

Cryptography in the Real World

Daniel D. Houser, CISSP-ISSAP, CISM, CITP

Sr. Security & Identity Architect
Enterprise Architecture
Cardinal Health

Cardinal Health

- \$86 Billion Revenue
- 19 on Fortune 500
- 51 on Global 500
- Fortune #1 Most Admired firm in our sector
- 360+ facilities in 90 countries
- Founded in 1971
- Headquartered in Dublin

We're hiring!

Each day, Cardinal...

- Makes 50,000 deliveries to 40,000 sites
- Manufactures 4+million medical/surgical products
- Helps caregivers dispense 8 million doses of medicine
- Products used in 50% of surgeries, 90% of hospitals

Diverse!

- Supply Chain Services
- Medical Products Manufacturing
- Nuclear & Specialty Pharmacy Services
- Alaris[®] - Market-leading IV infusion systems
- Pyxis[®] - Market-leading hospital automation
- MedMined – real-time clinical data mining

Disclaimer

This presentation doesn't present any real-world cryptographic implementations at Cardinal Health, nor does this presentation represent statements of Cardinal Health policies or engineering regarding cryptography.

Cryptography is tricky. You should do it very carefully, and this presentation isn't a complete guide to avoid being burned by cryptography. Hire an expert.

Cryptography in the Real World

- Real-world cryptographic implementations, and lessons learned.
- Balancing security with expediency to deliver "good enough" crypto
- Real cryptanalysis efforts, and what gaps in design principles led to compromises
- Pragmatic view of cryptography
- Crypto implementations aren't always pretty

Interactive forum – ask questions

Cryptography is Math

- Pretty formulas
- Theoretical, Logical
- Sound Premises lead to proofs after proofs
- Unassailable conclusions on paper
- Digital
- 4000 years old
- Means of turning coffee into tenure

11110000 XOR 00001111 =

Information Security is Engineering

- Excess of overlapping measures, controls sufficient to overwhelm...
 - Determined attackers
 - Nature / Acts of God
 - Idiots
- Layered defenses
- Dozens of years old
- Often poor tools, indetermine environment
- Constant change
- Engineers think they can solve for X

Information Risk Mgmt is Analog

- Even younger science
- Rough Economic Models
- ALE, RAROC
- Probabilistic Losses
- People issues
- Balancing act
- Imperfect measures



Cryptography in Context

Perfect math wrapped in imperfect algorithms, implemented by lazy ADD developers through human processes, flawed testing and low-bid projects to achieve forecasted sufficiency, and maintained by untrained, overworked admins on buggy, general-purpose OS.

- From Deterministic to Probabilistic
- From Digital to Analog
- From Perfect Math to Perfect Mess
- Yields some Surprises

SSL is weak

- SSL is not the magic elixir
- Overused because it checks the box

SSL Accomplishes Two Pragmatic Things:

1) Server Side Authentication

- But only server-side authentication 99% of the time
- So... why doesn't it solve phishing?

2) Link to link confidentiality

- Control where there is NO attack
- Zero provable ROI (return on investment)
- Lawyer Repellent
- Private keys are almost always in the clear

SSH has a fundamental weakness

- SSH is a GREAT protocol, but...
 - Key management is required
 - In an enterprise, distributing digital certificates to all SSH servers and clients would be onerous
 - System admins are too lazy to validate SSH keys

```
The authenticity of host 'example (192.168.1.2)' can't be established.  
RSA key fingerprint is 6f:8c:47:bf:63:5f:e2:fb:80:5b:48:1a:db:81:cc:34.  
Are you sure you want to continue connecting (yes/no)?
```

- However, if SSH keys fingerprints aren't validated, then man-in-the middle attacks and spoofing can be used to grab admin passwords.
- You should require admins to distribute and keep a log of validated hashes.

DES = 3DES = AES

“If the use of DES is your weakest control, then your site is very secure indeed” - *Bill Murray*

We spend way too much time on protocols, not nearly enough on:

- Key controls and key management
- Key change/exchange procedures
- Cryptographic toolkits
- Random number/seed generators
- Process & documentation
- Training

Nobody Brute Forces Crypto

“We always cheat. We never go after the algorithm. We always go after the implementation, and it works.” - *Ben Jun, VP Cryptography Research*

- Cryptosystems fail due to implementation flaws
- Brute force is too expensive
- Only NSA uses brute force
- WEP – failed due to implementation, not RC4
- CSS – failed implementation, cleartext key
- Enigma – bad practices, poor key choice

Key Strength is Overrated

- Arguing over 1024, 1536, 2048 and 4096 bits...
- Is somewhat like arguing about the number of pins in the cylinder of the lock on your tent.
- Thieves don't pick locks.
- Again, much more important to worry about:
 - Key management
 - Key exchange protocols
 - Avoiding key re-use
 - Cryptosystem DRP/BCP
 - Repeatable, documented processes
 - Training of crypto personnel

Crypto doesn't erode

- Engineering for key length focuses on:
 - Brute force work required
 - Moore's Law
- However, cryptosystems don't erode, they collapse catastrophically
- Remember, brute force is last resort
- So, why do we establish life of keys based on Moore's Law? Because it's easy. It's also wrong.
- Shouldn't the life of the cryptosystem be based on risk management?

Crypto is about Identity

- Almost all crypto addresses identity management issues...
 - Who are you?
 - Who can access this file?
 - Who can update the file?
 - Has an unauthorized change occurred?
 - Did this really come from you?
- Without solid identity management, you can't implement solid cryptography
- Example: Sending OTP tokens through US Mail that were requested by Hotmail account user

Most of our keys are English

...or are protected by English passwords.

- Laptop encryption
 - Windows EFS
 - PGP keys
 - Unix/Windows passwords
 - Service execution passwords
 - Sample code dropped into your source code
-
- Likely dictionary words
 - English has high entropy – 0.8 bits per byte
10 character password provide 8 bit crypto
 - Far easier to guess and brute force
 - Theoretical vs. actual strength

Databases should not be encrypted

- Database encryption only buys you one thing: protection from theft of server disk. No ROI
- There are very few fields that should be encrypted in a database – password is a good example
- Encrypt rows, not tables
- Protect the database with access controls
- Protect the server with data center access controls
- Protect backup tapes with a password and physical access controls

People buy bad crypto

- Regulatory pressure is pushing us in the wrong direction to check the box
- “Project ABC will implement Cryptonite 7.1”
- Check the box, problem solved, we're done
- However, have we improved our capabilities?
- Built on a shaky foundation
 - Focus on building capabilities
 - Start with the basics: Key management, cryptographic use policies, information classification, identity management, cryptographic governance, crypto team

PKI vs. PKE

- PKI is usually too much
- Identity-based PKI is really difficult
- PKE = “Public Key Enough”
 - Enough crypto to get the job done
 - Don't seek a perfect engineering solution
 - Tactical, infrastructure PKI is good start
 - Leverage existing PKI (Notes, AD, LDAP)

Crypto Example

Demo site



[Openwall Project](#)
bringing security into open environments

[/home Owl JtR Pro crypt pam passwdqc tcb phpass scanlogd pop](#)
[advisories presentations / services donations / wordlists passwords / si](#)

John the Ripper password cracker

John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), Windows, OpenVMS. Its primary purpose is to detect weak Unix passwords. Besides several crypt(3) password hash types most commonly found on various Unix flavors, it also cracks Kerberos AFS and Windows NT/2000/XP/2003 LM hashes, plus several more with contributed patches.

Advanced [Word](#), [Excel](#), [Access](#), [PDF](#) ... Password Recovery

[passwords.openwall.net](#)
[/passwords/](#)

Archives:
[ZIP](#), [RAR](#), [ACE](#), [ARJ](#)

Microsoft Office:
[MS Word](#), [Excel](#),
[Access](#), [Project](#), [VBA](#)

Microsoft Internet Explorer,
[Outlook Express](#), [Outlook](#),
[Internet Mail](#),
[Money](#), [Backup](#)

[Adobe Acrobat PDF](#)

Corel WordPerfect Office:
[WordPerfect](#), [QuattroPro](#),
[Paradox](#)

John the Ripper is free and Open Source software, distributed primarily in source code form. If you would rather use a commercial product tailored to your system, please consider [John the Ripper Pro](#), which is distributed primarily in the form of "native" packages for the target operating systems and is easy to install and use while delivering optimal performance.

Proceed to the John the Ripper *Pro* homepage for your OS:

- ◆ [John the Ripper 1.7.2 Pro for Linux](#)
- ◆ [John the Ripper 1.7.2 Pro for Mac OS X](#)

Download the latest free "development" version:

- ◆ [John the Ripper 1.7.2 \(Unix - sources, tar.gz, 790 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)

or the latest free "stable" release:

- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.gz, 784 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(Windows - binaries, ZIP, 1360 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(DOS - binaries, ZIP, 895 KB\)](#) and its [signature](#)

Note: a few Windows "antivirus" and "anti-spam" products have started to recognize password recovery tools as if they were "trojans". This is how they are



Openwall Project
bringing security into open environments

[/home Owl JtR Pro crypt pam passwdqc tcb phpass scanlogd popo](#)
[advisories presentations / services donations / wordlists passwords / si](#)

John the Ripper password cracker

John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), Windows, OpenVMS. Its primary purpose is to detect weak Unix passwords. Besides several crypt(3) password hash types most commonly found on various Unix flavors, it also supports Kerberos AFS and Windows NT/2000/XP/2003 LM hashes, plus several more with contributed patches.

Advanced [Word](#), [Excel](#), [Access](#), [PDF](#) ... Password Recovery

[passwords.openwall.net](#)
[/passwords/](#)

Archives:
[ZIP](#), [RAR](#), [ACE](#), [ARJ](#)

Microsoft Office:
[MS Word](#), [Excel](#),
[Access](#), [Project](#), [VBA](#)

Microsoft Internet Explorer,
[Outlook Express](#), [Outlook](#),
[Internet Mail](#),
[Money](#), [Backup](#)

[Adobe Acrobat PDF](#)

Corel WordPerfect Office:
[WordPerfect](#), [QuattroPro](#),
[Paradox](#)

John the Ripper is free and Open S...
system, please consider [John the R](#)
to install and use while delivering o

Proceed to the John the Ripper Pr

- ◆ [John the Ripper 1.7.2 Pro f](#)
- ◆ [John the Ripper 1.7.2 Pro f](#)

Download the latest free "developr

- ◆ [John the Ripper 1.7.2 \(Unix](#)
- ◆ [John the Ripper 1.7.2 \(Unix](#)

or the latest free "stable" release:

- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.gz, 784 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(Windows - binaries, ZIP, 1360 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(DOS - binaries, ZIP, 895 KB\)](#) and its [signature](#)

Note: a few Windows "antivirus" and "anti-spam" products have started to recognize password recovery tools as if they were "trojans". This is how these pro

Security Alert

Information you exchange with this site cannot be viewed or changed by others. However, there is a problem with the site's security certificate.

- ⚠ The security certificate was issued by a company you have not chosen to trust. View the certificate to determine whether you want to trust the certifying authority.
- ✔ The security certificate date is valid.
- ✔ The security certificate has a valid name matching the name of the page you are trying to view.

Do you want to proceed?

Yes No View Certificate

... would rather use a commercial product tail...
... packages for the target operating systems and



Openwall Project
bringing security into open environments

[/home Owl JtR Pro crypt pam passwdqc tcb phpass scanlogd pop](#)
[advisories presentations / services donations / wordlists passwords / si](#)

John the Ripper password cracker

John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), Windows, OpenVMS. Its primary purpose is to detect weak Unix passwords. Besides several crypt(3) password hash types most commonly found on various Unix flavors, it also cracks Kerberos AFS and Windows NT/2000/XP/2003 LM hashes, plus several more with contributed patches.

Advanced [Word](#), [Excel](#), [Access](#), [PDF](#) ... Password Recovery

[passwords.openwall.net](#)
[/passwords/](#)

Archives:
[ZIP](#), [RAR](#), [ACE](#), [ARJ](#)

Microsoft Office:
[MS Word](#), [Excel](#),
[Access](#), [Project](#), [VBA](#)

Microsoft Internet Explorer,
[Outlook Express](#), [Outlook](#),
[Internet Mail](#),
[Money](#), [Backup](#)

[Adobe Acrobat PDF](#)

Corel WordPerfect Office:
[WordPerfect](#), [QuattroPro](#),
[Paradox](#)

John the Ripper is free and Open Source software, distributed primarily in source code form. If you would rather use a commercial product tailored to your system, please consider [John the Ripper Pro](#), which is distributed primarily in the form of "native" packages for the target operating systems and is easy to install and use while delivering optimal performance.

