Blindsided by Security

The Reality of Web Security for the Visually Impaired

Britta Offergeld (Royal New Zealand Foundation of the Blind)
Laura Bell (Lateral Security)

Released for OWASP New Zealand Day - August 2012

Abstract

As web developers we try to design systems that can protect as well as provide for our clients. As security consultants, we develop guidelines and frameworks that people can use to decide if a web application is trustworthy and secure. Even the least technical home users are becoming more confident in spotting suspicious behaviour online.

Unfortunately, for the visually impaired, it's not that simple. In a world where visual clues are not enough and where additional technologies such as screen readers are business as usual – web security is a very different matter.

In this whitepaper, Lateral Security and The Royal New Zealand Foundation of the Blind examine the guidance and security best practice commonly in use for web applications today and how effective they are for those with visual impairments. In addition, a series of improvements and solutions are outlined.

Introduction

Web application security is a fast evolving area of information security. Internationally, security researchers are working to push the technologies employed for online commerce and communications in search of exploits and vulnerabilities. While some of these researchers do so in the hopes of helping to secure these services for these users, an increasing number are using these vulnerabilities for criminal or financial gain.

As fast as the technology evolves, our web application developers are expected to adapt. Constantly seeking new techniques and paradigms to address the security concerns of business. To complicate matters further, these advances in security need to be delivered in a way that not only makes users safe, but makes web applications intuitive and user friendly.

Finding the balance between security and usability is challenging in normal circumstance, but when designing with the blind or visually impaired in mind the decision we make as developers can render an innovative application unusable.

A good web application developer needs to not only keep up to date with the latest best practice in security, but also adapt their technical choices to suit an audience that includes those of us reliant on screen readers and adaptive technologies to operate online.

New Zealand is home to over eleven thousand blind or visually impaired internet users. People who are embracing technology to allow them to make use of the powerful communication and business opportunities offered online.

In this whitepaper we examine five elements of modern web application design, how they are commonly implemented and the consequences of these implementation choices for the visually impaired.

In addition, we will outline simple changes to these designs and implementations that would improve the usability and security for this demographic without compromising the overall appeal and usability of the site.

Assistive Technologies for the Blind

Whether an individual was born blind or whether their visual impairment is as a result of an illness or event later in life, the range of technologies and techniques employed to operate in an online environment are the same.

These technologies centre on products and devices that can interpret the information on the screen and present it in a more suitable format. Traditionally this is audio or braille presentation.

The primary difference between a visually impaired and sighted internet user concerns the use of input devices. Unlike the average internet user, a blind or visually impaired user relies on a keyboard, voice or physical gestures to interact with the display. Mouse usage within this group is very low and few technologies exist that can tolerate the tracking of co-ordinates on a screen to the level of precision required to assist with the use of a mouse.

This change in input device is one of the primary considerations overlooked by web application developers.

In Association with

ROYAL NEW ZEALAND FOUNDATION OF THE BLIND
TE TUĀPĀPĀ O TE HUNGA KĀPŌ O AOTEAROA

LATERAL SECURITY
INFORMATION SECURITY SPECIALISTS

In addition to differences input devices, the use of screen reader technology to interpret the information displayed in the web browser can dramatically change the experience offered by a web application. Imagine an application that relied only on the parsing of its DOM and written content to appeal to the end user. An application unable to use font effects and graphics to draw the eye or focus the user's attention presents a much greater challenge than most developers and designers anticipate.

1. Multi-Factor Authentication

The principle of multi-factor authentication is not a new one. In the physical security field, multi factor authentication is commonly used for high value items such as security boxes, bank accounts and safes. For example a commercial safe will require a physical key as well as an access code to open.

The basic premise behind this authentication system is that the user is required to provide more than just a secret phrase or password to identify themselves and authenticate. Multi- Factor combines this "known value" with factors such as physical tokens and personal details. This is sometimes referred to as "something you have, something you are and something you know".

Multi-factor authentication is now increasingly being employed by banks and some email providers.

Modern online two factor authentication systems are divided in to two main groups, each providing their own challenges for visually impaired users. These groups are physical tokens/devices and software challenges.

Multifactor Authentication: Physical Tokens

Physical tokens in particular pose a great challenge to the visually impaired. Popular with not only banks but also businesses providing Virtual Private Network (VPN) solutions. These tokens, manufactured and developed by third party providers such as RSA and VeriSign, are integrated with web applications through the use of externally hosted web services or appliances.

In a standard implementation, a user is prompted to provide a username and password in addition to a random character sequence generated by a physical device. These sequences are often generated automatically and update at short intervals (typically less than 2 minutes). This information is then sent to an authentication server which compares this random sequence to the expected value for that token at this point in time.

Alternative configurations for physical tokens include the provision of printed grids of characters. Web applications then prompt for a series of characters from the grid and compare these values to those stored in the authentication system. This configuration is substantially cheaper than the physical computerised tokens but provide a much smaller set of random combinations of values.

