

CORSAIRE

EXPERTS AT SECURING
— INFORMATION —



A CORSAIRE WHITE PAPER: STATE OF THE WEB

A REVIEW OF APPLICATION SECURITY TRENDS OVER 5 YEARS

Project Reference	State of the Web.doc
Author	Glyn Geoghegan
Date	31 August 2007
Distribution	Client pre-release



Application Security Trends

Executive Summary

As business has embraced technologies like the Internet to provide an easy and convenient interface to clients, partners and employees alike, applications have become increasingly complex, and heavily integrated with sensitive internal systems.

As a result, the impact of a poorly built web site or application is no longer limited to defacement and brand-damage, but typically now leads to costly and embarrassing exposure of personal data, intellectual property or financial information. With the corresponding increase in legislation and corporate governance, and schemes such as the card industry PCI requirements, the impact on business can be severe.

As such, bespoke applications, particularly those exposed to the Internet, have become the Achilles heel of many otherwise secure organisations, undermining the benefits of the traditional network orientated security that has been heavily invested in over the past 10 years and more.

While the concept of application insecurity is not new, it is still the least understood and mitigated risk in many environments.

This paper is drawn from a representative sample of application security assessments, conducted by Corsaire over the last 5 years. The results have been collated and analysed, and a number of anecdotal and statistical conclusions have been

drawn from the data, which are explored in the following sections.

Security is considered as it relates to a number of key application areas, detailed in *General Findings* on page 4.

Figure 1 shows improvement in most areas over the 5 year period.

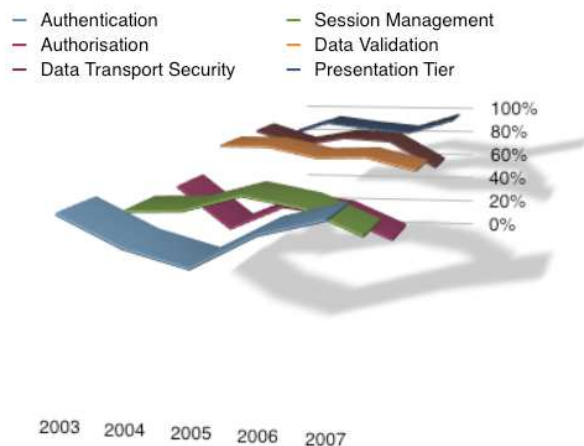


Figure 1 – Annual Trends (percent of tested applications with security flaws)

“Application security is improving, but as the threats evolve and complexity increases, risks are invariably present...”

While our study indicates that security risks were present in all of the tested applications, this must be taken in context. We are not comparing like with like – web applications in the early 2000s were often still static brochure sites with limited user interaction, applications tested now are far



Application Security Trends

more complex – and with complexity comes additional risk.

Two areas related to identifying the user and security within the presentation tier show a sudden upward trend; implying security standards have fallen in those areas. This is slightly deceptive, however, and relates to lower impact threats in the more complex systems now in use. This anomaly aside, application security does appear to be improving, at least within the sample set. Clearly submitting an application for third party assessment demonstrates a level of security awareness within the application owners that is not universally present.

“Security is too often a secondary consideration during application development...”

In many earlier assessments, it was clear during testing that the insecurities identified related to poor security during design and development, and subsequent attempts to retro fit the security. While this has become less common regarding internally developed applications, there is often still an inherent lack of security in the development process, leading to costly re-engineering at later stages when the risks are identified.

It is especially vital when outsourcing development that the security concerns and requirements are established through a risk assessment, and contractually mandated in order

to ensure security of the application and therefore the organisation.

“Sensitive data is still inadequately protected in transit...”

Encryption should be a ubiquitous part of any application handling private data, but all too often it is either left out, inadequately enforced or inappropriately used, potentially exposing that data to eavesdroppers on the Internet.

This exposes organisations in a number of ways, often under data protection and privacy legislation.

“Applications do not adequately validate users’ identities and manage their access...”

The purpose of most applications is to provide the right user with the right data; be it their banking information, shopping basket, online resources or any other tangible asset.

Unfortunately, this is often poorly enforced allowing inappropriate access to the wrong data, and exposing private data.

“Inappropriate trust is placed on remote user input...”

A common mistake in software engineering in general is to assume that users will only provide the expected input. This lack of foresight and misplaced trust is the root cause of many software vulnerabilities, and therefore security risks.



Application Security Trends

“Not all threats are equal...”

In some ways, the figures distort the reality of improvements in application security. *Show stopping* vulnerabilities are now found in a minority of applications.

More often areas where significant improvements can be made are found, rather than those with completely inadequate security.

“What does the future hold?”

This study reveals there have been notable improvements in application security, but the battle is not yet won.

As mentioned, increasingly complex applications have more and more associated risks.

Most of these risks may be eliminated through a more structured approach to security in development.

Rigorously specified security requirements, backed by developer education and regular security testing checkpoints, enable the organisation to define what security is, what is required and ensure personnel have both the understanding and skills required to deliver.



Application Security Trends

Table of Contents

EXECUTIVE SUMMARY	2
TABLE OF CONTENTS	5
OVERVIEW	6
GENERAL FINDINGS	6
ANALYSIS	8
Authentication.....	8
Session Management	11
Authorisation (access control).....	13
Data Validation.....	15
Data Transport Security.....	16
Presentation Tier.....	18
TRENDS.....	20
CAVEATS	22
OTHER OBSERVATIONS.....	22
'n from m' authentication schemes.....	23
Mouse-based password entry	23
Security by omission.....	23
One Time Password (OTP) / token authentication	24
CONCLUSIONS	24
APPENDIX A.....	26
References	26
ACKNOWLEDGEMENTS.....	26
About The Author	26
About Corsaire.....	26



Application Security Trends

Overview

Application Security is a rapidly evolving area, and one of increasing interest and activity as Information Security matures. It is important to understand the key areas of concern within web-applications and which of these are exposing organisations to the greatest risk in order to develop strategies to mitigate and manage that risk.

This paper will demonstrate that while application security is improving, fundamental security flaws still affect almost all applications, and many of the same mistakes present years ago are still made in new applications.

General Findings

The analysis considered around 45 findings categories grouped into 7 key areas from a representative sample of security assessments on applications from leading companies in a number of sectors across the UK, Australia, Europe, the US and Asia.