Proceed to the John the Ripper *Pro* homepage for your OS:

- ◆ [John the Ripper 1.7.2 Pro for Linux](#)
- ◆ [John the Ripper 1.7.2 Pro for Mac OS X](#)

Download the latest free "development" version:

- ◆ [John the Ripper 1.7.2 \(Unix - sources, tar.gz, 790 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)

or the latest free "stable" release:

- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.gz, 784 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(Windows - binaries, ZIP, 1360 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(DOS - binaries, ZIP, 895 KB\)](#) and its [signature](#)

Note: a few Windows "antivirus" and "anti-spam" products have started to recognize password recovery tools as if they were "trojans". This is how these pro...



Openwall Project
bringing security into open environments

[/home Owl JtR Pro crypt pam passwdqc tcb phpass scanlogd pop](#)
[advisories presentations / services donations / wordlists passwords / si](#)

John the Ripper password cracker

John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), Windows, OpenVMS. Its primary purpose is to detect weak Unix passwords. Besides several crypt(3) password hash types most commonly found on various Unix flavors, it also supports Kerberos AFS and Windows NT/2000/XP/2003 patches.

Untrusted Certificate

The security certificate presented by this website was not issued by a trusted certificate authority.

This problem may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage.

About certificate errors

View Certificates

Advanced [Word](#), [Excel](#), [Access](#), [PDF ...](#) Password Recovery

[passwords.openwall.net](#)
[/passwords/](#)

Archives:
[ZIP](#), [RAR](#), [ACE](#), [ARJ](#)

Microsoft Office:
[MS Word](#), [Excel](#),
[Access](#), [Project](#), [VBA](#)

Microsoft Internet Explorer,
[Outlook Express](#), [Outlook](#),
[Internet Mail](#),
[Money](#), [Backup](#)

[Adobe Acrobat PDF](#)

Corel WordPerfect Office:
[WordPerfect](#), [QuattroPro](#),
[Paradox](#)

John the Ripper is free and Open source system, please consider [John the Ripper](#) to install and use while delivering

Proceed to the [John the Ripper](#) F

- ◆ [John the Ripper 1.7.2 Pro](#)
- ◆ [John the Ripper 1.7.2 Pro](#)

Download the latest free "develop

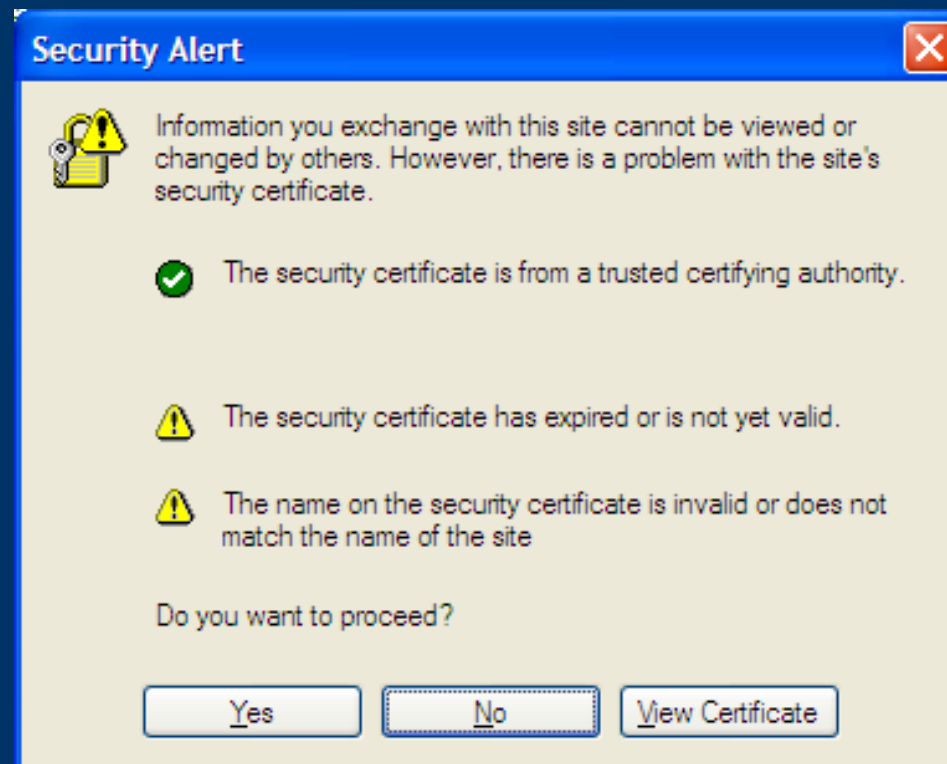
- ◆ [John the Ripper 1.7.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)

or the latest free "stable" release:

- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.gz, 784 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.2 \(Unix - sources, tar.bz2, 675 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(Windows - binaries, ZIP, 1360 KB\)](#) and its [signature](#)
- ◆ [John the Ripper 1.7.0.1 \(DOS - binaries, ZIP, 895 KB\)](#) and its [signature](#)

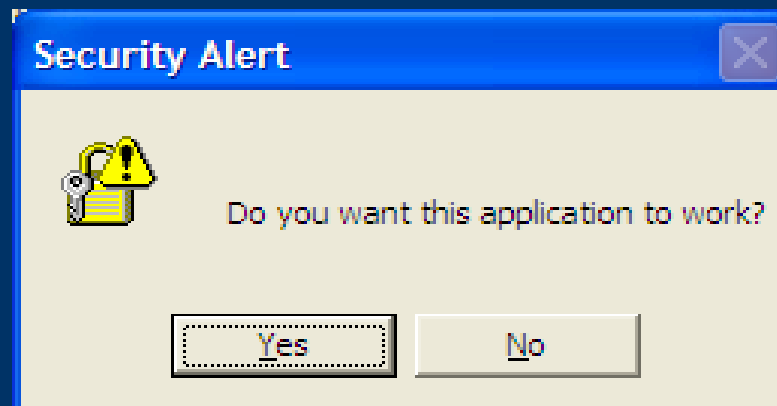
Note: a few Windows "antivirus" and "anti-spam" products have started to recognize password recovery tools as if they were "trojans". This is how these pro

Users Don't Care About Security



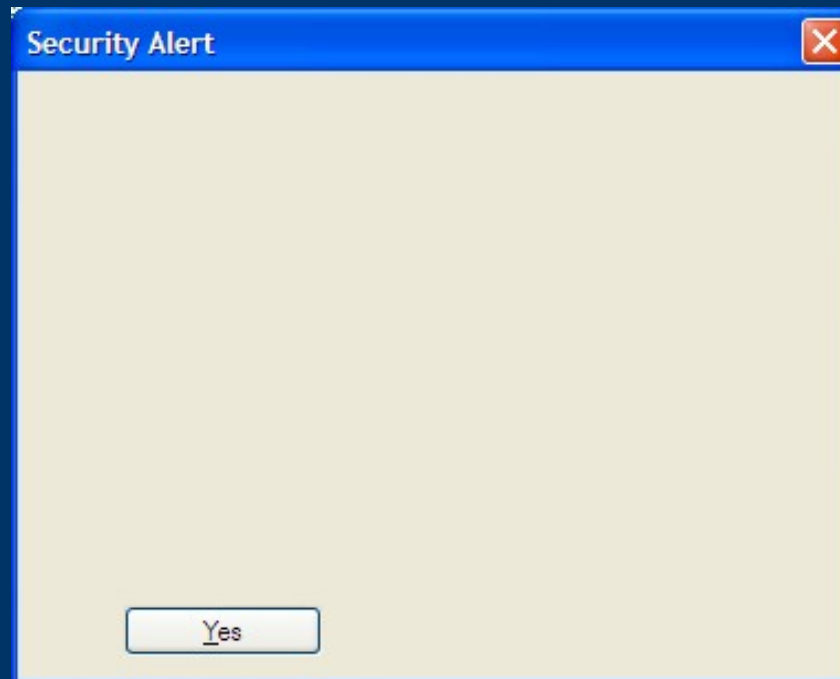
Make security decisions for end-users, fail closed.

Users Don't Care About Security



Make security decisions for end-users, fail closed.

Users Don't Care About Security



Make security decisions for end-users, fail closed.



Information you exchange with this site cannot be viewed or changed by others. However, there is a problem with the site's security certificate.



The security certificate was issued by a company you have not chosen to trust. *View* the certificate to determine whether you want to trust the certifying authority.



The security certificate date is valid.



The name on the security certificate is invalid or does not match the name of the site.

Do you want to proceed?

Yes

No

View Certificate

Self-signed Certs are Evil

- Why do we use self-signed certificates on our Intranets?
 - Cheap & economical
 - Easy
 - It's just test
- Result: Users are being trained to ignore errors
- 99+% of users will click-through certificate failure error messages
- Same issues with domain changes producing stale certificate warnings

Segregation of Duties is ROUGH

- Very difficult to achieve operationally without significant risk.
- “Hit by a bus issue”
- Operational crypto teams often with
 - Root/ administrator
 - Keymaster administrative account
 - Sensitive data access
 - Ability to substitute or pervert datastreams
- Segregation of duties would turn 3 people into 14 and dramatically reduce productivity.

Toolkits Matter. A Lot.

- There are lots of ways to screw up cryptographic implementations
- Key functions in toolkits can provide substantial armoring of cryptosystems
- Great skill to achieve comparable protection using default language libraries
- Use crypto toolkits such as Bouncy Castle, MS-CAPI and RSA BSAFE

Case study: Software key protection

Long ago in a company far, far away... We needed to use commercial language crypto tools

What we found:

- Implemented FIPS 46 – DES in ECB only
- No protection against weak keys
- No provision for PRNG, seed generation, LFSRs
- No ability to securely erase keys/sub-keys
- Key protection methodology was a Ceaser cipher
- Only hash available was SHA
- No support for key management functions

Case study: Software key protection

What we found:

- Very difficult to use keys that weren't alphabetic
- No salt or seed support without hand-rolling
- Difficult to perform XOR functions
- Very poorly documented – Largely by sample code
- Several weaknesses in sample code
- Sample code encourage poor crypto practices
- Search engine found wide use of sample code
- No real x.509 support
- Touted by vendor as strong cryptography

Cryptographers are smart people

- The converse is almost always wrong
- Just because they're brilliant at math, physics, programming or chess doesn't make them a cryptographer.
- If you're doing custom crypto, don't
- If you don't listen, then have it certified by a cryptographer.
- Cryptographers, cryptanalysts, cryptologists, and cryptographic implementers are different.
- Know the difference. Hire the right one.

Vendors use Snake Oil

- Case Study
 - Vendor one-way hash
 - Proprietary algorithm for protecting SSNs
 - Transposition cipher with algebraic transforms
 - $A' = \text{INT}(\text{ABS}(\text{mod}(\mathbf{B} + \mathbf{D}^2 - \mathbf{D}, 10)))$
 - $B' = \text{INT}(\text{ABS}(\text{mod}(\mathbf{A} * \mathbf{C}^2 * \mathbf{D} - \mathbf{G}, 10)))$
 - $C' = \text{INT}(\text{ABS}(\text{mod}(\mathbf{G}^3 * \mathbf{B}^3 - (\mathbf{D}^2 / \mathbf{A}), 10)))$
 - $D' = \text{INT}(\text{ABS}(\text{mod}(\mathbf{A} - 7, 10)))$ etc.
 - St00pid crypto
 - Mod 10 ensured it was easy to break
 - Perfect algebraic plug-n-chug using 6th grade algebra
 - My 3rd grader solved for A, B, C, D, E...
 - Only protection was knowledge of the algorithm
 - It looked smart and complex. It wasn't. He wasn't.

Snake Oil Danger Signs

- “We have this brilliant guy”
- Secret algorithms
- “We just did standard AES using standard Java libs”
- New cryptography (think “new drugs”, not “new car”)
- Won't disclose details, despite NDA and Star Chamber
- Not certified by a cryptographer
- Name dropping - “The DoD uses it!”
- Security experts, rave reviews, endorsements
- Technobabble
- “Trust us”
- Unbreakability

Snake Oil Danger Signs

- Revolutionary new concept
- “Military Grade” cipher
- Recoverable keys
- Our is better, because theirs is insecure
- Infinite keyspace/perfect security
- Provably secure using One Time Pad
- Bolt-on crypto added as a feature to an application
- GUI so easy to use, they forgot to make good crypto
- Foolproof products
- Vendors don't understand key crypto concepts

Kudos to Matt Curtin for helping me think of several more...