The small digital displays employed in such tokens can be impossible to use for those reliant on adaptive technologies. The combination of small size with a rapidly updating display means that OCR and reading technologies can rarely process the information in time. Often the random sequence has expired before the software or reader has successfully identified the character string.

Even physical grid card tokens rarely include braille representations of their character sets, rendering them unusable without assistance.

ASB FastNet Business banking RSA Token

The ASB banking RSA Token displays 6 digits, which change every 60 seconds. The Business banking user needs to enter these digits to complete the web based logging in process.

A visually impaired user, who can't read the digits, is not able to know what those digits are, in order to put them into the ASB website.

Rabobank Digipass Device

The Rabobank Digipass device requires the user to enter a Pin, and then the device displays a number for the user to enter into the website to complete the login process.

A visually impaired user would be able to enter the Pin into the Digipass device, however they would then not able to read the digits displayed on the device after Pin entry. So they are not able to complete the login process online.

The current Solution, for the user to deal with these Multifactor Authentication Physical Devices, is for them to get someone else to help them to log in to their account.

Multifactor Authentication: Software Challenges

Kiwibank Keepsafe Challenge Example

The Kiwibank Keepsafe Challenge is an optional multifactor authentication web application, which gets presented to the user after they login with their internet banking number and password.

The web application chooses one of 3 security question and answer pairs to present to the user. The security question is presented as plain text, and the security answer is presented as a series of empty green tiles, the exception being the 2 or more required letter tiles,

which are displayed as white. The first required letter tile has an arrow pointing to it on screen, to indicate that this is the first required letter that needs to be input.

By requiring the user to use a mouse to select letters and randomising the gaps and answers from a selection of three, Kiwibank is attempting to prevent user details being collected by malicious software such as key loggers. The theory is that no keyboard input and limited information on screen will reduce the ability of an attacker to steal and reuse credentials.

These are the issues a visually impaired users is likely to encounter when trying to use this web application on a PC:

- The position of the any of the required letters in the word is unknown.
- Instructions on how to go about entering required letters is unknown.
- Once the task is complete, the Proceed button pops up silently, so a visually impaired user does not know that they have completed the web application task successfully.
- There is no information on how to go back and check required letter entries, if the user thinks they have made a mistake.

2. Visual Security Clues

Browsers have a means of visually signalling to a user that there may be an issue with a website they are visiting. These visual clues are called web security indicators and advice around checking these, usually contain references to a "green browser address bar" or a "padlock icon".

Research into New Zealand banks security advice on the web, found that Bank of New Zealand was one of the only banks to make a reference to checking for the s in https, as an alternative way to check a websites authenticity. (Reference on BNZ website: "If the address bar changes from http to https this also indicates the connection is more secure" [1])

Extended Validation Certificates

Some banks make references to checking Extended Validation Security Certificates.

- Kiwibank : “Confirm our Extended Validation Certificate by clicking on the padlock to the right of the address bar”
- BNZ : “We advise checking the certificate when using publicly accessible computers in internet kiosks or utilising free wireless internet access, before entering your secret access credentials"

Checking Extended Validation Security Certificates Using A Screen Reader

Looking into how a screen reader user would check these EVCs, it was discovered that this could most easily be achieved in Firefox 14 in Linux or Windows.

The procedure for checking EVCs involves navigating to the information either side of the browser address bar, depending on what browser is used.

Browser	Version	Result
Firefox	14	EVC reading is possible and simple
Internet Explorer	9	EVC reading is possible line by line
Chrome	21	EVC reading not possible
Safari	iPad and Mac	EVC reading not possible

3. CAPTCHA

CAPTCHAs are a means of sorting whether a human is interacting with a website, or whether automated software is trying to interact with it instead.

They were introduced to combat the rise of automated scripts and spiders generated to fill out and submit forms online. Such bots and spiders not only cause a nuisance for web developers but can facilitate fraud and other malicious activities.

In a CAPTCHA protected system, the user is generally required to complete a task, the most common of which involves reading obscured numbers and letters, or listening to obscured audio containing words or numbers. The theory is, that automated software cannot easily complete these tasks.

In May 2012, WebAim did a survey of screen reader users, in which 90.6 % of users indicated that they find CAPTCHAs difficult. [2]

Despite persistent claims by CAPTCHA vendors that their software cannot be cracked or automated, a lot of research and work is being undertaken in this area, with varying degrees of success. Unfortunately, as vendors strive to improve their CAPTCHA solutions, they invariably increase the overall complexity and have a negative impact on the overall usability of the product.

An example of this from June 2012, saw Google respond to the Stiltwalker audio reCAPTCHA hack [3] by changing the audio reCAPTCHA background noise to be more indistinguishable from the input words, and

they require a user to listen for, and enter more words in total.

