because the mid-tier is recording user credentials and actions.

Advantage: This method provides fast, flexible use of established stateless services, while letting you leverage shared host resources and sessions. Logging of stateless access provides an accounting of who invoked the services and how the services were used. In addition, staged (parked) resources allow performance and host usage benefits that are not available in less flexible environments.

Disadvantage: If SOA governance and mid-tier service logging are not trusted or reliable for authentication and resource usage, the mainframe guardians cannot always be sure who is gaining access.

Appropriate use: This method is appropriate for a wide-ranging set of scenarios, but relies on a mature mid-tier. Whenever data does not need to be directly controlled by the host application, mid-tier logging is an ideal way to accommodate secure use. Both government regulations and auditing procedures typically require a log analysis, so this is a good basic business practice for a variety of uses. However, for many vital functions, such as general ledger or other accounting operations, a proxy log from the mid-tier may not be considered adequate.

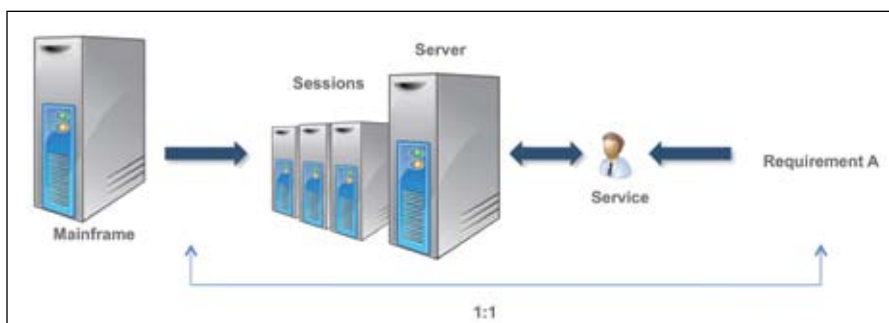
A major U.S. newspaper has successfully used Verastream resource logging to allow stateless interaction for a new SOA-based self-service

4. Stateful session-based services

In some IT settings, the mainframe application needs to directly govern its usage. When that's the case, you need to provide a service that directly identifies users and their level of security to the application. In other situations, like many customer-service applications, first-level access to customer data might be unique to the user's credentials on the host application; the data presented by the mainframe application is specific to their login ID.

For situations like these, services need a uniquely identified host session using the client's credentials for the duration of the interaction. To invoke those services, users must provide a specific set of data to have the service initiate a unique host session that they retain for their use. You can set it up to have the service pass the user identity and password – for both Resource Access Control Facility (RACF) and, if needed, the destination application used by the service operation.

Once the mainframe and the application verify those credentials, the service runs as normal using the authenticated session. When the user's series of service operations are complete, the service terminates the host session and waits for its next use. This approach provides the mainframe and application complete control over access and direct visibility into users throughout the duration of the service.



When the mainframe needs complete control over access, you can set up services that users invoke by providing a specific set of data, including their level of security.

Advantage: Stateful access lets you leverage anything in your enterprise with the same security you had prior to any SOA enablement. Because it uses the mainframe's existing security infrastructure to manage host sessions, you mitigate access risk.

Disadvantage: Although stateful connections can leverage the host resources directly, they consume a proportionally higher mainframe resource use for every service invocation. In a scenario where the mainframe may need to accommodate 1,000 simultaneous client requests, it would need to create and maintain 1,000 host sessions, each lasting the duration of the calling client's interactions. This is significantly different from stateless models that can handle similar workloads with just a few shared host sessions. Also, slow initial response times can result from the need to create unique session states – waiting for execution of the built-in security steps and instantiation of the host session for the service to use.

Appropriate use: This approach works best when the mainframe application needs direct control over its resource use or when sequential activities by the service are unique to the client's identity. For example, an IT enterprise on a college campus is successfully using this method for class registration. When a student needs to use the registration application, a security routine launches, the service asks for the user name and password, and then, using these credentials, creates a unique host session – one with a specific view of the student's data. The mainframe application requires a one-to-one match between user and action taken during the session. In essence, this approach makes the service client an extension of the host application.

5. High-speed stateful services

For applications that reside entirely within Customer Information Control System (CICS), there is a hybrid option that is worth exploring, as CICS accounts for the majority of mainframe applications today. With CICS, it is possible to create wrapped services that do not have host sessions behind them. They instead create a direct link to the CICS application, one that retains control over all application logic and flow.