Corsaire Application Security Assessments are typically performed on the following interdependent areas from a security standpoint:

- *Authentication* – to determine how the application validates the identity of the user
- *Authorisation (Access Control)* – how the application subsequently manages user access control over protected resources and data
- *Session Management* – to analyse how the user's session is maintained and managed during their interaction with the application
- *Data Validation* – to assess the effectiveness of the application's handling of expected and unexpected data passed between it and the client
- *Data Transport Security* – to review how the data sent between the client and application is secured and protected in transit
- *Presentation Tier* – to ensure the platform on which the application is presented is adequately secured to prevent compromise of the application or its data.

Security issues were identified within all of the tested applications. It is important to note that this does not mean that every application had exploitable issues violating security; rather that none conformed to every security control defined and tested for. The summary results relate to all levels of risk, from critical problems to low risk concerns. Figure 2 below provides a summary of the results in each category across the 5-year period.



Application Security Trends

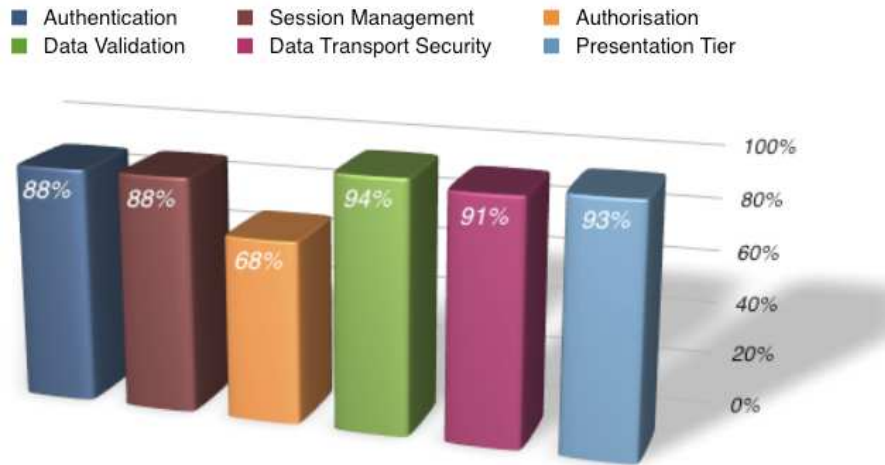


Figure 2 – % applications exhibiting each category of security vulnerability

Additionally, when testing, each application must be taken in context. For example, the security and usability concerns relating to an Internet Banking application are subtly different from those applied to an online retail site and very different from marketing or survey sites.

However, there are security concerns associated with any online application; such as data protection legislation applying to any personal data or protection of the corporate brand. Whilst the specific implementation of security controls may therefore vary, the core security concerns defined above relate to almost all applications.



Application Security Trends

Analysis

Authentication

The most serious security risk associated with the authentication process is that of subversion. If the login mechanism fails to properly identify the user and permits access to the application, the rest of the security mechanisms become irrelevant.

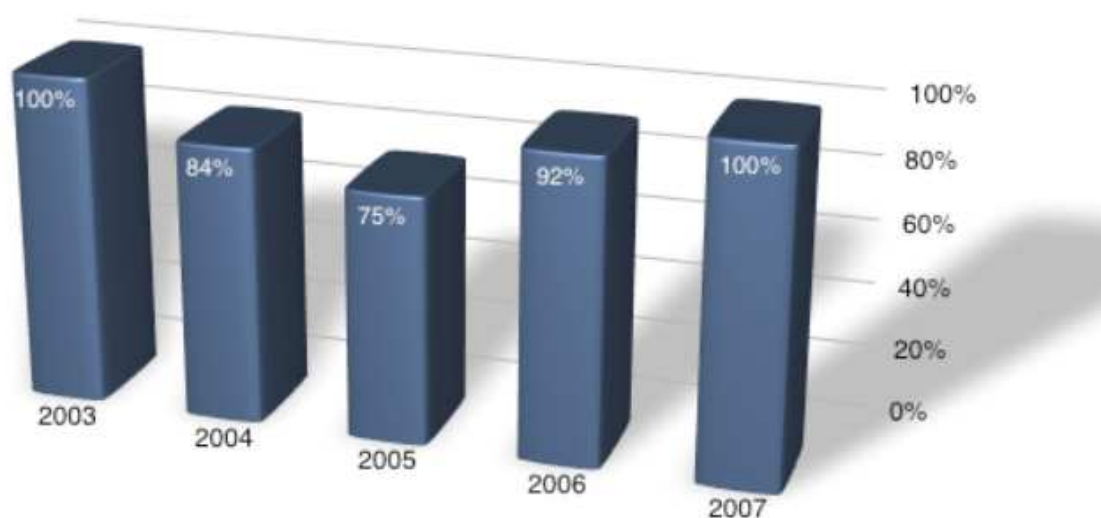


Figure 3 – % of applications with authentication security vulnerabilities

More commonly, however, the authentication mechanism will effectively enforce the login model, but demonstrates other insecurities. In fact, almost all tested applications (99%) had insecure authentication, as they were susceptible to credential replay attacks; for example through a key-logger, as they did not use a suitable form of 2-factor authentication (2FA). Even when taking this finding out of the results (as 2FA has a notable associated cost and may not be appropriate for every application) at least 75% of applications each year and 88% overall exhibited insecurities within the authentication model.

In many cases, the authentication model was not strong enough to adequately protect the application, with weak and simple passwords permitted in 53% (and as mentioned above, 99% susceptible to key-loggers).

44% did not enforce lockouts or escalating timeouts, exposing them to brute-force attacks. Even where bespoke 'secure' login procedures are implemented (particularly by financial institutions), these are frequently inadequate, offering limited (or no) improvement.



Application Security Trends

Authentication is also rife with information leakage, increasing the risk of unauthorised access to user data. Often usernames are based on publicly available or guessable information, such as email addresses, account numbers or even social security numbers. This is a growing trend – only 12% overall but affecting 14% of the applications tested so far in 2007. Login failure messages frequently allow enumeration of valid accounts, and even where they do not, the password recovery mechanism frequently does, affecting 44% overall.

While less common, it is not unusual to encounter HTTP Basic Authentication (and other weak schema) implemented as the mechanism for providing authentication services, 8% on average peaking at 16% in 2006. Basic Auth is grossly inadequate for modern web-applications, exposing usernames and passwords through Base64 encoding, providing no real session management controls and little granularity of access control. Its use is far more common on the Internet than our results would indicate, as typically Basic Auth applications do not get submitted for security assessments by third party consultancies.

Poor techniques in managing user registration, account lockouts and recovery lead to a number of the application layer denial of service attacks, affecting 36% of the tested applications.