In the process they made it a lot harder for visually impaired users, and even those who previously had little difficulty solving audio reCAPTCHA, were caught unawares by this change and found solving the new audio reCAPTCHA tricky.

Some research into alternative suggestions for CAPTCHAs, turned up some interesting material.

The Imperva, Hacker Intelligence Initiative, Monthly Trend Report #11, A CAPTCHA in the Rye [4], suggests, amongst other things, that users could play mini games instead of solving CAPTCHAs. They also suggest using anti-automation solutions to bolster CAPTCHA defences with traffic-based automation detection, behavioural analysis, content analysis and blacklists.

A website promoting alternative logic based text CAPTCHAs, also had some suggestions around alternative means of checking a user's "human versus machine" status. [5]

Sadly however, some websites still do not provide an audio alternative to the CAPTCHA they are presenting. This omission often prevents the CAPTCHA from being completed by a visually impaired user.

Examples of these, which have directly affected visually impaired users in New Zealand are:

- Travel Sim [6], sold to a visually impaired person in Wellington by a travel agent. Activating the sim card involved setting up an account, which had a visual only CAPTCHA.
- Avast antivirus on Ubuntu 12.04 Linux, the 1 year registration has a visual only CAPTCHA. (Windows AVAST registration does not have such a CAPTCHA)
- Wellington Hospital asks some people to register their Medical test kits [7]. The website has a visual only CAPTCHA.
- Commenting on a US electronics blog site [8]. The Resistor image based CAPTCHA requires the user to drag a slider with a mouse, visually matching the slider band colour to the resistor band colour.

Some websites do provide an audio alternative CAPTCHA. A lot of New Zealand websites use reCAPTCHA, when they offer an audio alternative. The examples listed are representative of the types of activities that require a user to complete a CAPTCHA online in New Zealand.

Examples are:

- Parliament make a submission website [9]. To make a submission via the website there is a text and audio reCAPTCHA.
- Consumer Affairs report a scam webpage [10]. To report a scam via the website there is a text and audio reCAPTCHA.
- Air New Zealand bank transfer payment for flights. A text or audio CAPTCHA is required before paying for flights, even when the user is logged in. There is no CAPTCHA required for a credit card payment for flights.

Where a website employs reCAPTCHA, an effective and popular text reCAPTCHA solving tool, amongst the tech savvy visually impaired community in New Zealand, is a Firefox plugin called Web Visum [11]. Web Visum allows users to solve a maximum of 10 CAPTCHAs a day. Web Visum's developers started the project to address the wide spread text CAPTCHA only use, on the internet. Web Visum puts the solved CAPTCHA into the user's clipboard, ready for pasting into the CAPTCHA text entry edit box.

Web Visum is unable to solve all text based CAPTCHAs, but text reCAPTCHA is, more often than not correctly solved.

4. Instructions and Error Messages

Consumer Web Security advice

There are some great consumer security advice websites in New Zealand. These are excellent resources to direct visually impaired users to, however there are a couple of, simple to resolve issues that are preventing visually impaired users from getting all the information they need.

Security Central website

The Security Central [12] website has consumer advice arranged into topics, which can be easily accessed via a mouse driven drop down menu. Although it is possible to get to the main topic websites, not all the information contained in the menus is presented on those main topic websites, and therefore are inadvertently "hidden" from a visually impaired user.

An example is the Home Internet Users Topic. Issues:

- A visually impaired user is able to access the link to the Home Internet Users webpage, but the subsection on 'Smartphone and mobile internet' information is then not presented on the Home Internet user's webpage.
- If the visually impaired user knew that the Home Internet Users link was actually a menu, they would be able to use the screen reader 'On Mouse Over' emulation command, to drop the menu down (this is something an expert screen reader user

would be able to do, if they knew it was a 'onmouseover' menu)

The solutions to this issue are simple.

Solution 1: The mouse only event handlers should have an equivalent keyboard event handler. That way, when a screen reader user Tabs to the menu link, the menu drops down automatically, providing access to the submenu items.

Solution 2: Use ARIA roles, states and properties to indicate to the user that they are on a drop down menu, and then manage keyboard interaction and focus as the user chooses to explore the menu, or not.

The source code for this solution, which simply alerts the screen reader user they are on a menu, can be downloaded from the following location:

https://github.com/britta-nz/blindsided-by-security/tree/master/securitycentral%20org%20nz%20website

NetSafe Flash Website

The widely advertised NetSafe NetBasics [13] website, contains a wealth of information about staying safe online. The heavy use of Flash within this site however renders it unusable by a screen reader user. The exact same information is available in text only format on the core NetSafe site [14] which a screen reader user is able to read, but is unlikely to locate easily.

The simple solution to this, is for the NetSafe NetBasics Flash website to contain a link called "NetBasics Text only" [14] (or similar), which links to the excellent screen reader accessible information on the NetSafe website.