Out of Band Key Exchange is Vital

- Increasing sophistication in cybercrime means increased token attacks, and SSL-based attacks.
- Untrained or lazy personnel may use in-band or unvalidated key exchange methods
- Keys, once exposed, permit man-in-the-middle attacks to own previously secure communications
- FedEx shipment of CD using password-protected PKZip file is sufficient for most B2B implementations.
- Is this a likely threat? No – but cheap to remedy.

Recommendations

- Focus on capabilities, implementation, documentation & processes, not products and protocols
- Layered controls mean more than keyspace
- Use risk analysis to determine when to bring in the cryptographic hired guns
- Custom cryptosystems should be certified commensurate with risk
- Avoid vendor snake oil
- Test cryptographic operational functions
- Train your admins on key functions & protection

Q & A

- Surely, there are questions???

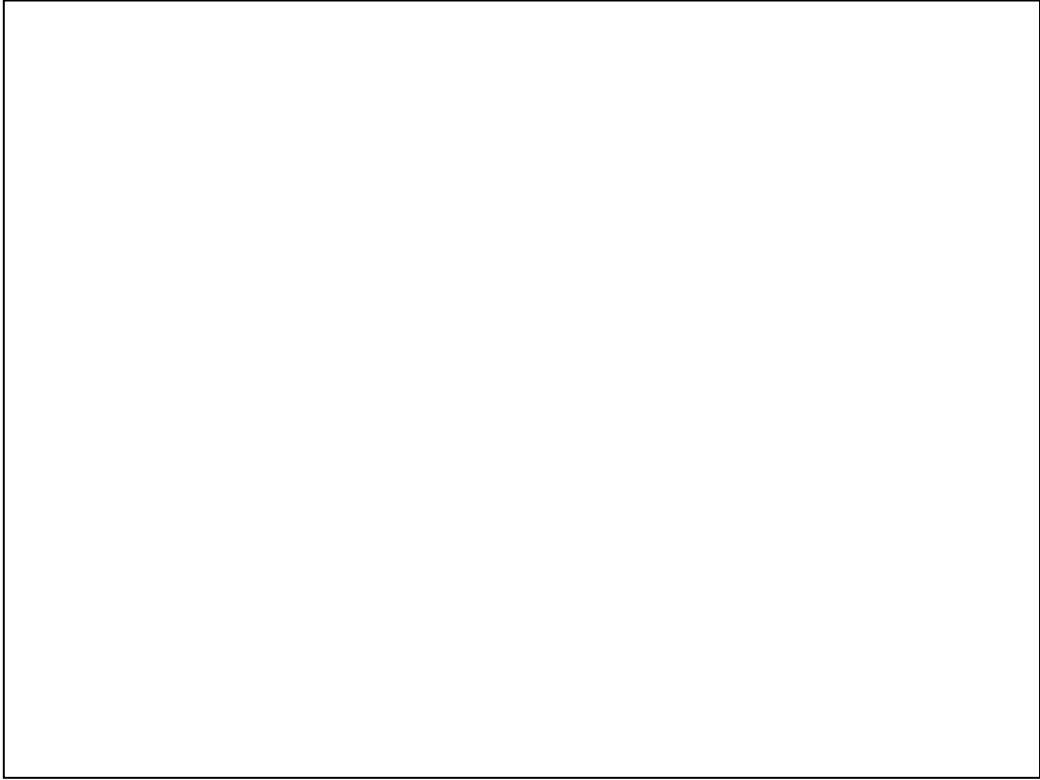
Logistics

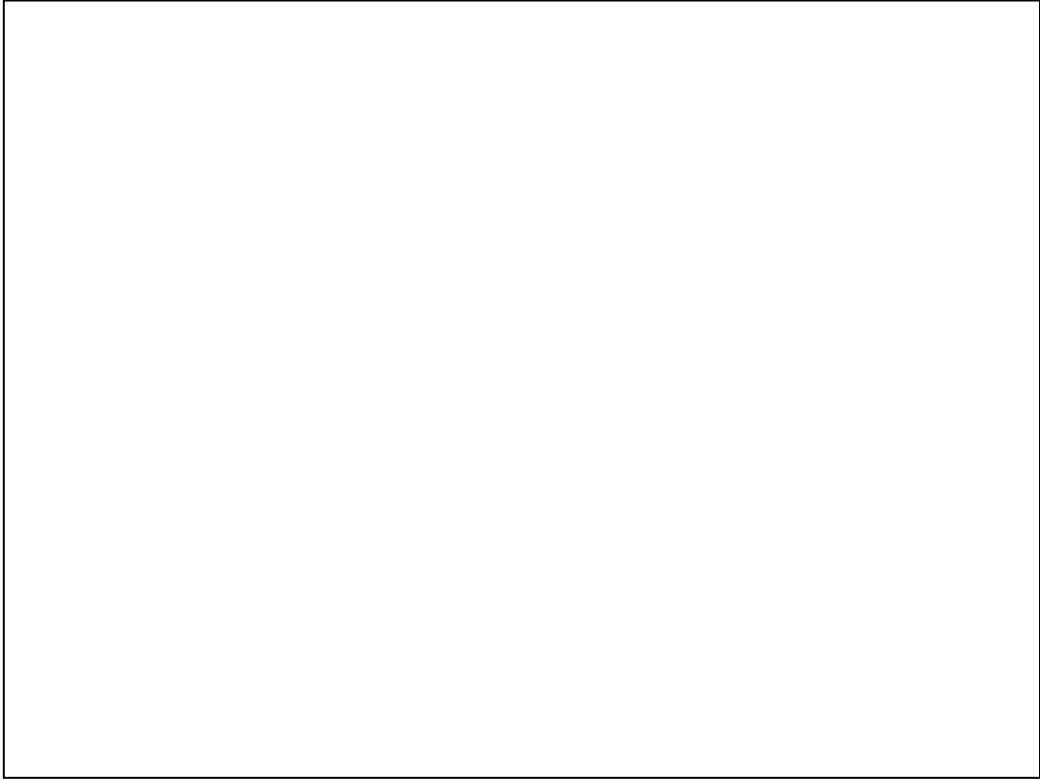
Please fill out evaluation forms

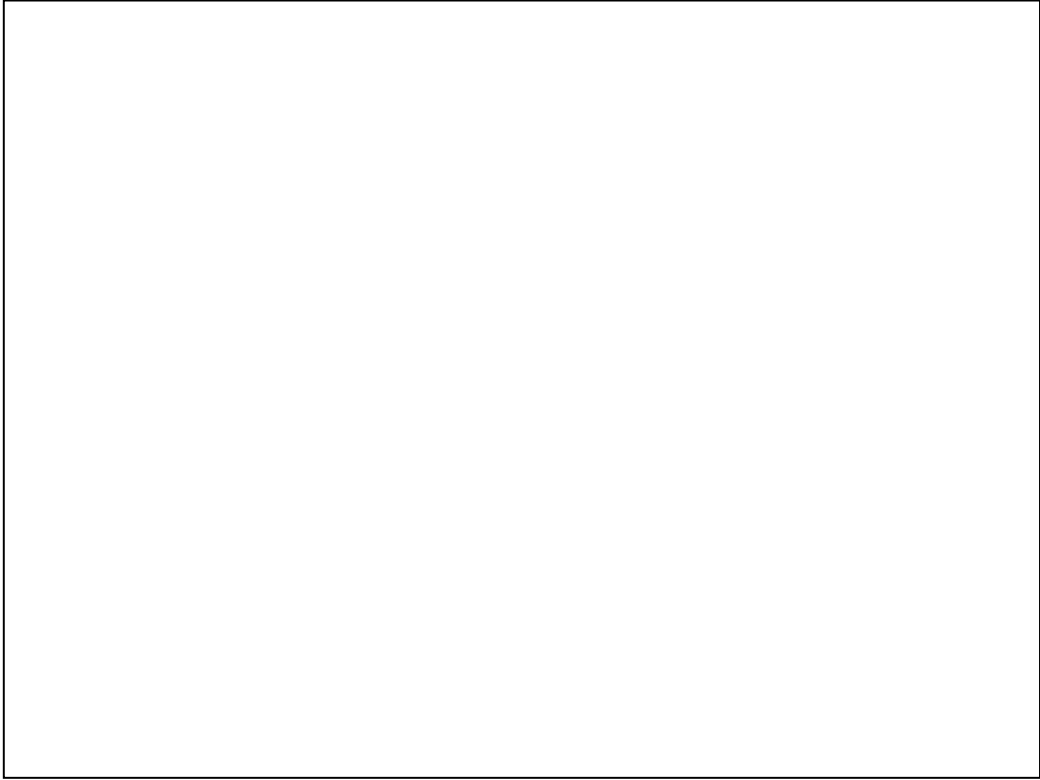
Contact information:

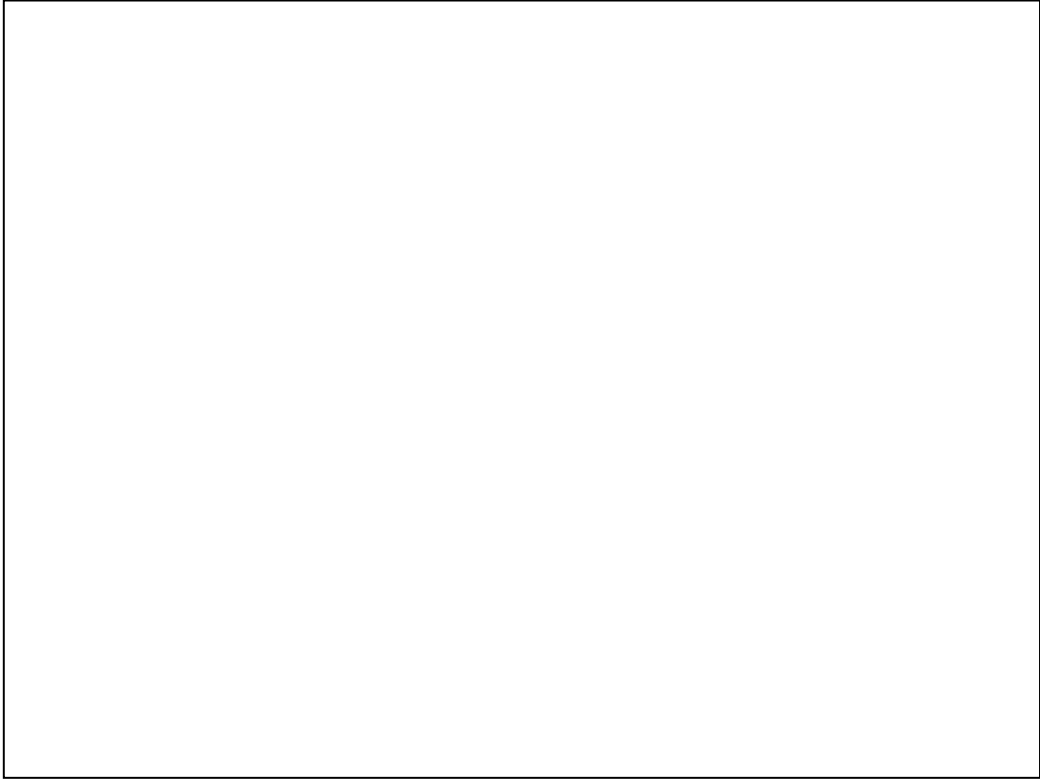
Daniel D. Houser
Sr. Security Architect
Cardinal Health

