designedGov Pitch

Chris Morin <chris.morin2@gmail.com>

Table of Contents

1. Introduction

I want to create an application which will simplify how a government interacts with the persons and organizations that wish to interact with it. The application will be secure, robust and be built with expandability for unforeseen future needs. This project will specifically be built with the Canadian governments in mind but it will be very "government neutral" and should be easily adapted to other governments. From here on out, this application will be referred to as designedGov.

To make things a little less abstract, a possible service of designedGov could keep track of a person/organization's financial transactions and remove the need to "manually" file taxes. Another example would be a person having access to their entire medical history and medical service providers being able to access it instantaneously regardless of their geographical location. An important aspect of this application is that persons and organizations will be able to access almost all government services through a single online portal while only needing to keep track of a single set of account details (user name and password).

> I like writing software. I like trying to make things work better. In many ways, the end result is unimportant - it's really just the excuse for the whole experience.
>
> — Linus Torvald *TechCrunch interview*

2. Definitions

Government layer
: Official administrative organization which manages a certain layer of a society (layers being federal, municipal, …).

Government
: When used without the term layer or without specifying a certain layer, government refers to all the layers of government.

Organization
: A group of persons, public or private which need to interact with the government as a single entity.

GO (governmental organizations)
: The organizations that provide public services and owned by the state. These organizations are administered by the government.

NGO (non-governmental organizations)
: Same as GO but these are private organizations. This definition conflicts with the definition proposed by the United Nations as the UN definition excludes conventional for profit businesses. For this I may update the text if I come up with a better term.

Person(s)
: Individual who needs to interact with a government. This can be citizens, exchange students, tourists, … . Most of the time, these people are located in or have the intention of being located in the country of the government they wish to interact with. Otherwise, they would most likely interact with the government through an organization.

User
: person or organization wishing to interact with a part of the government.

Application
: A collection of software which works together to provide a service.

Account
: representation of a user within an application. The application can remember user information and often provides a secure way of ensuring only authorized interaction with an account occurs.

3. Justification of Need

> **Note**
>
> Feel free to skip this if you're already exited.

I'll spare the spiel about how computing and connectivity is becoming more ubiquitous and about how we should take advantage of it to better organize our public services because I find it glaringly obvious. I'll focus on why I felt the need to create this project instead of waiting for the inevitability of the government creating their own solution.

Already, the government is implementing and rolling out digital services or digital front ends to their services. It would therefore be fair to ask why I feel the need to create this application. My answer is that the way government services evolve is very bad. They build upon previous iterations and all evolve mostly independently (and dependently in some bad ways) so that at the end of it you have a bunch of scattered systems where most of the design decisions weren't made to increase usability but to fit into the existing one. This is a pain for the end user as they need to not only learn how to use all

these systems but it also makes them need to keep track of many different account details. Although this might seem like a mere inconvenience (which is enough for me to justify this system, but not for others), the larger problem is security.

Organizations use user authentication which was never intended for the used purpose. This is called functionality creep and is a costly problem. An example is the Canadian social insurance number (SIN): """"The SIN was created in 1964 to serve as a client account number in the administration of the Canada Pension Plan and Canada's varied employment insurance programs. In 1967, Revenue Canada (now the Canada Revenue Agency) started using the SIN for tax reporting purposes."""" The SIN has become the de facto national identification number, a purpose for which it was never meant to be used. The security flaws are glaringly obvious: the SIN must be passed around as plain text and with it someone can steal a person's identity. This has become such a problem that the Canadian government has recently decided to phase out physical SIN cards because of rampant identity theft. Instead of developing a new system as I am attempting, their solution is to rely on stricter privacy laws.

Different GOs have created their own account details such as the student "Permanent Code" or the driver's license number. Problems with these are that once again they are inconvenient and aren't made with security in mind. I'll talk more about it later, but I feel that the people designing these systems don't know about modern cryptographic technology which should be a core tool in any system of this type (though I might be underestimating them).