Staying Safe - PC Security - How To

During the first half of 2012, a noticeable increase in the number of infected computers and computers with expired antivirus, was discovered amongst visually impaired Internet users, in New Zealand's Lower North Island.

The issue had a lot to do with the silent notifications, popular antivirus software uses to notify users to renew license registrations. The free version of Avast, which is popular amongst visually impaired users in New Zealand, was found to be expired, or in some cases, was an entire version out of date, on some user's computers. The notifications were not being read out by the users screen reader, and so the user was not aware of the issue and never did anything about it. A similar issue was found on a Windows Vista computer, whose Microsoft Essentials software was telling the user to upgrade versions with a silent popup.

As application developers, there is a responsibility to ensure that all error and warning messages presented to the user meet the following:

- Use plain, unambiguous and easy to understand language
- Be presented clearly in a way that can be seen and interpreted by all users
- Only present necessary messages, verbose or overly frequent messages confuse users and encourage users to accept without reading.

Whilst a solution should be implemented by antivirus developers, the needs of New Zealand blind users require a more immediate solution.

To help improve this issue in the short term, the Foundation of the Blind aim to provide resources to vision impaired Internet users, on how to check their computers security status, and making them aware of the need to set reminders to check the status of their antivirus software, if the software has an expiry date.

Ensuring new screen reader users are able to independently manage their computers security, is particularly important, because on average, 1200 people a year register as new members of the Foundation of the Blind, in New Zealand. The proportion of new members who use, or have used technology is also increasing. Making the transition from formerly using a piece of technology with vision, to now using it with assistive technology, can leave a gap in the users knowledge, on how to check their technologies security status. This gap needs to be closed and the Foundation of the Blind is actively looking at and addressing this area.

5. Security Design Considerations

Visually impaired technology users, like a lot of sighted technology users do not get involved in making decisions around what security applications are purchased, installed and used in their workplaces. An IT department with the requisite knowledge is often charged with making those security product decisions. This is accepted practice and this procedure is not something that, in any way, needs to be changed.

Some unintended consequences can occur however, when security products are purchased, installed and rolled out to visually impaired users, without those products having been tested for usability prior to rollout.

An example is a VPN application which was recently deployed onto new computers at the Foundation of the Blind in early 2012. This software was intended to allow staff to connect to central systems while out of the office or travelling.

Issues began to arise with the system when it was first used remotely by a visually impaired manager trying to access files from core RNZFB systems.

The VPN application installed on the users system required an icon to be double clicked, and then a username and password had to be entered to connect.

As a screen reader user, the manager in question was reliant on integration between the VPN client and the screen reader to create a connection. To the frustration of the user however, the icon and its surrounding application interface had been implemented in such a way that the icon could not be accessed in any traditional alternative methods. This included keyboard access, direct object access and mouse emulation. This rendered the tool unusable and unfit for purpose.

Even though the VPN usability issue may seem trivial, and the solution simple, not being able to access email and files whilst on a business trip, is not a trivial situation for any manager to experience.

Whether new systems are the result of application developer effort or the selection of existing commercial solutions, testing is essential. This is especially true when screen reader users are part of the workforce.

Balancing Security And Usability

One of the biggest challenges facing the developers of Internet Browsers is finding the balance between providing a powerful user experience whilst protecting the host operating system from malicious activity.

Google's Chrome browser changed the way in which Internet Browsers implemented this protection by effectively segregating aspects of the system. The main browser UI runs in a separate Browser Process, to the Webpages, which are run and rendered in Renderer Processes.

As a result of this change, the Chrome browser was not able to be used with a screen reader users own screen reader, until Chrome 17

The Chrome Renderer processes contain the webpage DOM and accessibility information. These processes cannot interact directly with the user's operating system. This means Renderer processes can't send or receive events to, and from the operating systems accessibility API.

As a result, the screen reader could not inform the user of any webpage UI components in Chrome.

No accessibility API information, essentially means 'no webpage content'.

Web Components and Usability Issues

Web components are being used by web developers in order to take advantage of the functional encapsulation they offer within the DOM. Web components insert HTML into the Shadow DOM, not the regular HTML DOM. The Shadow DOM is the part of the mechanism that allows for this functional encapsulation within the DOM. Where events cross the shadow DOM boundaries, the event's information about the target of the event is adjusted to maintain upper boundary encapsulation. In other words, the details of where the events originate etc., are kept "hidden".

Key points to remember about Screen Readers and how they interact with the DOM of a webpage:

- Screen readers are able to access items that reference each other within the shadow DOM.
- Screen readers are able to access items that reference each other within the regular DOM.
- If visual components reference each other across the shadow boundary; due to event retargeting, the screen reader accessible information remains null.

No solution to this potential issue has been found. It is unknown how web developer's implementations of web components will affect screen reader users in New Zealand.