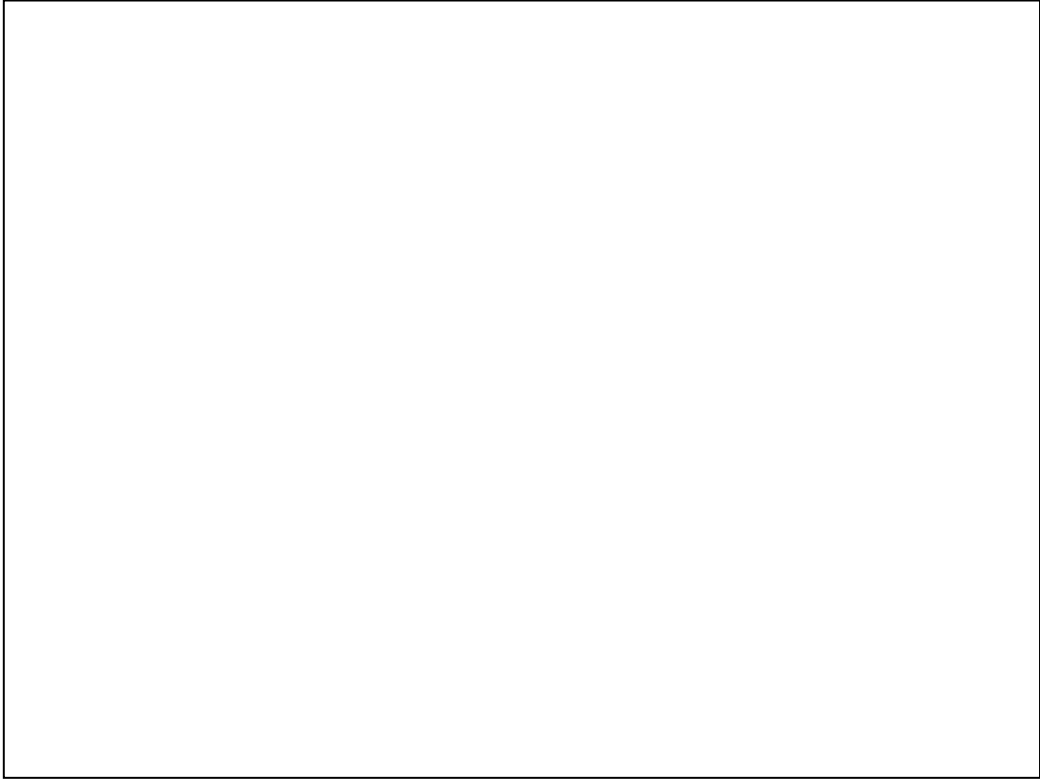
dan.houser@cardinalhealth.com

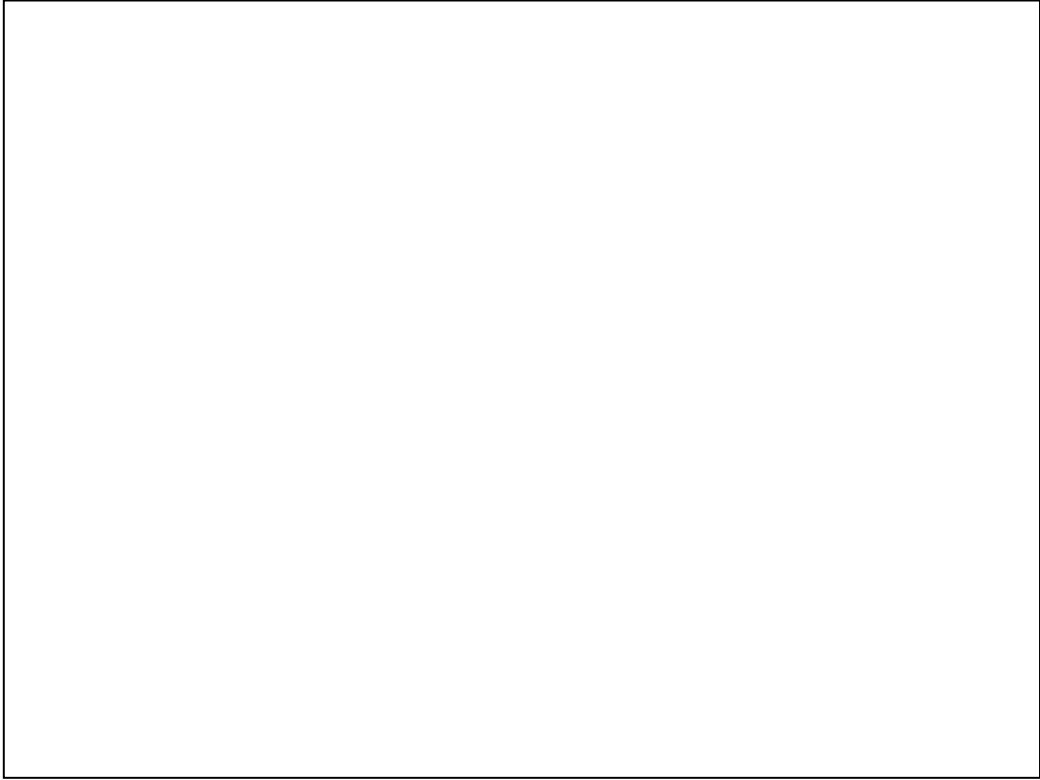


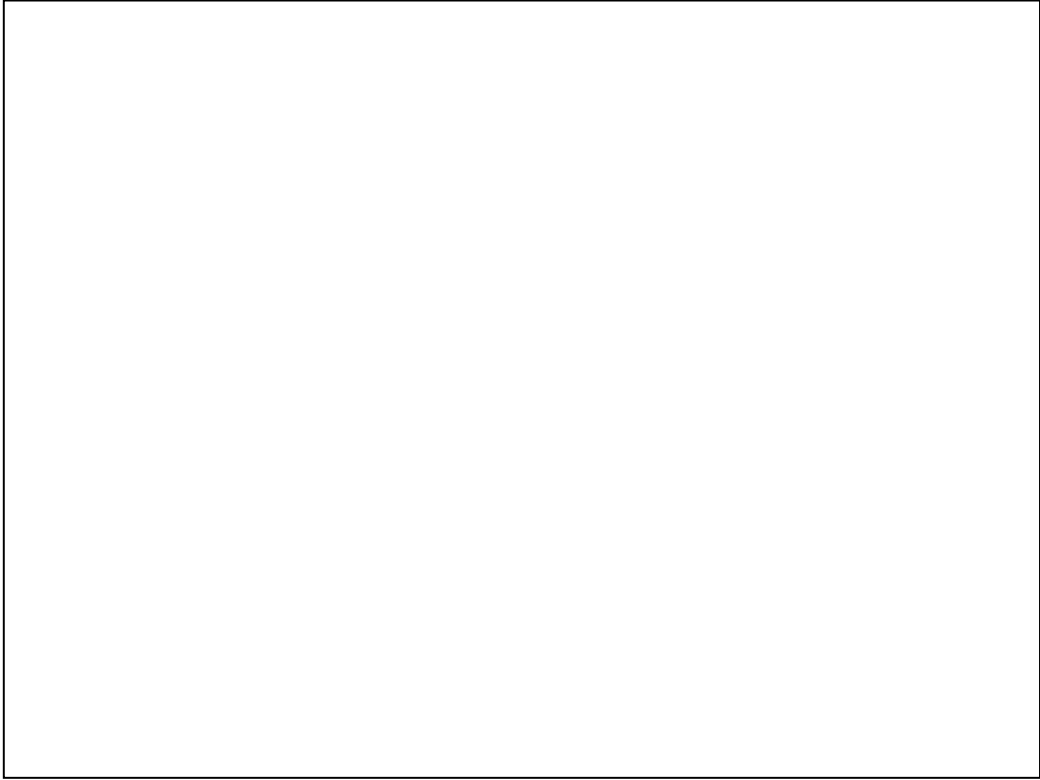


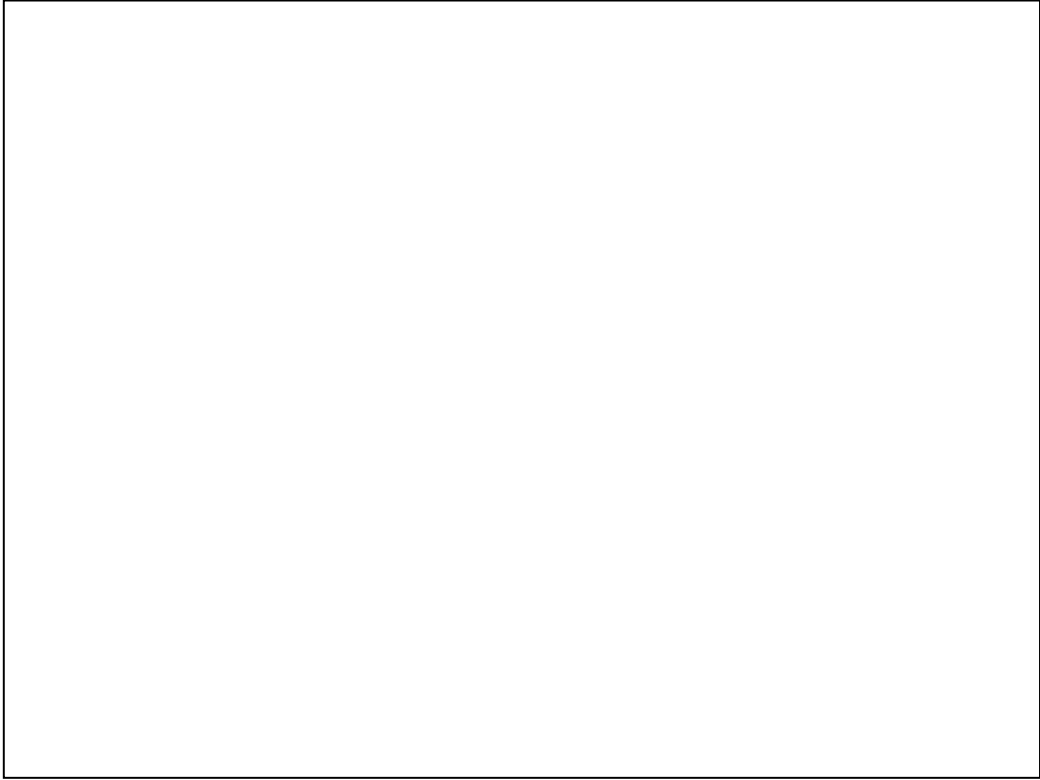