Application Security Trends

- User enum & info leakage
- Weak passwords
- Weak authentication (e.g. Basic Auth)
- Insecure 'remember me' functionality
- No password change
- Password recovery insecurities
- Inadequate re-authentication
- Default/guessible credentials
- Public usernames
- Susceptible to key-logger
- No lockout/susceptible to brute-force

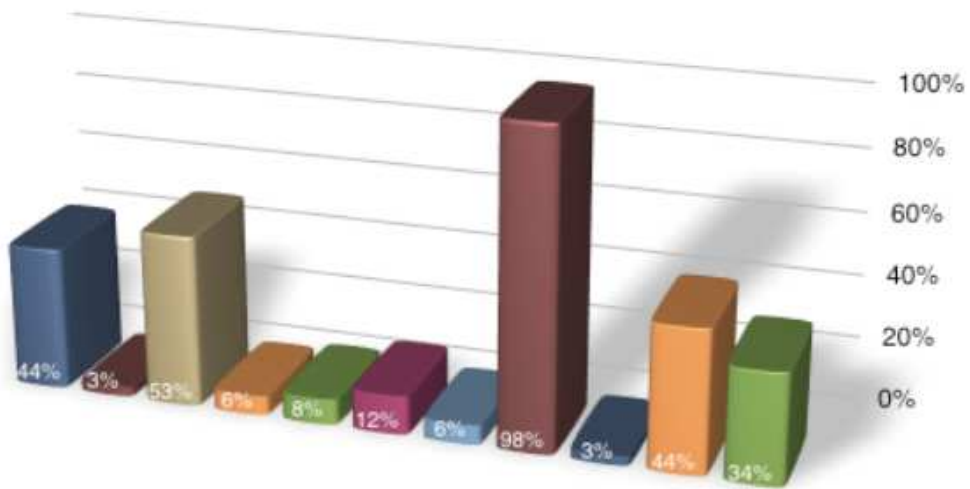


Figure 4 – % of applications with individual security vulnerabilities



Application Security Trends

Session Management

Authentication is only a small part of managing a user's access to data. Critical to the process and oft overlooked is the area of session management. Insecure session management can negate the security implemented by both authentication and access control (authorisation) mechanisms.

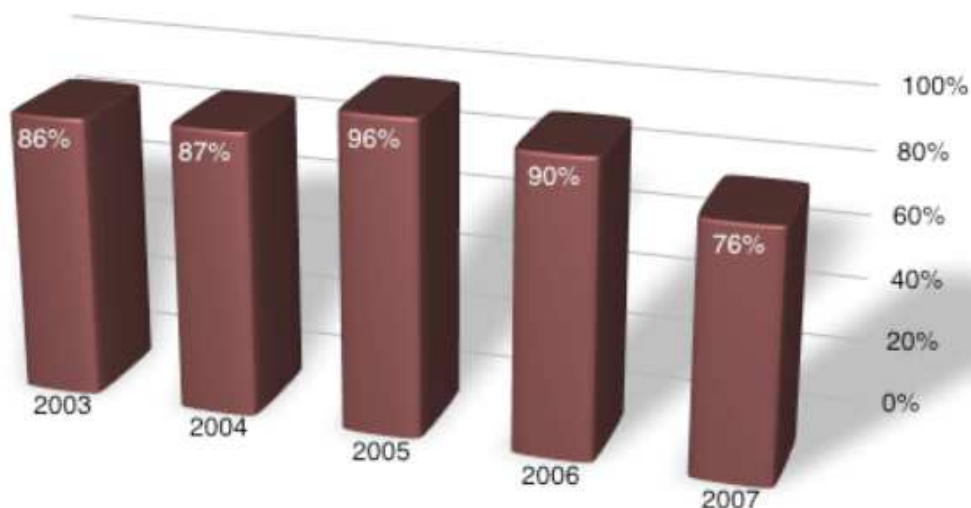


Figure 5 – % of applications with session management security vulnerabilities

Due to the limitations of HTTP, all interactive applications need to connect the user's stateless HTTP session with their stateful application session. Typically, this is achieved through a Session ID sent to and associated with the user during the login process, often as a cookie. Having authenticated, this Session ID is the only token that the application checks when determining what data the user may access, and what privilege they have. As such, it is vital that the Session ID is treated with at least the same high level of security as the user's credentials. Compromise of the Session ID *will* allow an attacker to assume the identity of the user.

88% of the tested applications contained flaws in their session management implementation, with 9% displaying no effective session management (e.g. using a stateless technique such as Basic Auth).

It is vital that the Session ID is always handled securely and passed over encrypted transport to maintain privacy of the token, and therefore user. The *secure* cookie directive helps ensure this, but was not present in 53% of the applications. It is also vital that the cookie is sufficiently random, to prevent easy prediction by an attacker; 29% were insufficiently random, 22% were persistent or reused (greatly increasing the chance of brute-force attacks) and 7% were static, the greatest level of exposure.



Application Security Trends

It is also important that the individual sessions are securely managed, with effective logout/logoff/revocation (41% were missing or non functional) and session timeouts (35% did not).

Frameworks like J2EE and ASP.NET provide proven, robust and frequently tested session management implementations. Despite this, 21% of tested applications used their own bespoke session management, including some built on the above platforms. All of the bespoke session management implementations harboured security issues, including resource management issues contributing to the 36% of applications susceptible to application layer denial of service.