Proposed Solutions

Multifactor Authentication Solutions

Solution 1: Kiwibank Keepsafe Challenge Web application

The Kiwibank Keepsafe Challenge Web Security Application can been made usable for a visually impaired user.

By presenting the user with additional information, they can be enabled to complete the task, provide required letters of their security answer.

In fact, the web page does not have to be changed visually to accommodate a visually impaired user, all the additional information can be included in metadata and none visual fields.

Please note: The coded example shows the help text visually. The help text does not have to be displayed to a screen reader user in this way, and the visual component of this example is merely for the benefit of sighted developers.

- When the user logins in with their account number and password, the security question is presented, along with additional information on what position the first required letter is and how to go about entering it.

- When the first required letter is entered, the user is presented with information on what position the second required letter is in the security answer, and how to go about entering it.
- When the user has completed the task of entering all required letters, the screen reader user is told to navigate to the Proceed button to complete logging in.
- The user is given instructions on how to check for and correct mistakes.

The above solution was implemented via the addition of normal HTML tags and ARIA attributes [18] to the webpage HTML. Because there was no direct access to the Web Security App HTML, a Greasemonkey [19] script was used to layer the required tags and ARIA attributes into the HTML, on one of the authors PCs.

The Kiwibank Keepsafe Challenge Web application Solution implemented

This script has not, and will not, be installed on any visually impaired users PC and live testing would need to be conducted in conjunction with Kiwibank.

For the purposes of this whitepaper, a limited prototype was developed to work across the following screen readers: JAWS 13, NVDA 2012.2.1 and ORCA 3.4.2. (Voiceover for Mac was not tested. This requires installation of the nightly build, not the stable release Firefox 14.)

Like any cross platform implementation each screen reader provided its own challenges and some ARIA features were discovered to be supported slightly differently in the screen readers (e.g. live region reporting). This was not deemed to have a significant effect on the solutions overall effectiveness.

The source code for this solution can be downloaded from the following location:

https://github.com/britta-nz/blindsided-by-security/tree/master/Kiwibank%20Keepsafe%20Challenge

Solution 2: Multifactor Authentication: Hardware based Devices

To read the digits on the ASB RSA hardware device, a cell phone OCR app was trialled with one visually impaired user.

The method worked but 2 important factors made this solution an unreliable method for consistently valid logins.

The first factor was the Time factor. 60 seconds is enough time to complete the OCR procedure and for the user to login in theory, but in the user trial, the visually impaired person was not aware when the digits on the device changed, so they were unable to tell whether they had enough time left to complete the task of entering those OCRd digits online.

The second factor was the light conditions in which the OCR app performed the capture, which affected subsequent processing.

A more workable solution would be to have an OCR app which could tell the user when the digits on the device have changed.

The addition of a stand the phone could sit on, at an optimal distance from the hardware device, in a well lit room may control variances due to light and user steadiness during capture.

If these 2 suggestions could be integrated into the Banks cell phone banking app, a complete solution for the visually impaired user would be possible.

CAPTCHA Solutions

Solution 1: Resistor CAPTCHA

Reliance on Visual only CAPTCHAs, is common on the web. The workaround for this Resistor CAPTCHA was developed for a visually impaired electronics blog reader, who was prevented from commenting on blog stories.

The Resistor CAPTCHA consists of an image of a resistor with different coloured bands. Each band has a slider associated with it. Each slider consists of 10 colour values, and the user is required to use a slider handle with a mouse, to drag the handle to the correct colour value on the slider.

The resistor CAPTCHA is not considered to be a good overall design, however for the majority of Internet users, desire to view and interact with a web site or application will over rule any concerns introduced by security design.

The workaround consists of 2 simple JAWS 13 screen reader scripts that were made to work on the users Windows PC.

- The first script produces a dialog box, from which the user can choose the resistor band they want to sample the colour for.
- The screen reader then moves the mouse cursor to the resistor band in the image and samples the colour on the resistor and the cursor jumps to the matching slider handle div.
- The user needs to activate the mouse emulation cursor (known as the JAWS cursor), enable scroll lock and arrow down.

- At intervals the user runs the second script, which samples the colour on the slider and tells the user.
- Once the user is finished, they have to toggle scroll lock off and activate PC cursor mode.

Issues with this Resistor Captcha solution are:

- The user has to listen to a lot of instructions, and follow them carefully.
- The user has to count down arrow keys carefully as they are pressing them

The source code for this solution can be downloaded from the following location:

https://github.com/britta-nz/blindsided-by-security/tree/master/Resistor%20Captcha

Solution 2: Web Visum

If a website uses reCAPTCHA, then the Web Visum plugin was determined to be an effective means for a visually impaired Internet user to be able to solve the text reCAPTCHA.

Results of a 28 day Web Visum CAPTCHA solving trial

This trial was conducted between August 1 and August 28, 2012, using the Parliament Make a submission website. The trial was conducted to test the effectiveness, and determine the average length of time to solve a reCAPTCHA text CAPTCHA.