Verastream Bridge Integrator allows this use. Its approach is made possible by leveraging the IBM Link3270 bridge for access. With this access method, direct mainframe authentication is possible (though not required); it can leverage authorization from

both RACF and the application itself providing the highest level tracking and security. Although these authentication routines are the same as in approach 4 above (with logging, usage, and visibility handled directly by the mainframe), you skip the session creation and associated authentication processes. Instead, you build your services to create a direct mainframe-authenticated link to the CICS application, from scratch, for each service operation – something the bridge access method can do with little to no cost in resource use or timing.

With this direct-access approach, when a client application (such as a web site) calls a service, mainframe security parameters can be passed as part of the service operation. Then the service creates a direct, authenticated link with the destination CICS application, performs the service interactions, disconnects, and terminates the link.



This one-to-one access approach lets a client application call a service, which can create a direct, authenticated link with the destination CICS application to perform the interactions.

Advantage: This method eliminates the costs of creating host sessions for stateful purposes that have a short life. You can use it for state-based or stateless services – with many of the same performance benefits as a stateless implementation. It can also remove any middle-tier component, so you can simplify the locking down of communications with SSL. Typically a stateful process would result in high overhead and slow instantiation, but with this hybrid access approach available to the service, even a stateful service is faster and less mainframe-intensive than a standard host session-based stateless service.

Disadvantage: There are two issues with using direct application link as the access method for services. First, the host application must be entirely within CICS. Although it can span CICS regions or even mainframes, all application components that the service interacts with directly need to be within CICS. The second issue is the requirement to add a third-party product into one or more of the CICS regions. Because this method uses a CICS host-based component, it can complicate the approval process for

implementation. (However, like the services using host session-based communications, it does not require any modification to the host applications themselves.)

Appropriate use: This approach has proven successful in performance-intensive applications like banking ATM and teller operations. It is especially useful for transactions that must return a rapid response. For example, when customers hit the keys on an ATM, they expect to get data back instantly; long response times are out of the question. In this situation, an ATM calling a service that harvests host information via the CICS 3270 Bridge can have high-speed responses with all the tracking available from a stateful service. It's the optimum solution for security; the mainframe has full insight into the user's identity and can track all interactions itself.

Finding the Right Solution

To find the correct solution for your IT environment, you need to explore the available access options while keeping security as your topmost consideration. To clarify the above discussion, here is a recap:

- If you are a system administrator working in a trusted IT environment, open/stateless access with no locking will be adequate.
- If you are extending mainframe access beyond the "glass house," you can have a secure deployment by simply locking your mainframe conversations down with SSL.
- If business-critical operations are involved, you're better off tracking service interaction with logging at the service level in place.
- If you need direct mainframe visibility into users and their actions, stateful access with specific host sessions will work for you.
- If your environment is CICS-dominated and you have access to the infrastructure, you can eliminate actual sessions and consider instead fully stateful sessionless-based mainframe access.

The most secure approach will emerge when you perform a thorough analysis of your business demands and your particular IT architecture; then match those with a solution that gives you no more – and no less – functionality than you really need.

Guidelines for Division of Labor

As you sort through access options 1 to 5 above, keep the skills of your technical staff in mind. Almost every IT enterprise has an inherent skill-set divide that leads to questions about how and where services are created,

the ownership of the resulting services, and – perhaps most important – who is responsible for keeping the services in line with current business needs.

This divide centers around the skills of the people who maintain legacy host applications vs. the skills of the people who work with mid-tier components. So before handing off services built from mainframe assets, be sure to have a plan for division of labor. That means IT specialists must parse the host application in such a way that mid-tier specialists aren't given services that are too granular for their needs. In addition to practical concerns, this could have security ramifications; there are few reasons for web developers to need direct access to general ledger information on the mainframe.

The bottom line: don't put change control of critical host business logic in the hands of mid-tier specialists who don't have the needed context to maintain it.

Instead, consider having COBOL specialists retain control of services that require host changes and keep the services up to date. Mid-tier control can cover the use of the services such as orchestration, security, policy needs, and other SOA-specific concerns.

When mid-tier specialists receive business objects generated at the right level, they'll know how to use them. And when ownership stays in the hands of the mainframe experts, you don't have to worry about keeping the services maintained.