Even if they were made with security and even cryptographic tools in mind though, a secure system requires the user has knowledge of how the security mechanisms work and what can be trusted. A system can be cryptographically secure but if a criminal impersonating a government official can call an unknowing user and convince them to divulge their account details, the system isn't secure. Each user having a single account for all services makes it easier for them to learn the knowledge needed to keep their information secure. I'll talk more about security education later.

The government has also shown us time and time again that they can't be relied upon to efficiently create a good application at a reasonable cost. A recent example that comes to mind is the Canadian gun registry. It's an extremely simply system and the development and maintenance costs should have been negligible. As an outsider, I know I don't have the full story, but no programmer I've talked to understands how a project so simple could cost so much.

I could write a book on the shortfalls of our government's current digital infrastructure and why I chose to create this application, but I know when to stop beating a dead horse; I'll finish on an idealistic note instead. I believe an application that will be this pervasive and important to our society shouldn't be created by a company contracted by the government but should be an open source project developed and maintained by volunteers. The government will of course take over the reins when (if) they start taking it seriously, but even then I feel it's too important to be developed behind closed doors and should remain open to community contributions and input. An application of the people, by the people, for the people.

4. Guiding Principles

I was unsure whether to merge this section and the next because they basically come down to the same question: If we were to through away our preconceived notions of how a government should provide services and envision the ideal application to perform this task, what would this application look like? I'm not talking about aesthetics here, but about fundamental questions like what scope of services

should the application cover? What amount of technical knowledge should we expect from users? What kind of privacy should we guarantee? I decided to put the principles I envision that would (I think) create the least controversy in this section and then discuss issues we as a people will have to decide in the next.

Simple. Consistent. Secure.

4.1. Principles:

-Users will be given a single unique set of account details along with a single access point to most government services. Since we are designing the system as a whole, we have this option and there is little reason for us not to take it. One could argue that having a single entry point into a person's entire "identity" is too vulnerable. That is fair, but I feel that implementing heuristic identity theft detection and teaching users about basic digital security would render it more secure than a multiple account system. Also, because of the way our system works, having multiple account systems just provides multiple points of vulnerability to someone's entire "identity" because they can use access to one account to gain access to others.

-The application stack will be free and open source. I touched on this before. Considering the importance of this application, it would be safest to develop it in the public eye. This protects against both government incompetence and corruption (I'm not claiming they are incompetent or corrupt, but history has taught us that we must be ready for when they are).

-The best security system is a computationally secure one with educated users. Anything else isn't acceptable. This might be the hardest principle to swallow but I believe it to be the most important. The application will be computationally secure, but users will need to take the time to learn about security. I don't think this will be too difficult for them. Even at that though, an unknowledgeable person would be safer using designedGov over Canada's current systems. Even now, most people (even policy makers!) don't understand how modern security (i.e. cryptography) works while they still use these systems, like online banking or credit cards, on a daily basis. Instead of relying on users being educated, these systems use deterrents like threats of legal action against criminals. These deterrents should remain in place but evidence suggests they aren't enough. I would suggest a simple online security course. I don't think this would take more than a day for even the least tech savvy of users. A good general computing course should be part of our high school curriculum too. I'm not talking about how to use MS Office, I'm talking about basic programming, networks and how to stay safe.

-We shouldn't blindly trust the government. A centralized system like this, while making us safer from crimes such as identity theft can make a country vulnerable to a corrupt government (when you think of it though, our current systems aren't really any more secure from government corruption than the worst digital system). A possible worry would be the government being able to "delete" politically opposing citizens while individuals have no recourse. They could even create fake identities and have those identities vote for them. Some of the challenges are more difficult than others, but a great cryptographic tool at our disposal is called a digital signature. This tool allows a government to virtually sign some data like a person signs a paper document. Only the government can create this signature and it can be validated by anybody. A cautious user could then have the government sign his basic account information (Name, address, citizenship status, …) daily and store this information on a USB thumb drive and bury it in his garden. It would be incontestable proof that at that time his account was in the state he claimed it was and the government would have to explain any modifications such as change of citizenship status or even deletion of an individual.

-Mobile computing will become more ubiquitous and we should design for that. Already, most people have mobile computing and connectivity capabilities and ten years from now, all but the most cloistered will. We should assume that most users will have the ability to access this application anywhere and so should create service interfaces for them.