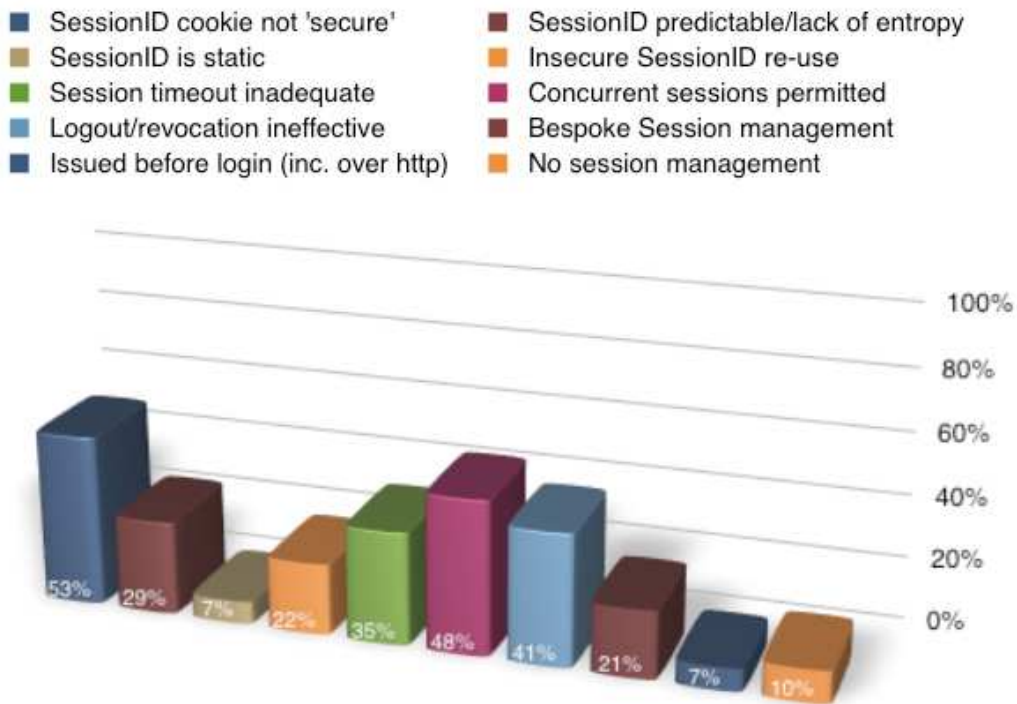


Figure 6 – % of applications with individual security vulnerabilities



Application Security Trends

Authorisation (access control)

Having confirmed the identity of the user and tracked them through session management, the application has to enforce access control over its data. This should be performed each time access to a resource is requested, by checking the user's identity (through their session ID) against an access control matrix. Invariably, the purpose of a web-application is to provide only the correct information to authorised users, and as such strict authorisation, or access control, is another essential element to application security, and insecurities will adversely affect the application and its data.

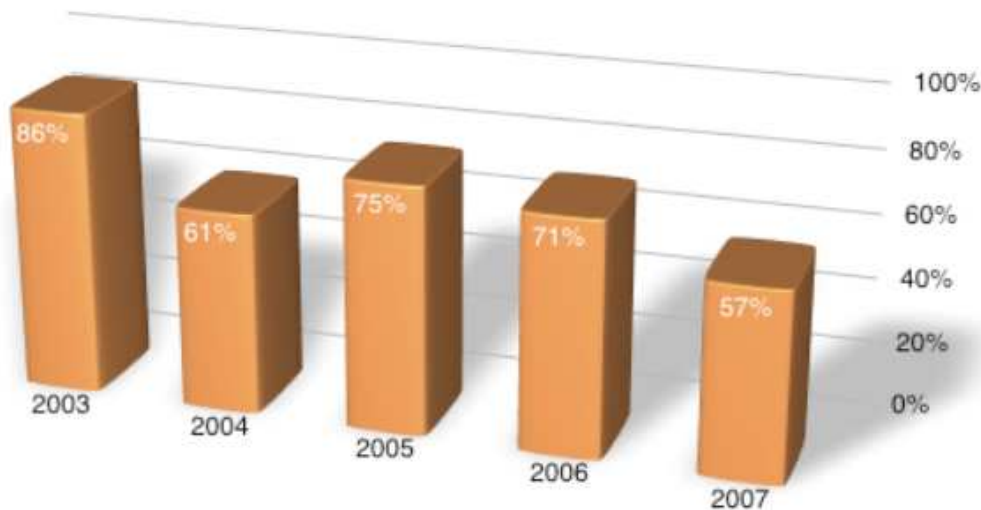


Figure 7 – % of applications with authorisation security vulnerabilities

Inadequate access controls affected 49% of the tested applications, although this has been steadily dropping from 86% in 2003 to 57% so far in 2007. In 6% of cases it was possible to escalate the user's privilege to that of administrator (either of the application or the web-server itself).

Frequent mistakes include: assuming that if a user 'knows' how to access a resource (e.g. the URL) then they are permitted to access it. This is commonly associated with applications that use restricted views to manage user access control; that is custom menus or pages are generated for the user that only contain links to the resources they are permitted to access. However, if no further checks are made, then users can guess (or observe) how to access resources of other users and access them directly.

Other common errors include using simple references in the URL or forms to specify resources, for example a static client ID or account number. Again, if no access control checks are made against the resource and



Application Security Trends

the user's identity it is trivial for an attacker to guess or brute-force unauthorised access to data. In poorly designed applications these URLs can often be accessed without authenticating.

It is a universal rule that complexity is the enemy of security: the old adage of KISS applies in particular to application security. Where possible, a single token (the Session ID) should be passed between client and server, and all other user tracking and access control conducted using data stored at the server-side. In 30% of applications, potentially unnecessary and excessive data was passed back and forth (in cookies, URLs, hidden form fields, view-states etc.) and in many cases it was this interaction that could be subverted to bypass the proper authorisation controls.

Without proper access control, authentication and session management are rendered ineffective, which results in a failure of the applications security model.

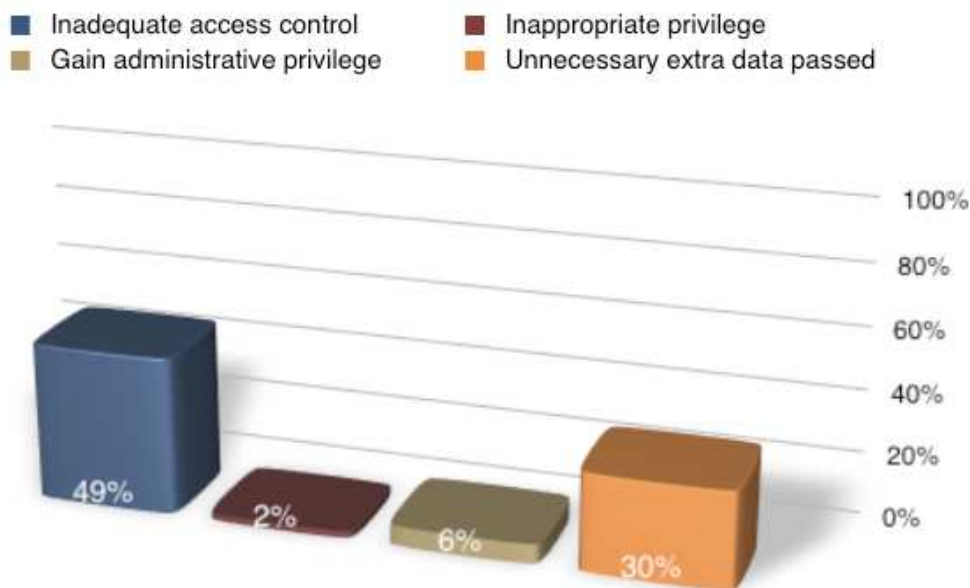


Figure 8 – % of applications with individual security vulnerabilities



Application Security Trends

Data Validation

Having established a robust, secure and reliable system to identify, track and control access to resources for a user, security of the application, connected applications and systems and their underlying infrastructure must be considered. Many serious attacks on applications and their users rely on poor data validation on input from the client side and on data sent back to those clients. Such attacks include SQL injection, Cross Site Scripting (and variants), and more traditional attacks such as buffer overflows and format strings that have affected fat-client applications for some time. It is therefore essential that client input and application output is validated and sanitised, to ensure dangerous characters and inappropriate content are removed or encoded safely.