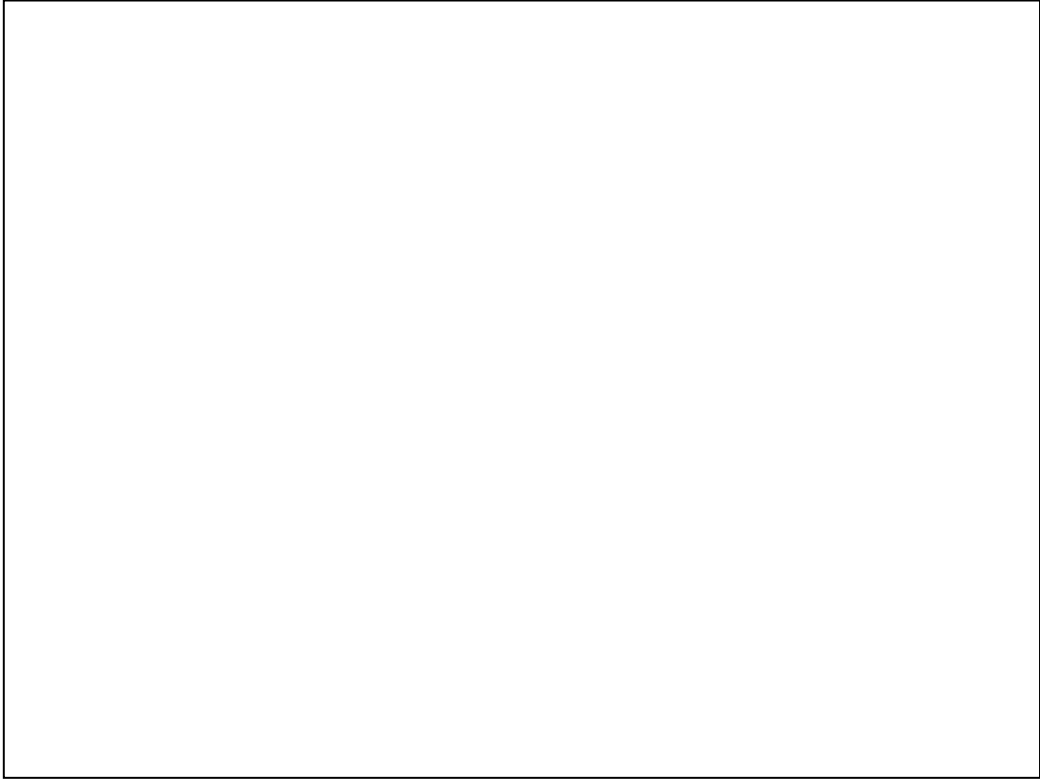


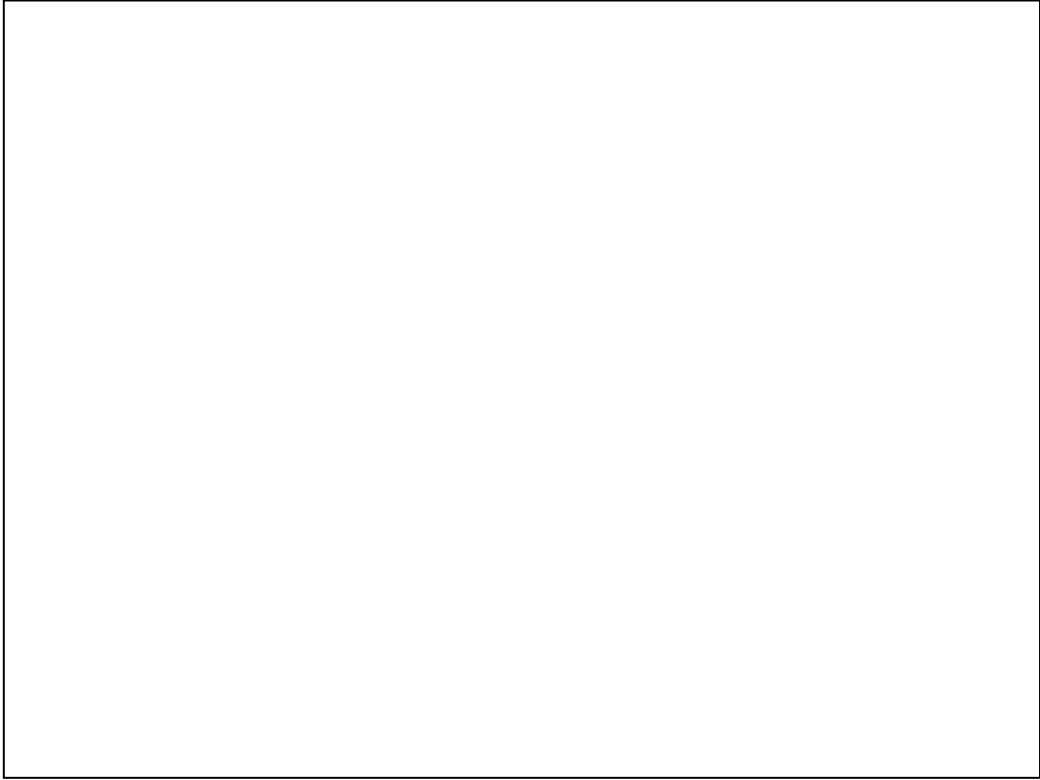


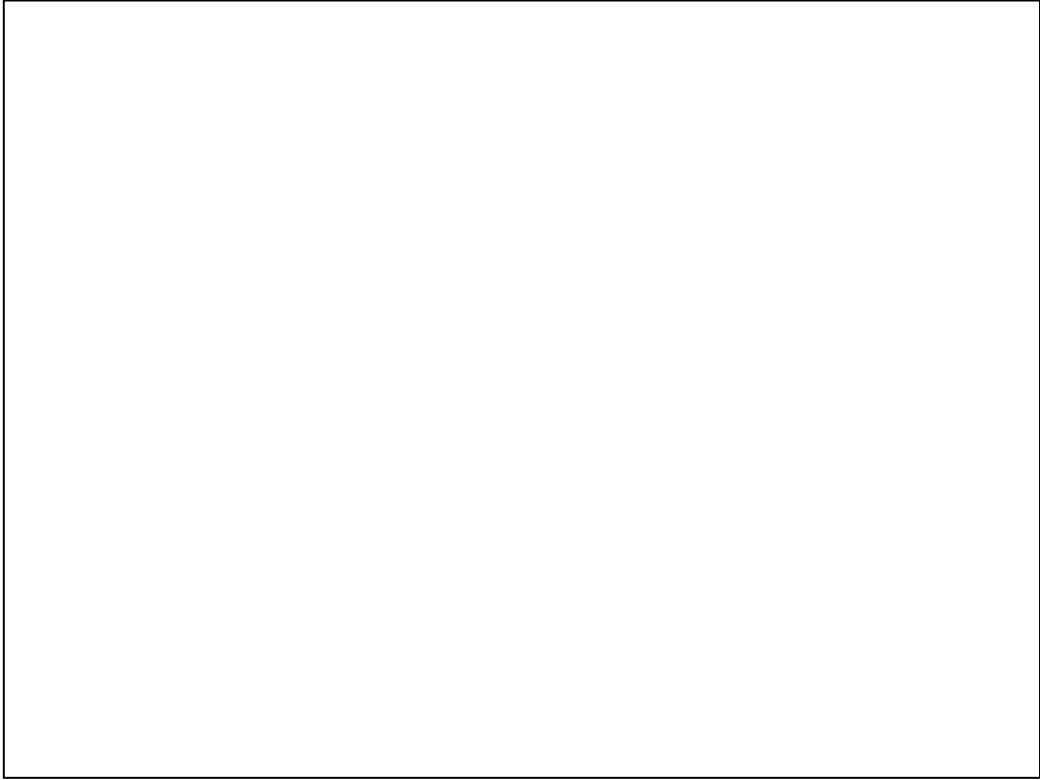


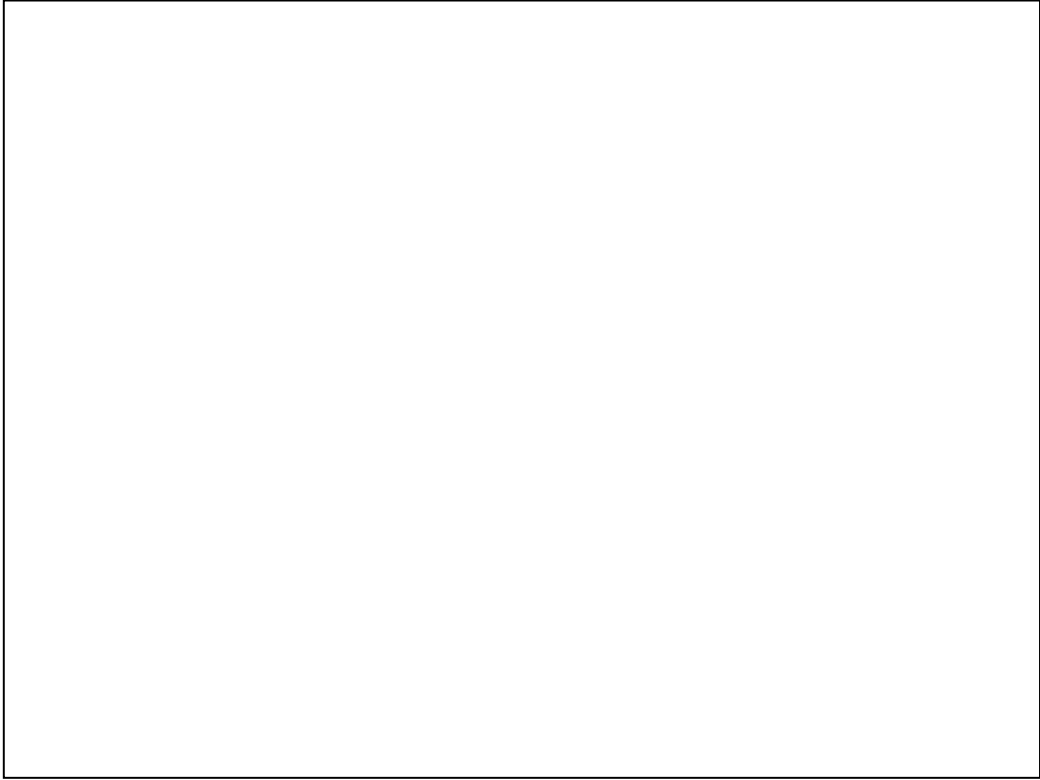


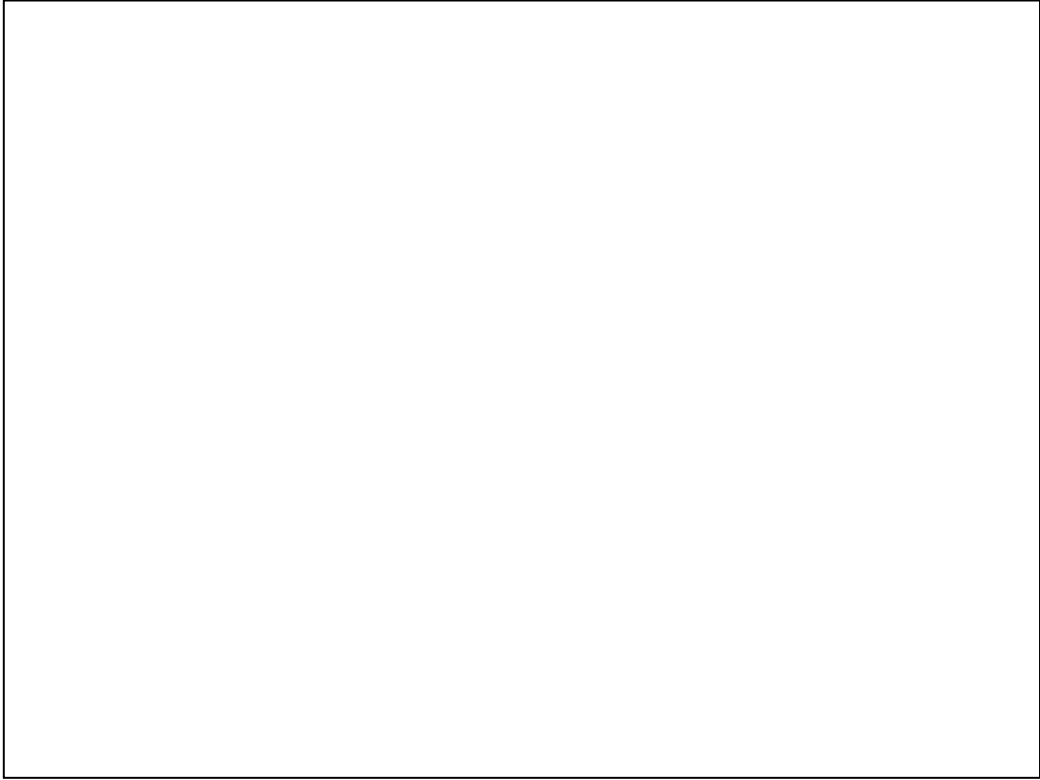