-Authentication of user identity should be a government service. A user should have the ability to guarantee his identity and this will then be authenticated by the trusted government service. This service has many applications such as preventing a criminal from opening a bank account under a false identity. This is part to the recurring "consistency and simplicity principle". It will simplify and secure many interactions across the spectrum.

-A user should be able to securely give account permissions to other users This builds on top of the previous principles. This consistent and freely available permission service will cut a lot of red tape. Imagine buying a car at a dealer, and at that same point of transaction having it automatically registered with the vehicle registration authority, having the plate mailed to you along with all the relevant fees being processed. Staying with the car theme, even selling a used car to another user could be automatically completed. A prospective buyer could pull the vehicle's history (repairs, accidents, …) with the seller's permission to make sure the seller is honest. This reduction in red tape should in turn bring the costs of delivering the services down.

-The application should consist of a plug-in framework. Having a plug-in framework on top of the authentication service will allow different GOs to develop their own respective plug-ins while maintaining consistency across them. This will also allow GOs to add (and remove) services to the application as they are created (dissolved) and as they grow (shrink). I'm not yet sure if NGOs will be allowed to have their own plug-ins but it's a possibility I'm exploring.

5. What we the people will need to decide

I have my own vision for this application, but I acknowledge that some aspects might not line up with the views of the general population. These possible points of contention are related to how much we should expect users to know about the system, how dependent should we be on this application and what kind of privacy policy will we have.

Ideally, all users would be well versed in cryptography and the specifics of the system. It is inevitable that some unknowing users will make mistakes that compromise their identity and even their financial well-being. We can always restrict the freedoms of users to prevent them from "shooting themselves in the foot". My personal view is that although there might be a few hitches at the beginning, designedGov will be part of everyday life and people will discuss and share amongst themselves best practices. Also, because of how an identity is centralized, a stolen identity would be very easily recovered and could be as easy as a password change. This will have to be discussed and debated.

Next is how dependent we should be on it. Having the view that this application is more secure than our current systems, I believe that this should be the default form of government interaction. That being said, how much trust we should put in the safety of the system will be contentious. Would people be comfortable voting through this system? When someone authenticates themselves after completing a large transaction, should more traditional authentication methods be used (e.g. photo ID). The convenience provided by the system would follow "increasing returns" with additional trust as any human interaction would be a bottleneck in an autonomous system. The trade-off is not unlike that of the previous paragraph.

5.1. Privacy

Most importantly though is the issue of privacy. Unlike the other two points, I won't voice my opinion on this because any opinions will cause a chunk of the population to immediately discredit the whole project. Also, I'm not convinced about the right way the application should handle privacy. Before anyone shoots the project down because of privacy concerns though, ask yourself whether the current system is really anymore private. More often than not you'll find it's not.

With cryptography, we can guarantee that even though the government authenticates a user and stores his data, it can't actually access any of it. Even if someone opened up the central server and directly read the hard drive, they would find an encoded message only a password held by the user could decode. This is much more privacy than we have now. The problem with this is two-fold: if a user forgets their account details nothing can be done to recover their data. Secondly, it could be beneficial for a government to have access to your data. In fact the whole point of designedGov is just that, liberating data to sinplify interactions.

Perhaps a more practical solution would be to have each GO store user data associated with them on their servers. While they can access it at will, other GOs need permission. The governmental organization in charge of health could store and freely access a person's medical records (someone can't really give a password if they're unconscious in an ambulance) while say the organization in charge of tax collection would need a user's permission if they wanted to confirm you had a certain disability for tax benefits. Every time an account is accessed this information could be logged and this log could be made available to the user, but there is no way of guaranteeing an organization can't access their own data on a user without his permission in this system.

Another point of contention is whether certain authorities should have access to a user's data and those accesses not be made known to the user (not logged). How much information should police have access to? Would a judge have the authority to pull information from any organization at the request of detectives for an investigation? Giving police the unrestricted ability to secretly read data would greatly ease investigations but would have many worried about them having too much power.