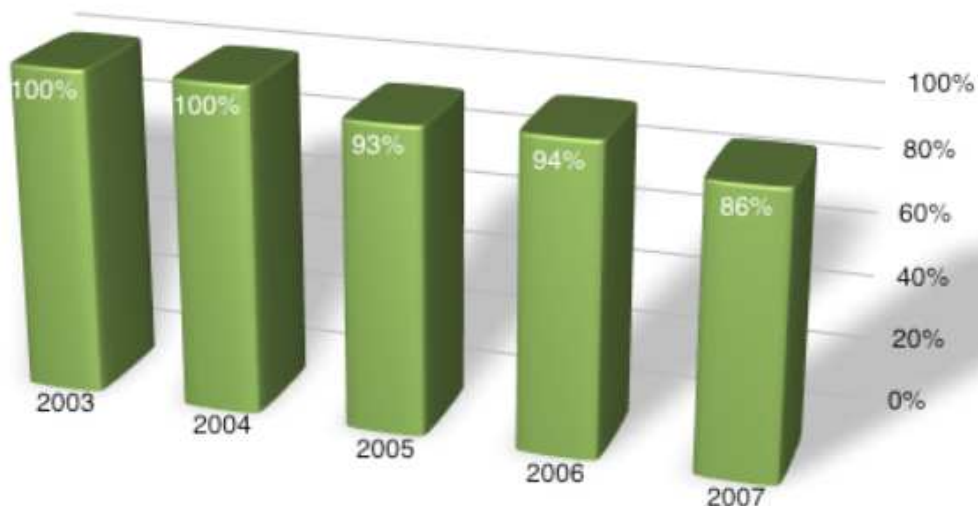


Figure 9 – % of applications with data validation security vulnerabilities

Data validation is the most commonly cited example of poor application security, leading to issues such as SQL injection (23%) that is interacting directly with the database and Cross Site Scripting and its variants (transient 60%, persistent 17% and generic HTML injection 48%) where unauthorised code is executed in the user's browser within the context of the application.

At its tameest, poor data validation and sanitisation (on input and output) leads to application errors that degrade the user experience.



Application Security Trends

At their worst, data validation flaws allow compromise (or destruction) of the data within the application, subversion of authentication and access control, social engineering attacks on users and compromise of session tokens.

A common historic mistake was to implement checks on data only at the client-side, without replicating them at the server-side. As these checks are easily bypassed by an attacker, this may lead to a common attack vector used to compromise a number of applications. Evidence of this type of flaw has dropped from 29% in 2003 to around 10% for 2006 and 2007.

Data validation security issues were identified in 94% of the applications, leading to about half of the 36% of applications being vulnerable to application layer denial-of-service attacks.

- Inadequate validation/sanitisation
- SQL injection
- XSS (transient) - including CSR etc.
- GET/POST interchangeable
- Client-side not replicated at the server
- XSS (persistent)
- HTML insertion
- Denial of Service (any application layer)

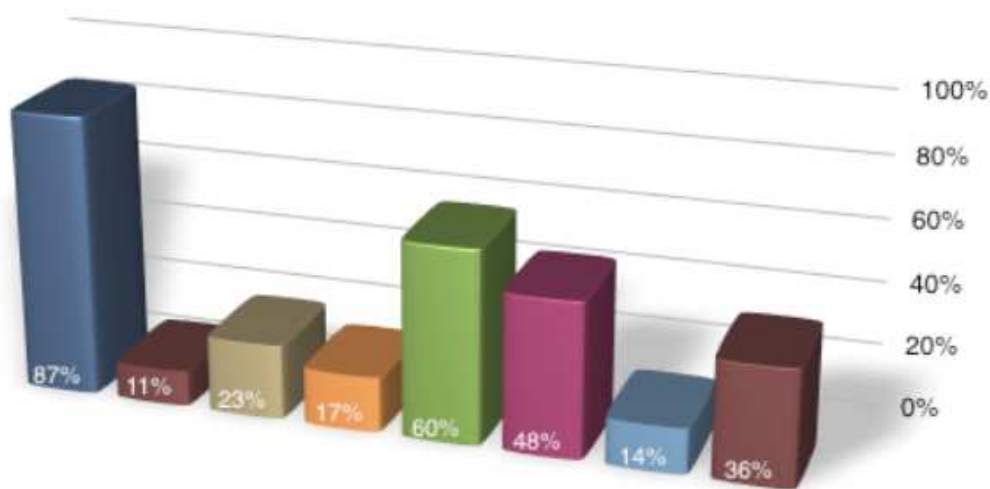


Figure 10 – % of applications with individual security vulnerabilities

Data Transport Security

The public nature of the Internet affords many opportunities for snooping on data between the user and application, or client and server. Use (and enforcement) of end-to-end encryption and strong controls to prevent caching of data by local or upstream devices is a vital step in ensuring the security of data in transit.



Application Security Trends

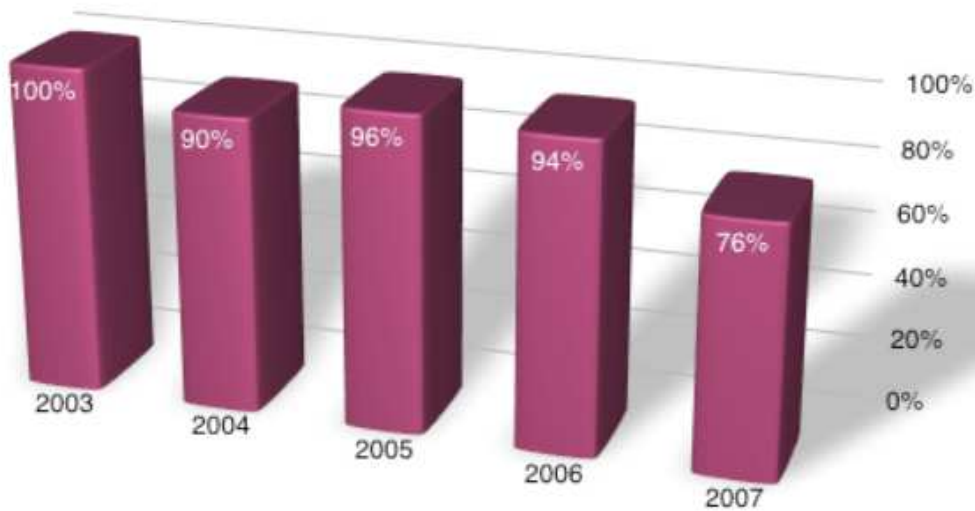


Figure 11 – % of applications with data transport security vulnerabilities

While encryption has become a de-facto security measure in most applications, its presence is often inconsistent, poorly implemented and in many cases undermined by other poor security practices, leading to issues in 94% of tested applications.