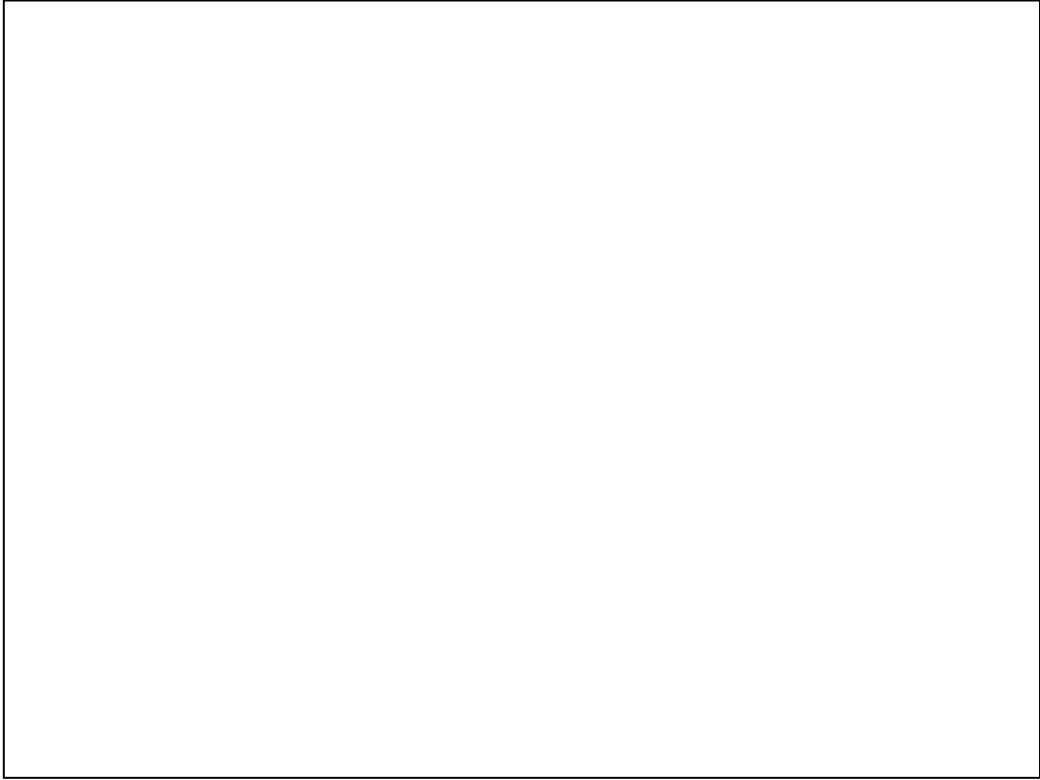


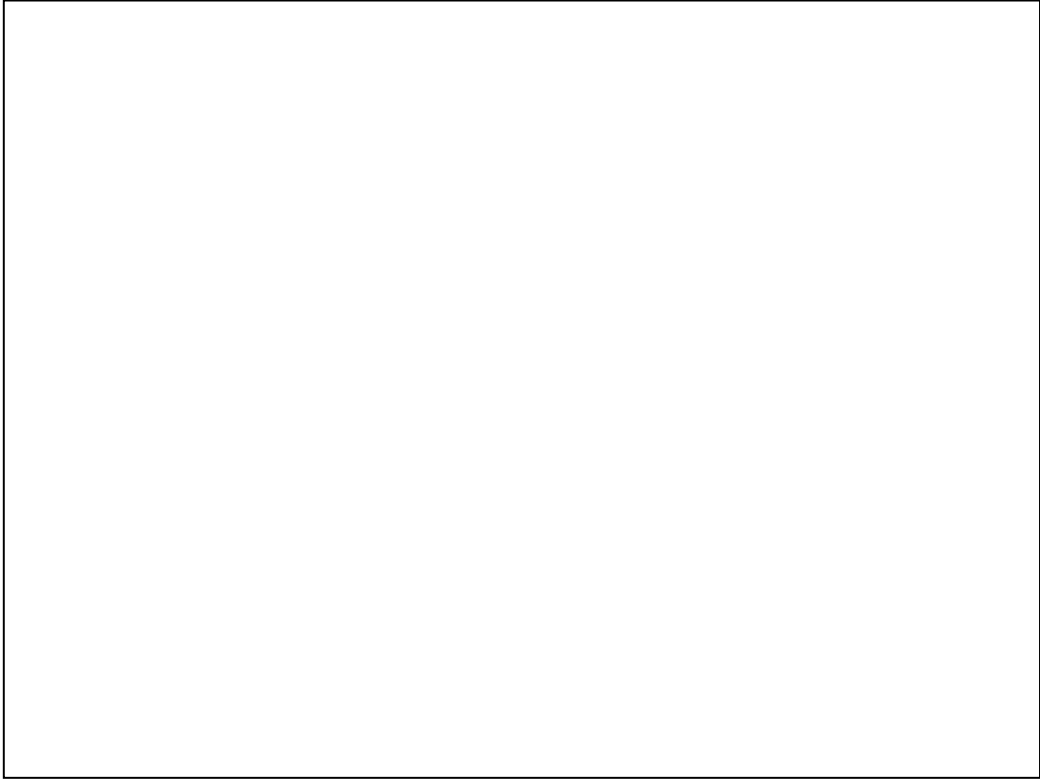


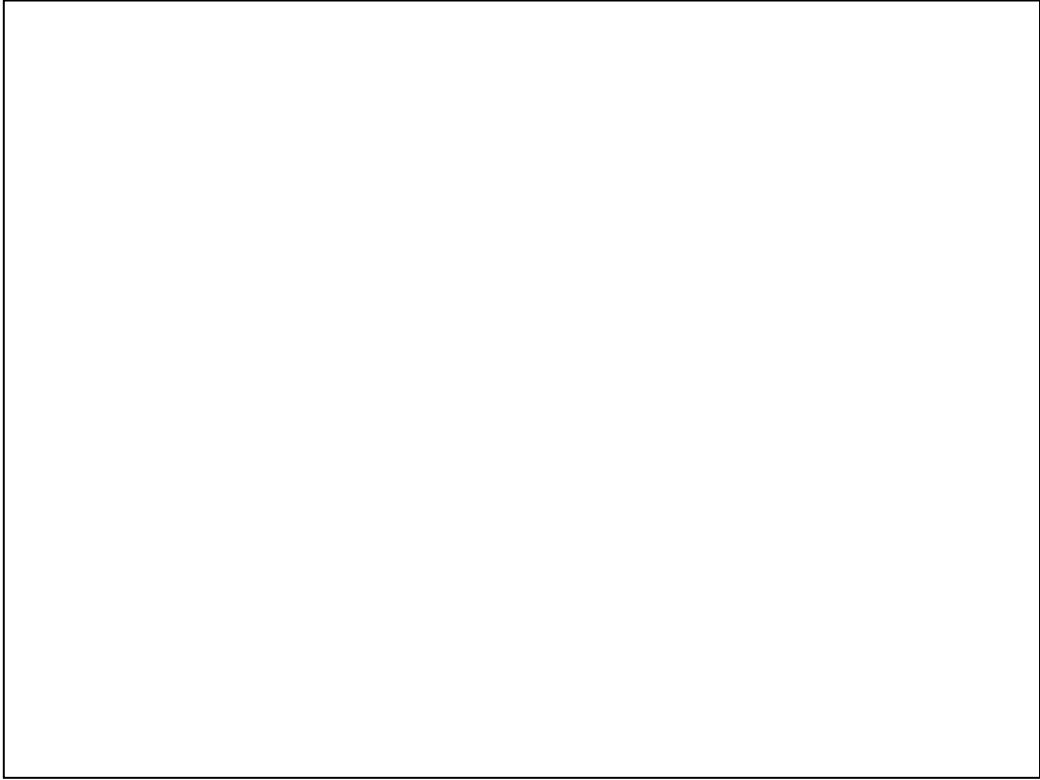


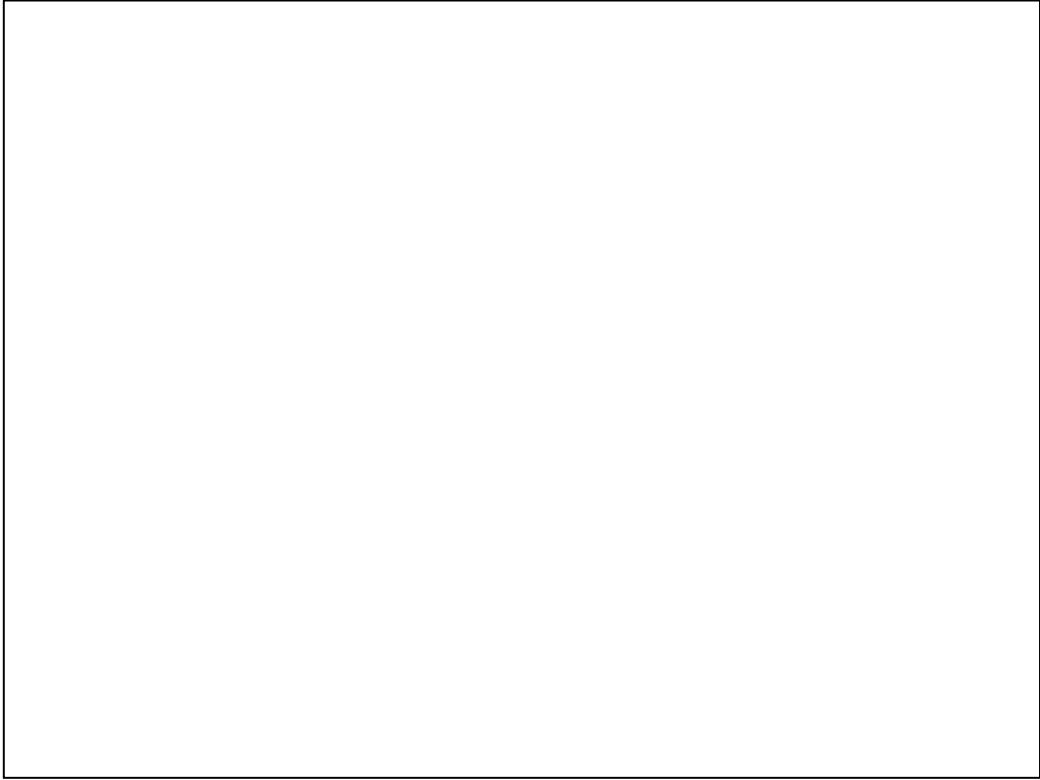


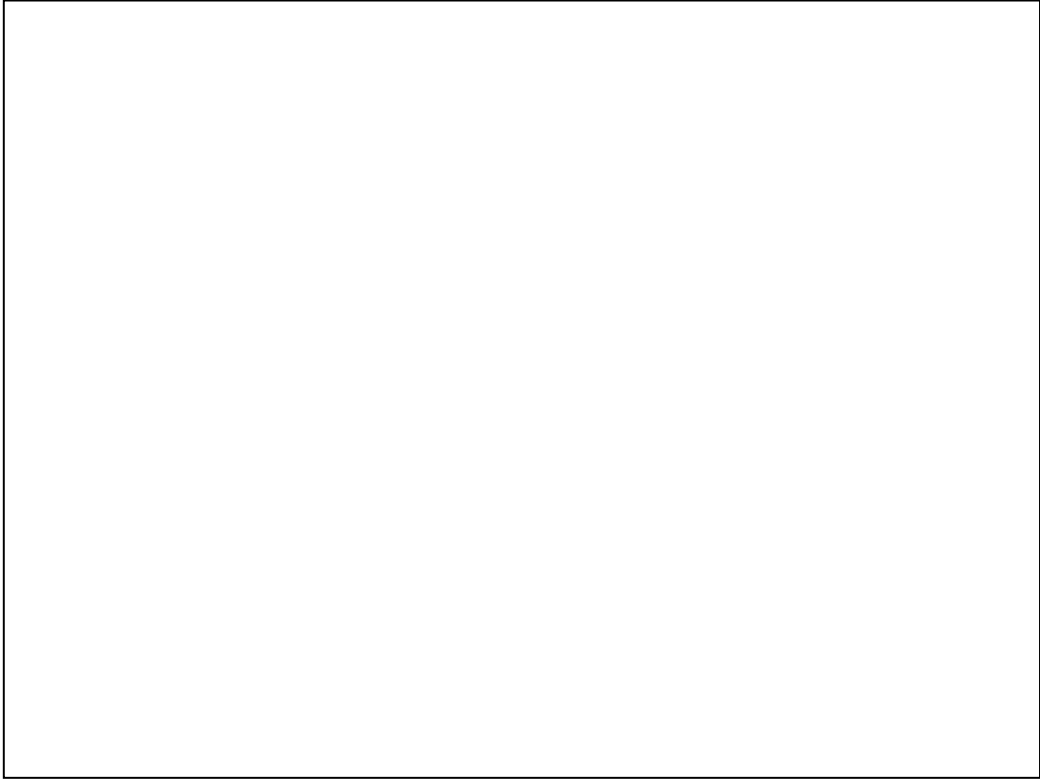