Mainframe Assets: Too Valuable to Extend?

So let's revisit the question we began with: When optimizing mainframe applications for reuse with modern technologies, can you really gain flexibility without sacrificing security? The answer is yes. But let the buyer beware. The business operations controlled by your mainframe have never been more critical.

Safeguarding the Modern Mainframe

When dealing with sensitive mainframe data and logic, a noninvasive methodology is critical to your short- and long-term success. Changing valuable host code or associated business processes could result in catastrophic business losses. Any of the approaches discussed in this article can be done noninvasively.

Your present mainframe security is another factor. If you have a security system that is working for you – whether it's IBM RACF or Computer Associates ACF-2 or TopSecret – that's the one you should use for authentication in any access scenario. Bypassing your mainframe security is a practice that should never be taken lightly.

In addition to working with your current security programs, any application integration method should support federal regulations including the Federal Information Processing Standard. FIPS is a U.S. government guideline used to authenticate cryptographic modules. While FIPS support ensures compliance with government data-protection standards, it will also safeguard your private-sector business data at the same time.

Today's sluggish economy has resulted in tightened budgets for IT organizations, but businesses must continue to innovate and stay competitive. When it comes to mainframe application integration, this set of circumstances calls for cost-effective decisions that can deliver quick, risk-free results.

The best application integration approaches have options for both stateless and stateful access, letting you decide what would work best in your individual IT environment. In fact, modernizing your mainframe applications is not as hard as some IT professionals might assume. The key is to do it in a way that protects the integrity of your valuable mainframe assets.

Making it Easy with Verastream

If you're considering SOA enablement of your IT assets, but your legacy systems have been holding you back, you can look to the Attachmate Verastream product line. It provides application integration tools that quickly and noninvasively service-enable all types of mainframe and enterprise host functionality.

Unlike other legacy integration software, the Verastream product line includes solutions for options 1 to 5 in this white paper. You might be interested in these specific options:

- **Screen Integration**

Whether your environment is IBM zSeries (S/390), IBM iSeries (AS/400), UNIX, OpenVMS, or HP e3000, you can use **Verastream Host Integrator** to transform your legacy applications into reusable services for building composite applications.

For accessing applications via the screen interface, Verastream Host Integrator is the ideal tool. It lets you create services or components (e.g., COM, .NET, Java, or web services) that can be mixed, matched, and reused in today's SOAs.

- **Transaction-Level Integration**

With **Verastream Transaction Integrator** you can combine data and logic from existing programs developed for the IMS and CICS environments. Key features include on-host and client/server implementation options, native security support (including RACF, ACF2, and TopSecret), and web-based centralized management.

Verastream Transaction Integrator is the ideal tool for service-enabling ECI-compliant CICS applications in IT environments where you cannot install software on the mainframe. It directly accesses business logic through the COMMAREA via DPL.

- **CICS 3270 Bridge Integration**

Verastream Bridge Integrator is a native, mainframe-resident adapter that provides rapid, high-performance integration with any mid-tier .NET or J2EE application. CICS interaction is enabled through the IBM Link 3270 Bridge in any format required by the corresponding CICS application.

Verastream Bridge Integrator is the ideal tool for service-enabling CICS-based applications in IT environments where you can install software on your mainframe. It runs inside the CICS Transaction Server and leverages the power, security, and reliability of the mainframe.

Note: None of the Verastream integration solutions requires changes to your mainframe application code.

About Attachmate

Attachmate delivers advanced software for terminal emulation, application integration, and secure communications. Our NetIQ business provides solutions for automating IT processes and managing performance, security, and compliance of distributed IT. With our technologies, more than 65,000 businesses worldwide are putting their IT assets to work in new and meaningful ways. www.attachmate.com.



Corporate Headquarters
1500 Dexter Avenue North
Seattle, Washington 98109
TEL 206 217 7500
800 872 2829
FAX 206 217 7515

EMEA Headquarters
The Netherlands
TEL +31 172 50 55 55
FAX +31 172 50 55 51

Asia Pacific Headquarters
Australia
TEL +61 3 9825 2300
FAX +61 3 9825 2399

Latin America Headquarters
Mexico
TEL +52 55 9178 4970
FAX +52 55 5540 4886

WEB attachmate.com
E-MAIL info@attachmate.com

For regional office information, visit www.attachmate.com.