Also, the government could provide a service that allows unrestricted or restricted access to anonymous data. This could lead to very interesting finds. For example, health records could be data mined and novel medical discoveries could be made like an unknown cause of a disease. Studies using traditional methods can take years and millions of dollars but this could be done in a few minutes and be virtually costless.

I plan on implementing the "practical" solution, not because of its merits but because it's closest to the system we have now and so people will have a harder time rejecting it on that front.

6. Project

Finally we get to the good stuff. This is really just a top level view of how the different application pieces come together so many details have been omitted. The layers making up this application from top to bottom are as follows:

- Web interface (optional)
- Organization applications

- Communication protocol
- Central identity server
- User definition

I'll describe them from bottom to top.

6.1. User definition

Each user, be they an organization or individual, will be given a unique number. The details of this number such as length are to be determined at a later time. Babies will be given numbers when born (under control of parents until a certain age), immigrants when they land and NGOs when they register themselves. Each user will also have their own public-private key pair. For those unfamiliar with cryptography, this pair allows a user to authenticate his identity to others.

Note

Notice how no password was mentioned. In an ideal world, no password would be needed as a user could always use their public-private key pair to authenticate their identity. In the real world though, we have to worry about users loosing and worse having their private key stolen. For this reason, it might be best for each user to have a password that is solely used for modifying their public key held by the central identity server (more about this below).

6.2. Central identity server

The central identity server is the authoritative server on who is who. It will simply contain a list of all existing user numbers and their associated public keys. Any user will be able to query it for the public key associated with a user number. The server might also contain additional user information such as what type of user it is (citizen, government organization, for profit company, …) and their IP address if applicable. The central identity server will have it's own public-private key pair which is the only thing needed for a user to be able to authenticate the identities of other users. When user A claims to be the owner of a certain user identification number, a skeptical user B can query the central server for the public key of the number user A gave him. Using cryptography, user A and B can now authenticate user A's identity. The way this is done is quite technical but will be simple for the end user to use.

Note

Although I said that the server could authenticate a user's identity, the astute reader will have noticed that it only matches a user number and a public key. All useful identity authentication (such as: "Is this woman standing at my door a detective as she claims") will then be done by the organization applications, but they will all need this initial exchange to occur.

6.3. Communication protocol

This consists of a protocol for allowing users (individuals and organizations) to interact with each other. This protocol will have to be very flexible as it is likely unforeseen functionality will be needed in the future. Also, here is defined how the permission system will work; this system ensures all data

access is authorized. The user will have large control of these permissions and understanding this system will be paramount to the security of their account.

6.4. Organization application

GOs will be responsible for creating and hosting their own applications. They can develop them as they see fit, but their applications will need conform to the communication protocol for any external communication. If they wish to use legacy systems, they will need to develop an intermediary application which translates information between the GO’s legacy system and the communication protocol.

6.5. Web interface

On top of the most popular organizational applications, a web portal will be created which allows users to easily access these applications through their a single portal. This interface will also allow them to control who has permissions to their account. This will be a reference implementation, and I hope other interfaces will be created on different platforms. These interfaces can compete to be the most usable and to support the most organization applications. Interfaces can be created to target specific niches like an interface for small businesses or an interface for speakers of a foreign language.

7. Foreseen Challenges

This project isn’t challenging from a technical point of view. That being said, I’m just one programmer and in order to get a "toy" project up and running which will be used to demonstrate what will be possible with designedGov, I’ll have to learn about many different technologies. Also, acquiring the necessary design skills to make this toy "pretty" enough will be tricky.

The real challenge though, will be convincing the Canadian government and the people of Canada that this is a good idea. Once that’s done, the challenges will be teaching Canadians how to use it and getting all the GOs to develop their plug-ins and cooperate with each other.

8. Hopes for the Future

I hope designedGov becomes adopted by the Canada while still following most of the principles laid out here. I also hope this will greatly reduce the red tape and administrative costs associated with our government. If this happens, Canada will be a role model when it comes to integrating technology into society. If other countries implement similar systems, it would then be possible to develop a global system which would allow users from different countries to interact with each other.