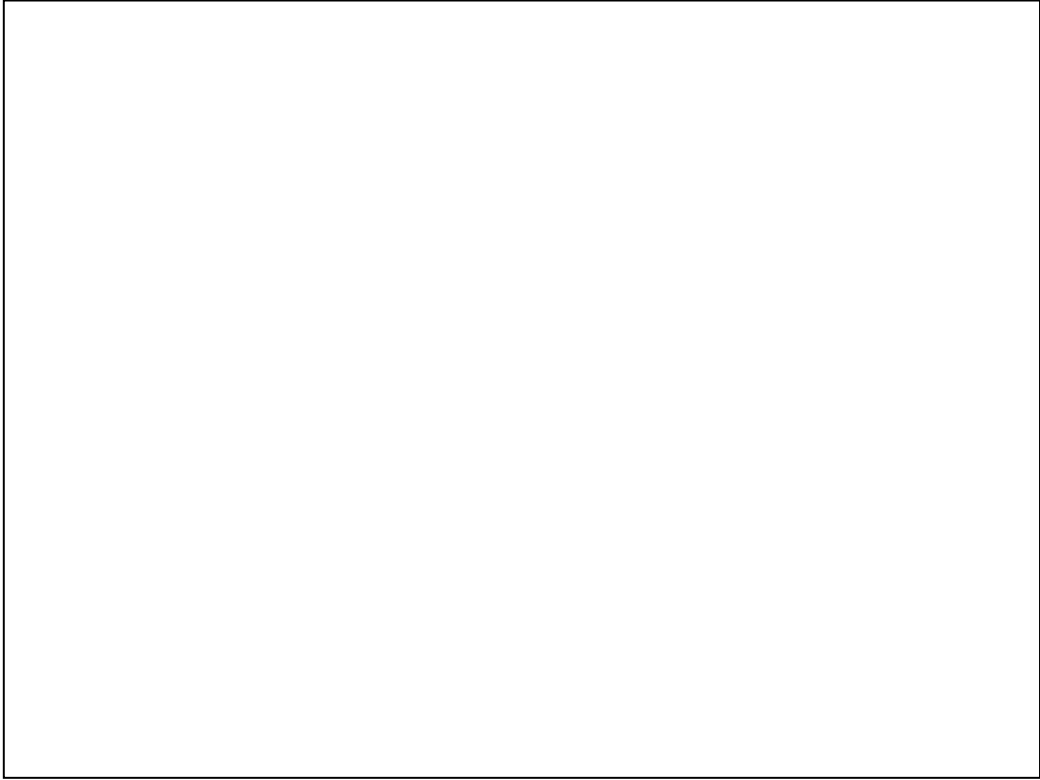


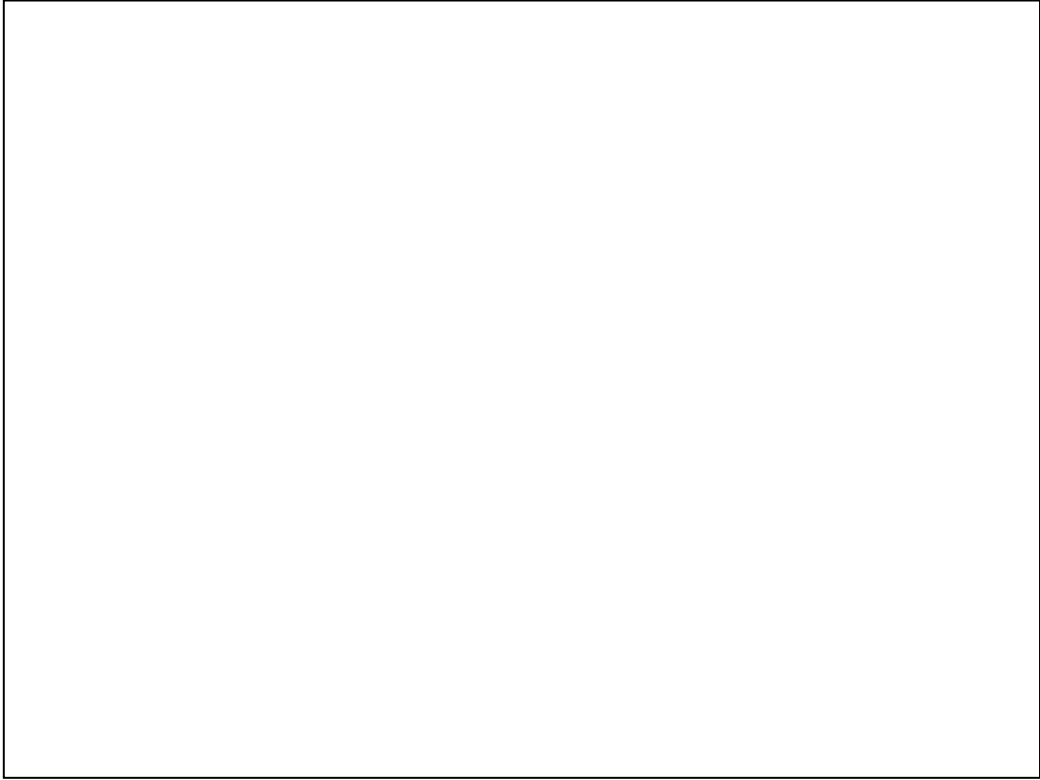


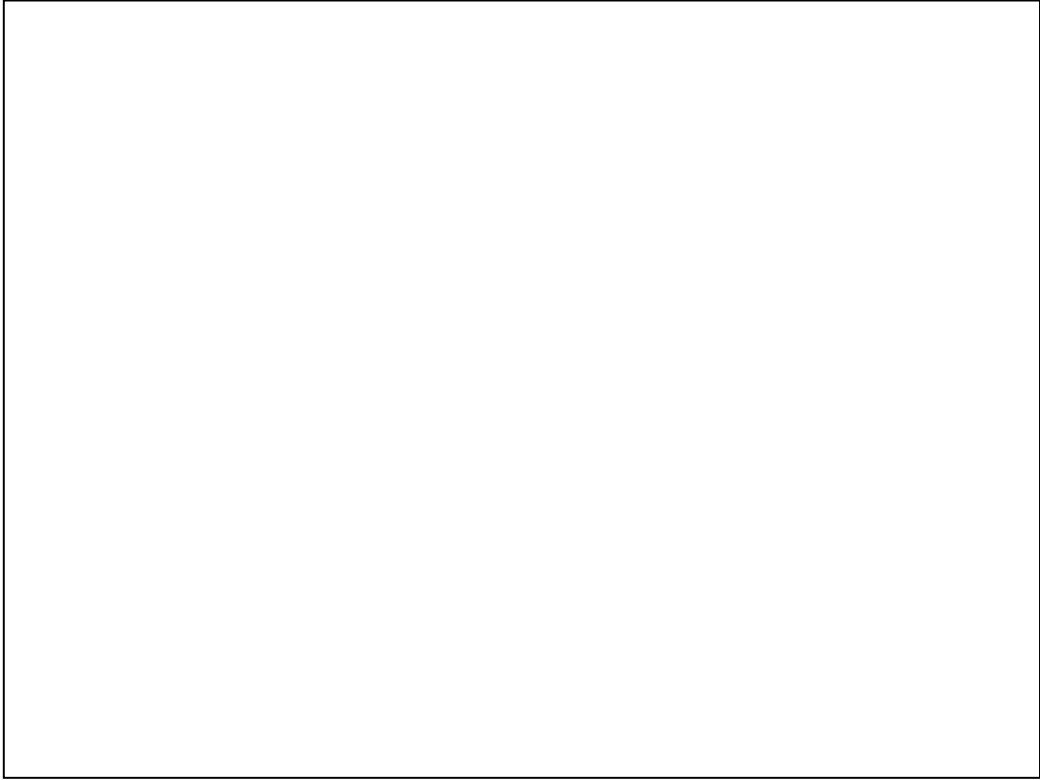


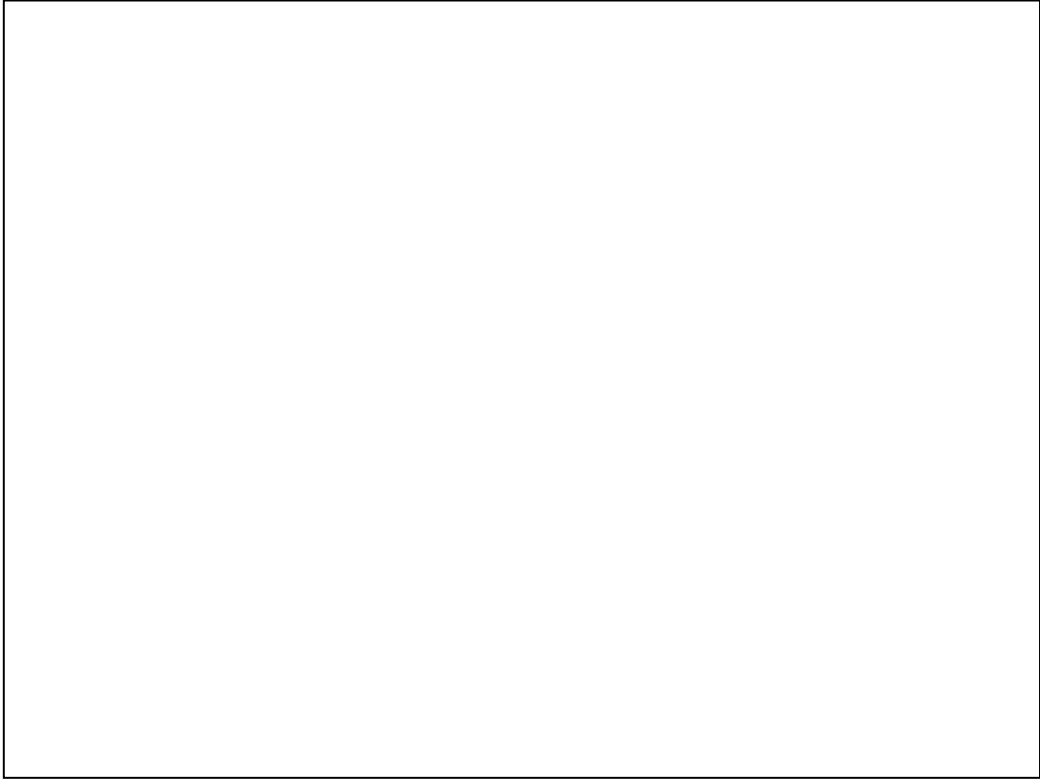


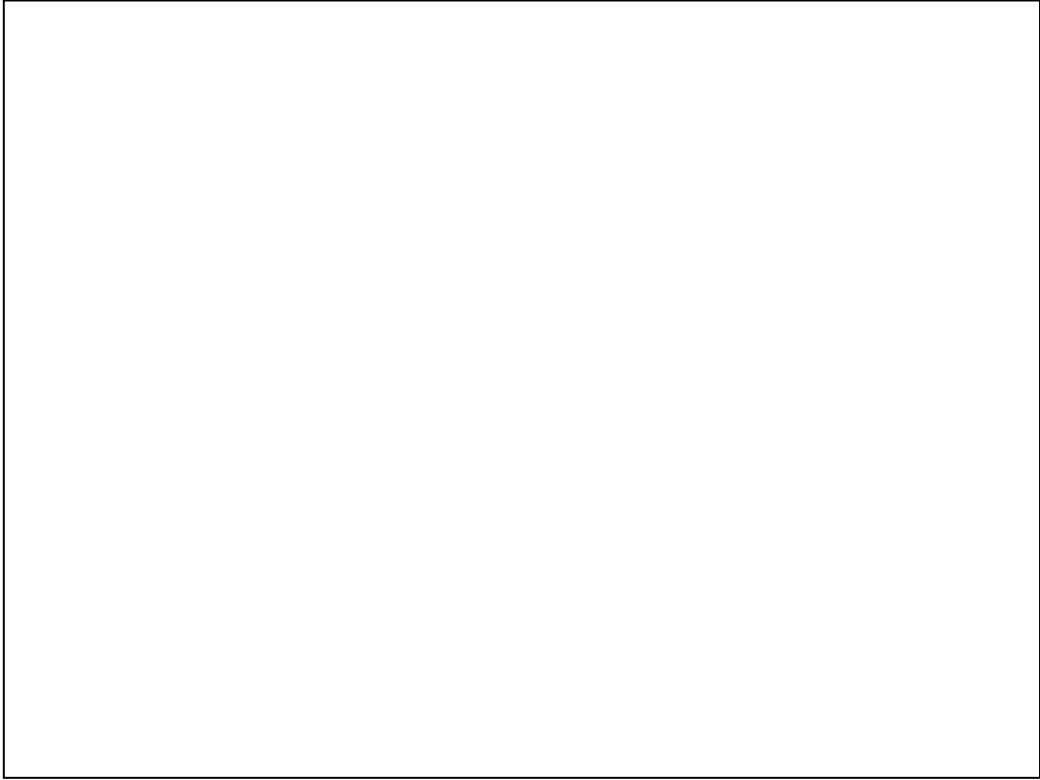