The average time to solve a reCAPTCHA was 33 seconds. Out of a total of 28 CAPTCHAs, 20 were correctly solved, 6 were not correctly solved, and 2 were not able to be solved.

Security Design Decisions Solutions

Solution 1: Security Product Decision - VPN application JAWS usability script

The solution to the VPN security application icon not being keyboard accessible (and hence the application not being able to be used), involved a JAWS screen reader script, which was installed and compiled on each visually impaired users computer.

The script allows the screen reader cursor to jump onto the icon using relative positioning, the screen reader then uses mouse emulation mode to double click the icon. Once the logon window opens, a user can successfully complete the login process.

The source code for this solution can be downloaded from the following location:

https://github.com/britta-nz/blindsided-by-security/tree/master/Shrewsoft%20VPN%20Jaws%20script

Solution 2: Chrome Web Browser - Usability versus Security

In the early stages Chrome developers made their own screen reader available to be used. It is called Chrome Vox.

Aside from the fact that New Zealand visually impaired users tend to prefer their own screen readers, visually impaired Windows users found that the default Chrome Vox navigation commands rotate their screens, instead of navigate them around web pages.

A better solution to Chrome Vox, in terms of usability, was implemented from Chrome version 17 onwards.

Chrome uses a cache of the accessible DOM tree in the browser process. This only happens if a screen reader is detected.

When the renderer accessibility flag is on, the browser requests a complete accessibility tree from the renderer every time a page finishes loading. Each node in the tree contains a reference to a unique ID in the renderer, allowing it to associate every node with the corresponding WebKit node.

When Assistive Technology requests accessibility information, the response is served directly from the browser's cached accessibility tree. When the renderer changes anything on the page, it notifies the browser, and the browser notifies the Assistive Technology. [20]

Conclusion

Web applications can be challenging for those users with visual impairments. The implementation of common security features such as multi-factor authentication, CAPTCHAs and visual security clues can make the difference between an inclusive and enjoyable web experience and the complete exclusion of this demographic.

Catering to the needs of the blind however, need not be difficult, expensive or at the cost of innovation. This whitepaper has outlined several simple implementation changes that will help New Zealand web application developers to create secure and useable Internet experience.

References

[1] BNZ, “Security Certificates Advisory,” [Online]. Available: https://www.bnz.co.nz/personal-banking/footer/privacy-and-security/security-certificates.

[2] WebAim, “CAPTCHA Survey,” [Online]. Available: http://webaim.org/projects/screenreadersurvey4/#captcha.

[3] www.hackaday.com, “Hack A Day - Stillwalker beat audio recaptcha,” 15 June 2012. [Online]. Available: http://hackaday.com/2012/06/15/stiltwalker-beat-audio-recaptcha/.

[4] Imperva, “Imperva Security Blog - A Captcha in the Rye,” June 2012. [Online]. Available: http://blog.imperva.com/2012/06/a-captcha-in-the-rye.html.

[5] TextCaptcha, “Do you really need a captcha?,” [Online]. Available: http://textcaptcha.com/really.

[6] Travel Sim, “Travel Sim Homepage,” [Online]. Available: http://www.travelsim.co.nz/.

[7] 23andMe, “Kit Registration Page,” [Online]. Available: https://www.23andme.com/user/claim/.

[8] AdaFruit, “Blog,” [Online]. Available: http://www.adafruit.com/blog/.

[9] NZ Parliament, “Submissions Page,” [Online]. Available: http://www.parliament.nz/en-NZ/PB/SC/MakeSub/.

[10] Consumer NZ, “Report a Scam,” [Online]. Available: https://www.consumeraffairs.govt.nz/report_scam.

[11] Web Visium, “Web Visium Browser Addon,” [Online]. Available: http://www.webvisum.com/.

[12] NetSafe, “Security Central,” [Online]. Available: http://www.securitycentral.org.nz/.

[13] NetSafe, “NetSafe NetBasics,” [Online]. Available: http://www.netbasics.org.nz/.

[14] NetSafe, “Computer Security Section,” [Online]. Available: http://www.netsafe.org.nz/keeping_safe.php?sectionID=computers&pageID=253&menuID=253.

[15] H. Accessibility, “Simple Web Componants - ARIA Example,” [Online]. Available: http://www.html5accessibility.com/tests/webc/samples/aria-relationships/.

[16] W3C, “Introduction to Web Componants,” [Online]. Available: http://dvcs.w3.org/hg/webcomponents/raw-file/tip/explainer/index.html#introduction.

[17] W3C, “Shadow DOM,” [Online]. Available: http://dvcs.w3.org/hg/webcomponents/raw-file/tip/spec/shadow/index.html.

[18] W3C, “WAI-ARIA Usage Guidance,” [Online]. Available: http://www.w3.org/TR/wai-aria/usage.