Bespoke 'encryption' is often present and almost always implemented badly or, more typically, flawed by design. It is folly to assume that the average programmer can design and implement a better (or even comparable) encryption algorithm to those produced by respected, dedicated cryptographers.

Even where established algorithms are used, they are often inappropriately deployed, negating their benefit. For example, encrypting a user's password before sending it with their username to the server login script provides no extra protection for the application. If an attacker intercepts the username and encrypted password, they can simply replay these to the login script to gain access. All that is protected is the user's password itself. In 63% of applications, sensitive data including usernames and passwords were sent unencrypted, with the Session ID exposed in 32%. In 47% of cases encryption was either inadequately enforced, inconsistently used or absent altogether.

Where end-to-end SSL encryption is used, in 38% of recently tested applications insecure ciphers and protocols are still permitted – including weak 40-bit ciphers or the flawed SSL version 2.



Application Security Trends

Should all of the above problems be avoided, inadequate cache controls potentially allowed sensitive data to be cached by third party devices (or the browser) in 75% of cases.

- Credentials/data sent/stored unencrypted
- SessionID sent unencrypted
- Inadequate/insecure cache controls
- Weak encryption
- Encryption inadequately enforced
- Inappropriate encryption

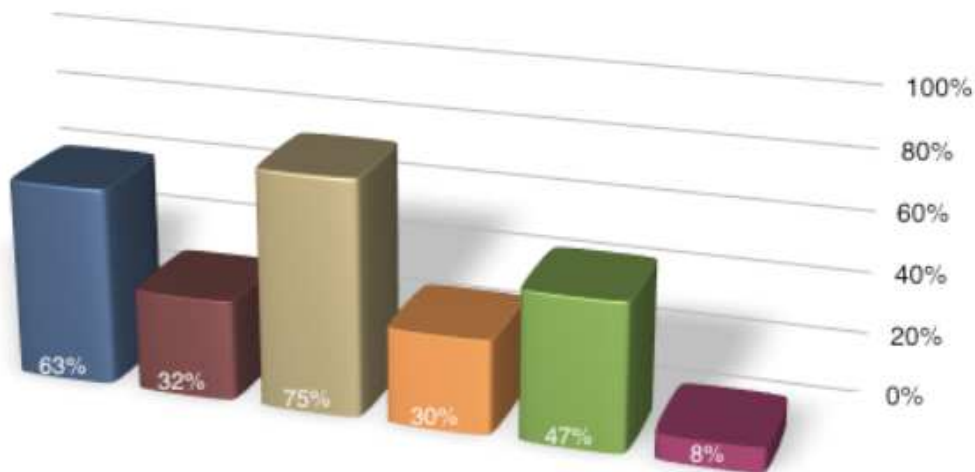


Figure 12 – % of applications with individual security vulnerabilities

Presentation Tier

The Presentation Tier can be considered the interface between the bespoke application and the underlying infrastructure on which it is presented. As with the foundations of a building, it is vital that this underlying infrastructure is robust and resilient to attack, or the application may be compromised through this layer (and vice versa). Insecure platforms may expose the database or application server, or allow compromise of the hosts for use as a staging post for attacks on connected internal systems, such as the database tier or corporate network.



Application Security Trends

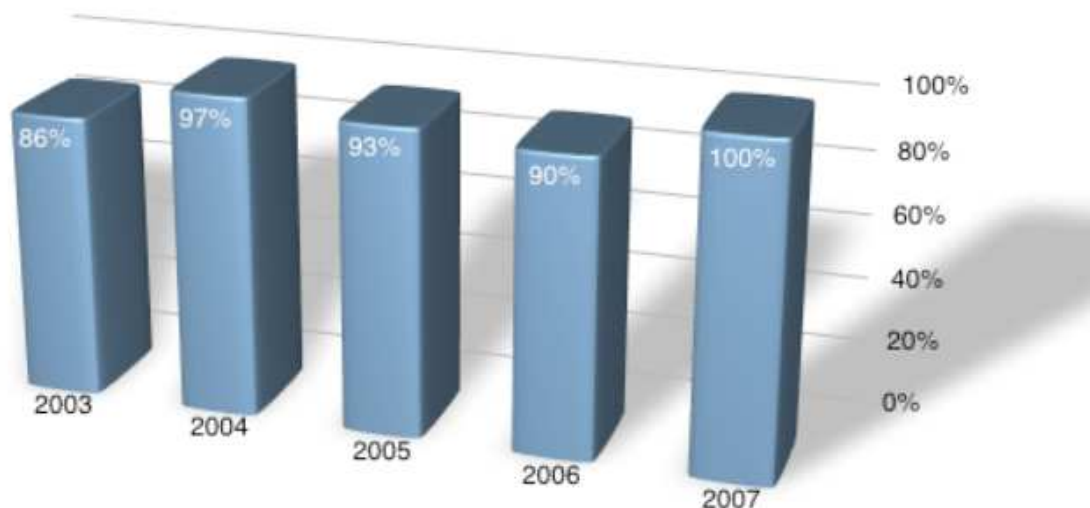


Figure 13 – % of applications with presentation tier security vulnerabilities

Security testing of the tier (e.g. the web-server and operating system software on the host) may be included in the application security assessment, or may be part of a separate infrastructure test. Regardless, in all assessments, the presentation tier is tested for the presence of old, unnecessary or insecure files, components and functions, information leakage and for its handling of error messages.

Finally, the platform on which the application is implemented is obviously critical to its security. Frequently, default or sample files, content and functionality are left on such servers, exposing them and the application to increased risk (35% overall, down from 57% in 2003).

Poor error trapping often leaks valuable information about the platform and application to an attacker, including stack traces, affecting 61% of applications and server platforms.

Information leakage through old source code, backup files and other legacy data can also contain useful information, in some cases database usernames and passwords, or even databases themselves. Server configurations leaked information in 38% of cases overall, with application source present in 27%.

Failing to implement simple measures such as disabling auto-complete on forms containing passwords (52%) and other platform insecurities lead to 91% of the presentation tiers being considered insecure, including (mainly low risk) issues with every presentation tier assessed so far in 2007.



Application Security Trends

- Auto-complete
- Information leakage (server)
- Default/sample content/functionality
- Inconsistent/informative error messages
- Information leakage (source)
- Insecure platform

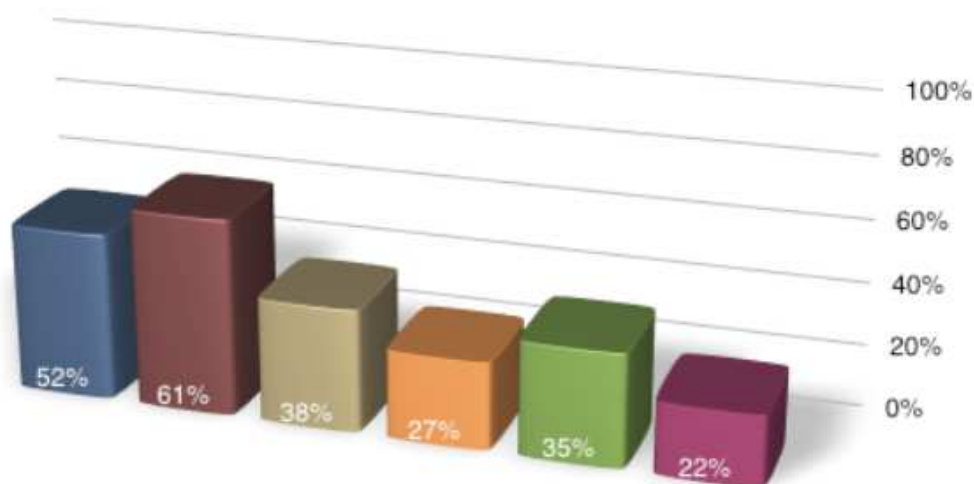


Figure 14 – % of applications with individual security vulnerabilities

Trends

A high level review of the analysis of application assessment results from 2003 to the present may initially be misleading. While weaknesses in most areas have reduced over time, insecurities in session management and authentication mechanisms fell and then rose again.

Description of Finding	2003	2004	2005	2006	2007
Flawed authentication mechanisms	100%	84%	75%	92%	100%
Insecure session management	86%	87%	96%	90%	76%
Weak access control (authorisation)	86%	61%	75%	71%	57%
Inadequate data validation	100%	100%	93%	94%	86%
Insufficient data transport security	100%	90%	96%	94%	76%
Insecure Presentation Tier	86%	97%	93%	90%	100%

Table 1 – Application security trends



Application Security Trends

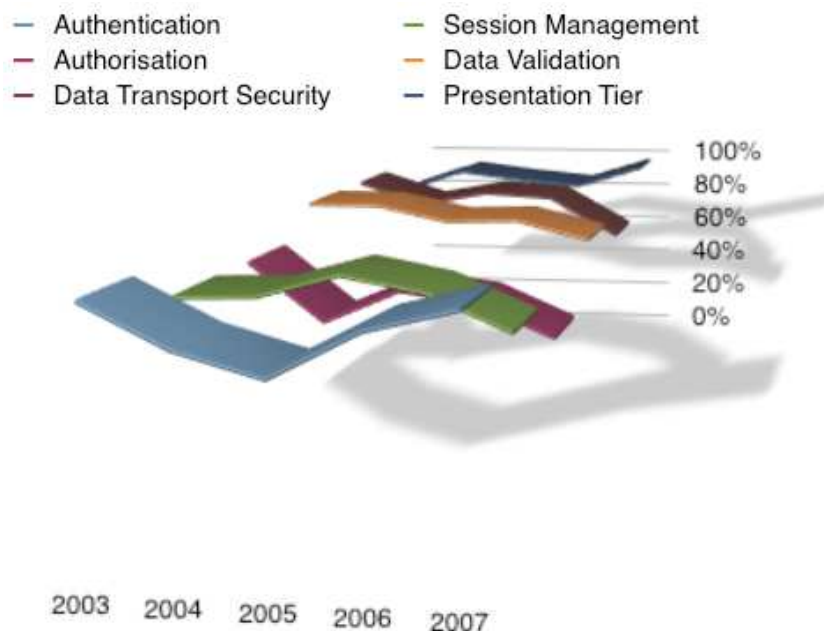


Figure 15 – % of applications with security vulnerabilities 2003-2007

The figures belie the fact that application security **has** improved over the course of the study, for a number of reasons.

Testing methodologies and benchmarks have evolved in this time. While 'inadequate input validation' in earlier studies implied a complete lack of validation within target applications, a similar finding now could indicate a single instance where data passed in a form or cookie wasn't thoroughly analysed and escaped.

The baseline for application security has also increased. It was more common to see simplistic Basic Auth or bespoke authentication session management implementations towards the beginning of the sample set. While Basic Auth is inadequate from many perspectives (lack of session control and accountability, and exposure of user credentials, for example) it does properly implement checking of user credentials.

These have steadily been replaced by more robust applications built on platforms like ASP.NET and J2EE. Some of these more complex implementations have introduced problems through authentication, highlighted by the security assessments and subsequently rectified.

The increase in outsourcing of development has often had a negative affect on security controls, and not always due to the third party. Security is frequently neglected in the planning and definition stages. Where



Application Security Trends

there is no direct (contractual) requirement - there is no incentive for third party developers to implement security – as it is often falsely deemed a costly and time-consuming exercise. It is clear from other research on secure development ROI (see References on Page 26). Furthermore, the security Intellectual Property built up within the organisation, such as the skills and awareness in internal development staff is rarely passed on to the third-party developers.

Caveats

As discussed through the trend analysis, the raw figures are affected by changes within the application and security assessment realms.

Web-applications themselves have become increasingly complex, even over the last few years. Long gone are the simple brochure-ware sites of the 90's, the applications now integrate heavily with back-end line of business systems and access more critical data in more complex ways, increasing the attack surface and criticality of any findings.

Some more complex findings, relating to application logic, or granular issues relating to application security simply were not relevant on earlier assessments. For example, a detailed analysis of session IDs and controls would not have been conducted on a Basic Auth application, because effectively there was no session management to test. Similarly, where an application was found to have no data validation, an observation that client-side validation wasn't replicated at the servers-side would have been irrelevant.

As this paper has been written in Q2, 2007, the sample set for 2007 were notably smaller than those for 2003 to 2006, leading to a single 'bad' application having more impact on the overall results.

Not all assessments included in the study were 'full scope'. Some of the data relates to application observations during an infrastructure test. Some are of fat-client applications, or specific areas of web-applications. While most were conducted as white-box tests with full privilege, some were conducted without credentials, limiting analysis of session management and access control, for example.