[19] Mozilla Corporation, “Greasemonkey Add-on,” [Online]. Available: https://addons.mozilla.org/en-US/firefox/addon/greasemonkey/.

[20] Chromium, “Accessibility Technical Documentation,” [Online]. Available: http://www.chromium.org/developers/design-documents/accessibility.

[21] [Online]. Available: https://docs.google.com/a/in2security.org.nz/document/d/1udgNGTB4X4RuygZkuhEJ3GU5wVIMbruaiQQikwURQio/edit#bookmark=id.owrf9gciuqno.

[22] WebAim, “Screen Reader User Survey #4 Results,” May 2012. [Online]. Available: http://webaim.org/projects/screenreadersurvey4/.

Appendices

Appendix A:Screen readers, Accessibility APIs, WAI-ARIA

Visually impaired people use screen reading and magnification software. Most do not use a Mouse, because using one requires the user to see where they are clicking/pointing on screen. Visually impaired users currently use, keyboards, gestures and voice recognition to interact with web sites.

Screen readers

Screen readers are text to speech software. They read text displayed on screen out for the user to hear, query objects to inform the user what interaction is required of them, and allow the user to navigate. On a touch screen phone for example, the screen reader doesn't just read out the UI element a user is touching, the screen reader also allows the user to touch the screen, without activating the element they are touching, so the user can decide whether they want to interact with the element or not. A screen reader provides information, allows a user to interact and provides a way for the user to navigate and explore effectively.

Linux screen readers

- ORCA ,
- speakup ,
- Adriane Knoppix with SBL ,
- Vinux project …

Mac and iOS screen reader

- Voiceover (free) (iOS triple click Home, Mac Cmd+F5 on/off)

Windows screen readers

- NVDA (free, open source) ,
- JAWS,
- Window Eyes,
- System Access to Go ,
- Supernova …

Android screen readers

- Talkback (free, open source) ,
- Mobile accessibility (paid)

Scripting screen readers

Screen readers can be scripted, but each screen reader has their own language and methodology for installing/compiling them.

The NVDA screen reader uses Python. JAWS screen reader uses a Proprietary pseudo language, which comes with a large function library. Window Eyes screen reader uses VBScript or JScript. Orca screen reader uses Python.

This difference in scripting languages and methods in applying scripts, leads to custom, one off solutions, which have to be re-scripted to apply to any other users screen reader.

For desktop applications these one off solutions are regularly the only method of being able to solve a visually impaired person's usability issue.

For the web however, the introduction of WAI-ARIA means that, one solution (and a little cross screen reader testing) can be applied to a usability issue, and the solution will work for any screen reader which has implemented support for WAI-ARIA.

For web based issues, this WAI-ARIA "one solution fits all screen readers", has now given web based applications and security implementations a tangible ability to be used by a visually impaired Internet user.

Refreshable braille displays and the web

Screen readers also output information via refreshable braille displays. The screen reader outputs text information to both audio, and the equivalent braille.

Braille is a 6 dot system, with each 6 dot sign representing either one character from the alphabet (un-contracted braille), or a sequence of characters (contracted braille). For example, on the internet, if the user is focused on a button, the braille display will show the braille signs for 'btn'. If the button is labelled "send", the braille signs on the display would show s5d, with ASCII 5 representing the contracted braille sign for "en". The whole button would therefore be displayed as, 'btn s5d', on the visually impaired users refreshable braille display.

Accessibility APIs

Accessibility APIs are platform specific APIs. When an element on a webpage receives focus, the browser communicates the role, state, properties and parent/child relationship information of the element, to the screen reader, via the accessibility APIs. The operating system also uses these accessibility APIs to bi-directionally communicate with a screen reader. This bi-directional communication allows the screen reader to inform the user what an element on a webpage is, and how they need to interact with it.

List of Platform specific Accessibility APIs

- Linux AT-SPI 2
- Windows/Linux IAccessible 2
- Windows MSAA
- Windows UIA (The Mono Accessibility project makes UIA available in Linux)

- Mac OS Ax/uiA
- iOS UIAccessibility Protocol Reference
- Android Accessibility API
- Java Access bridge

WAI-ARIA

WAI-ARIA consists of roles, states and properties, mark-up that can be added to a webpages HTML, in order for the screen reader to be able tell the user how to interact with a dynamic web element. An overview of WAI-ARIA and HTML 5 and some jQuery examples, can aid the understanding of the WAI-ARIA specification.

For a complete overview of all of the above concepts, and how they fit together, the book Pro HTML 5 Accessibility by Joshue O'Connor is recommended. Neither of the whitepaper authors is in any way affiliated with the author of this book. It's simply a good book on the subject.

Appendix B: FAQ - Visually Impaired Technology in NZ 2012

There is a preference for certain technologies in the New Zealand visually impaired community. These preferences can change over time, depending on changes in technology. There are some very good, user specific reasons for the preferences, and these examples are a current snapshot in time of those preferences:

Mac versus Windows versus Linux in 2012

In New Zealand a lot of visually impaired users have Windows PCs. One of the main reasons are that other family members may be using Windows PCs, and so a visually impaired user can get local support for their computer, by using the same technology.