Finally, results of application post-fix assessments (re-tests) have generally been excluded from the study.

Other Observations

A number of security mechanisms and controls are increasingly present in web-applications, particularly those relating to financial and merchant systems to try and address the problems of phishing, key-loggers and other client side attacks. All security mechanisms that make it harder to compromise a user are to be



Application Security Trends

lauded, as is the increased awareness of security instilled in users. However, it is important that these schemes do not offer a false sense of security, potentially exposing users and organisations to greater risk.

'n from m' authentication schemes

As a defence against key-logging, many institutions now request elements from the pass-phrase or password rather than the whole password, the logic being that the attacker cannot collect the username and password in one pass.

In the case of a 6 digit PIN with 3 requested elements in sequence (e.g. 1st 3rd 6th or 4th 5th 6th) there are only 20 possible challenges. It would therefore only be a relatively small increase in time for an attacker to gather the 20 challenges and responses required to determine the user's full PIN.

Different schemes introduce different numbers of possible changes and responses, but these are always finite.

Where phishing attacks are used to draw the user to a fake site designed to gather their credentials, the sites can be constructed to gather the whole PIN. With most users having multiple accounts with different institutions and varying authentication schemes, only the most astute would notice this variation.

Mouse-based password entry

Similarly, authentication based on clicking on numbers on a keypad on screen instead of typing them on the keyboard offer only limited extra security. All that is required is a slightly more sophisticated key-logger or Trojan on the victim's computer (or that in a web-café, for example). These exist 'in the wild' and target mouse clicks by taking screen shots of the user's activities for subsequent analysis.

Security by omission

"If you don't see this picture you're on a bad site"

Mechanisms where a picture or phrase chosen by the user is displayed as a watermark on a site are becoming more common. The rationale is that the user is trained such that they should only enter their credentials on a site displaying the watermark.

As with 'n from m' schemes, this requires the user to notice if the picture is missing or incorrect on a site, which with so many different security schemes in daily use may simply be too difficult. Indeed, a recent Harvard/MIT study cited by the New York Times indicated that over 90% of subjects in their trial failed to notice the absence of a security image on their Internet banking site.



Application Security Trends

<http://usablesecurity.org/emperor/>

http://www.nytimes.com/2007/02/05/technology/05secure.html?_r=1&oref=login

One Time Password (OTP) / token authentication

Some institutions are now accepting the risk is not one that can be mitigated sufficiently by software methods, and are supplying strong two-factor authentication (2FA) to clients.

At the present time, this use of 'something you have, something you know' strong authentication is the best compromise between security, cost and practicality for sensitive sites, but is not suitable for all.

Even 2FA solutions are not invulnerable to attack. Man-in-the-middle, session hijack and complex social engineering attacks can still be used to compromise users. In defence of his bank's position, one senior official indicated that the use of 2FA was pointless, as it simply diverted attackers to those with weaker security (despite having 2FA in use within some client facing divisions of the bank).

Conclusions

Is application security improving?

The short answer is 'yes and no'.

It is important to note that there is no 'one size fits all' solution to application security and absolutes are dangerous. User security on a social-networking site will differ greatly from a financial site – a 2FA such as that discussed above may be a suitable cost compared to the losses incurred through a breach of internet banking, but is probably not economical compared to prevent someone making false blog entries. A retail site that is security conscious, is keen to keep the user on the site and indeed help them access it through automated password recovery, while a bank is more likely to require you contact the call-centre.

While security is now often a consideration during the application development process, it is still too frequently considered a non-essential, costly, difficult and expensive factor that is too easily pushed aside in order to meet deadlines, costs or other project milestones.

The increasing complexity of applications makes security harder to implement, particularly if left as an afterthought.



Application Security Trends

In many cases the risks aren't truly understood. The arguments of 'we have a firewall and IPS' are too often offered as mitigation against application insecurities, whilst these are clearly useful elements in a security infrastructure, they cannot replace the assurance associated with a well-implemented secure application for many reasons; including lack of visibility of SSL encrypted data streams, undefined and unknown attacks and vulnerabilities due to logic problems in the application. Those reasons are beyond the scope of this paper (and have been discussed extensively elsewhere).

The solution?

Again, the short answer is 'develop secure applications', but in order to achieve this there are a number of steps that should be taken.

- Define and mandate security in the definition and design stages.
- Educate developers about application security (and insecurity).
- Factor security into all development phases.
- Test security during development.
- Validate security during QA.
- Regularly test security in live environments.

Considering and mandating security throughout the design, build and maintenance processes ensures it may be cost-effectively and successfully integrated into all applications.



Application Security Trends

Appendix A

References

Secure Development Framework – <http://research.corsaire.com/whitepapers/technical.html>

Application Level DoS Attacks – <http://research.corsaire.com/whitepapers/technical.html>

Acknowledgements

This Guide was written by Glyn Geoghegan, Principal Consultant at Corsaire.

About The Author

Glyn Geoghegan is a Principal Consultant in Corsaire's Professional Services team and heads up the company's Australian division. He has worked in IT Security and Internet related roles for eleven years since graduating in 1996 and has specialised in security design, deployment and analysis since 1997. He has spent the last five years focused on consultancy, assessment and audit at Corsaire and previously at Internet Security Systems where he was co-founder of their EMEA X-Force Security Assessment Services team and lead author on the ISS Ethical Hacking Training course.

Glyn's past roles have included that of Senior Network Security Analyst at a leading City of London Financial institution and as an Internet and Security Consultant at a global ISP. At both positions he was involved in corporate security at many levels and was responsible for consulting on the paper security policies and procedures, conducting vulnerability assessments, designing, deploying and managing the security infrastructure of the organisation.

Glyn is cleared by GCHQ to SC level and holds numerous accreditations including CISA and CISM and various vendor certifications. He has an honours degree in Mathematics and Computer Science from Imperial College, London.

About Corsaire

Corsaire are experts at securing information systems, consultancy and assessment. Through our commitment to excellence we provide a range services to help organisations protect their information assets and reduce corporate risk.

Founded privately in the United Kingdom in 1997, we operate on an international basis with a presence across Europe and the Asia-Pacific rim. Our clients are diverse, ranging from government security agencies



Application Security Trends

and large blue-chip FTSE, DAX, Fortune 500 profile organisations to smaller internet start-ups. Most have been drawn from banking, finance, telecommunications, insurance, legal, IT and retail sectors. They are experienced buyers, operating at the highest end of security and understand the differences between the ranges of suppliers in the current market place.

For more information contact us at info@corsaire.com or visit our website at <http://www.corsaire.com>