A sighted desktop user can approach a Mac, a Windows machine and a Linux Gnome desktop, and be reasonably comfortable finding where they need to go and what they need to do. Just by having used one OS, a sighted person will intuitively know where to click in another OS. This is due to the visual design of the desktops, and how similar those desktops look on the 3 OS'.

A visually impaired person can also use all 3 operating systems, with the OS specific screen readers, however, there is a little bit more of a learning curve involved for the visually impaired person to switch to, or between, these 3 operating systems. The reason a visually impaired person cannot just start using a Mac, if they have never used one before and have only ever used Windows, is because the screen reader keyboard shortcuts, terminology and functionality, and some of the OS shortcuts, are different.

For example, on a Mac, the Voiceover screen reader user has to interact and stop interacting with elements on a website, by pressing special interact/stop interacting shortcuts, and elements on a web page can be selected via a web rotor. On a Windows computer, no explicit interaction commands are necessary, the screen reader user can use navigation shortcuts or browse the webpage top to bottom using the arrow keys, or use elements lists, like links lists or headings lists. On Linux no elements lists exist, but navigation shortcuts do.

Differences in shortcuts can be learned, but it is not like a sighted person walking up to a Linux desktop and simply clicking menus with a mouse. Keyboard shortcuts differ between the platforms, and visually impaired users do have to look those shortcuts up and learn them.

With the more wide spread use of voice recognition and gesture input, it is hoped that the differences between the 3 OS', as currently experienced by visually impaired computer users, will have less impact

For an actual list of screen reader shortcuts, which will illustrate the differences, check out:

Mac (Voiceover, free, inbuilt Mac and iOS screen reader)

- Voiceover commands Part 1
- Voiceover commands Part 2
- Voiceover Commands Part 3
- Voiceover Commands Part 4
- Voiceover Commands Part 5

Windows (JAWS 13 commercial screen reader)

- JAWS commands

Windows (NVDA 2012.2.1, free, open source screen reader)

- NVDA commands

Linux (ORCA free, open source, inbuilt Gnome screen reader)

- ORCA commands

Android phone versus iPhone, which is the better phone in 2012?

In the sighted world this question is debated. In the visually impaired community in New Zealand, up until Android 4 came out, there was little debate. iPhones can have the screen reader turned on by triple clicking the Home button 3 times, instant access. And a lot of apps are able to be used with the Voiceover screen reader.

On Android pre version 4, someone had to turn on TALKBACK screen reader for the visually impaired

person, or, TALKBACK had to be installed (the initial Gmail account setup process also had to be done). Android 4 is making strides, but it is useful to know the history and reasons behind, the differences in technology adoption amongst visually impaired cell phone users, versus sighted cell phone users.

Most default apps on Android cannot be used with the TALKBACK screen reader. The default email program and calendar are generally not accessible, and up until June 2012, neither Firefox, nor Google Chrome, nor the default browser found on Android phones, was able to be used via the screen reader. Special apps had to be downloaded to browse and use email, but couldn't be accessibly downloaded through the Google Play app on the phone. So, up until recently, a visually impaired cell phone user had to download and install apps via the Google Play website, on their computer, with their computers screen reader with the phone plugged into the computer.

This is why a visually impaired New Zealand cell phone user will currently tend to buy an iPhone if they can afford it, or buy a Nokia C5-00 Symbian phone with talks or mobile speak software installed on it.

Blackberry has had a free screen reader available since May 2012. Few visually impaired cell phone users in New Zealand have a Blackberry phone. Prior to May 2012, they had to buy the blackberry phone and pay $500 for the Oratio screen reader as well.

Google Chrome versus Firefox which is the better browser in 2012?

In the sighted world this is discussed and debated. In the visually impaired community until Chrome 17 came out, there was no discussion. Chrome was not usable with a visually impaired persons own screen reader. Chrome Vox, the Chrome screen reader is not popular with visually impaired New Zealanders. The Chrome Vox default navigation keyboard shortcuts, happen to be the screen display flipping shortcuts in Windows 7. The default shortcuts can be changed, but it causes a lot of confusion initially, when nothing reads in Chrome.

This is why visually impaired New Zealand browser users, will currently tend to favour using Firefox or IE. Chrome cannot be used in Linux yet with ORCA, the Gnome screen reader.

KDE versus GNOME, which is the better Linux desktop in 2012?

In the sighted world this question is discussed constantly. In the visually impaired community this is not discussed. GNOME is accessible via ORCA screen reader, KDE are only just playing catch-up now, and they are trying to leverage the work done by the ORCA team. This is why a visually impaired Linux GUI user, will not tend to use KDE at the moment.