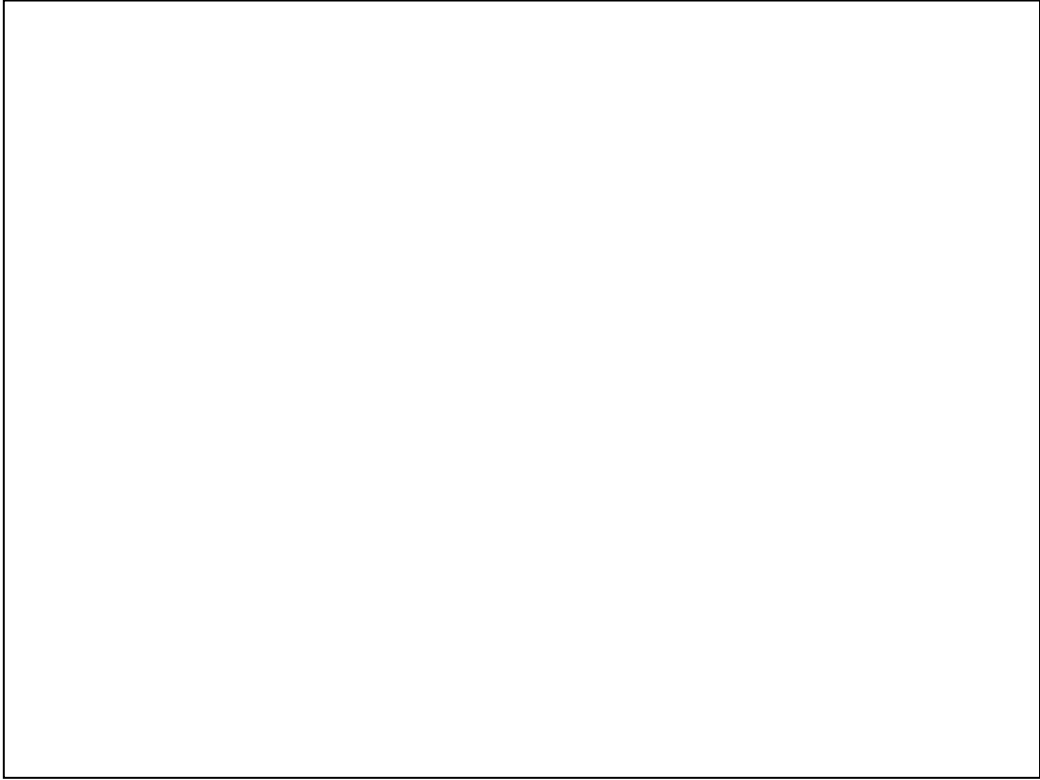


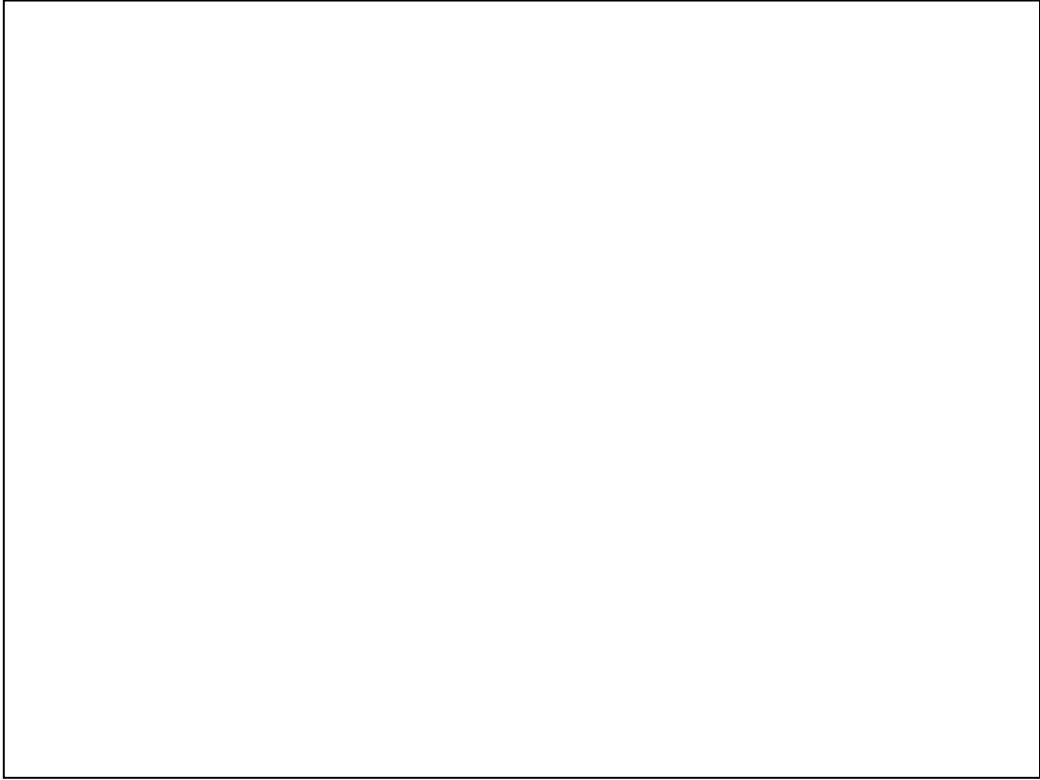








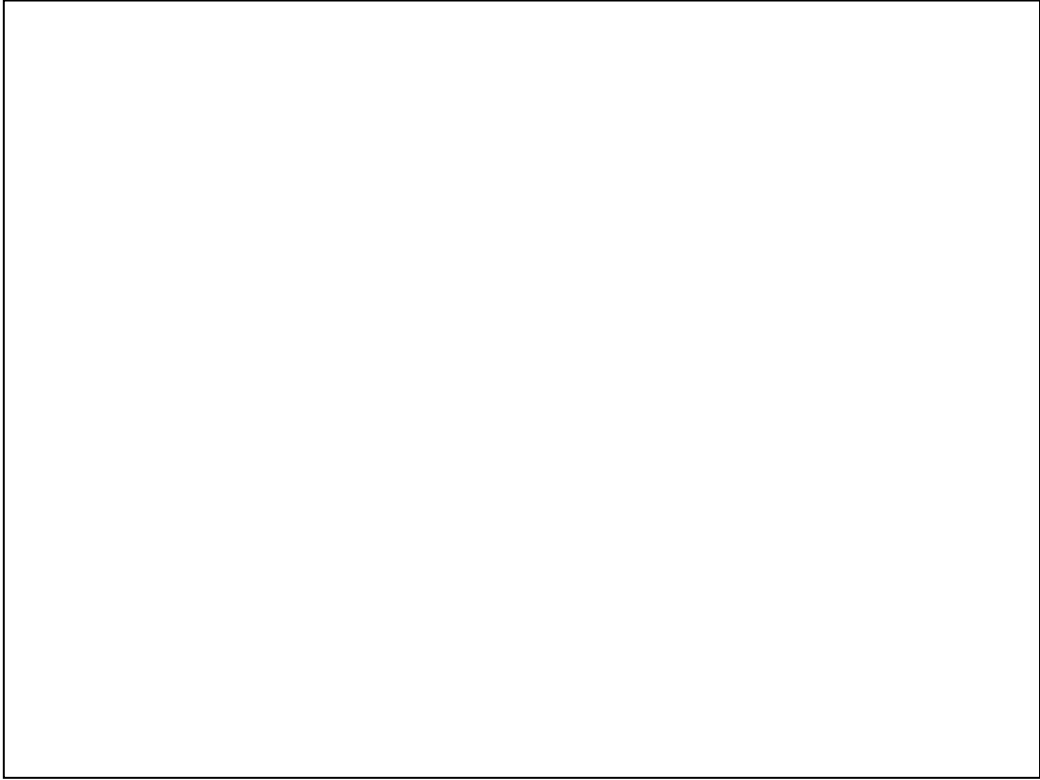


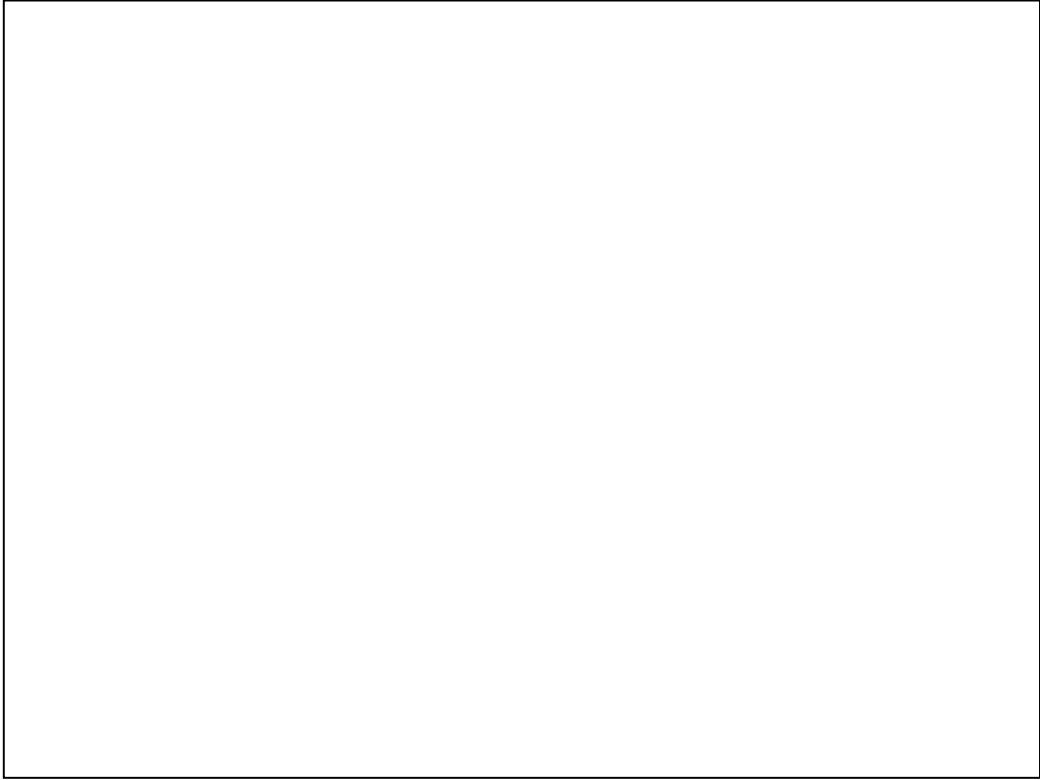


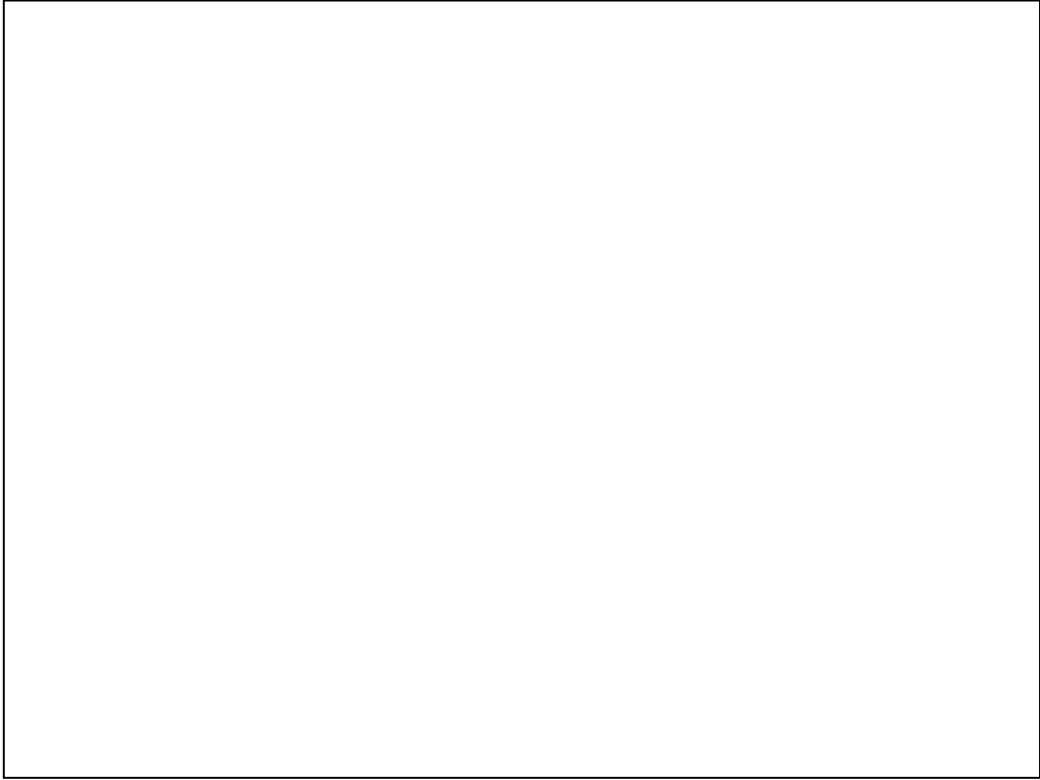


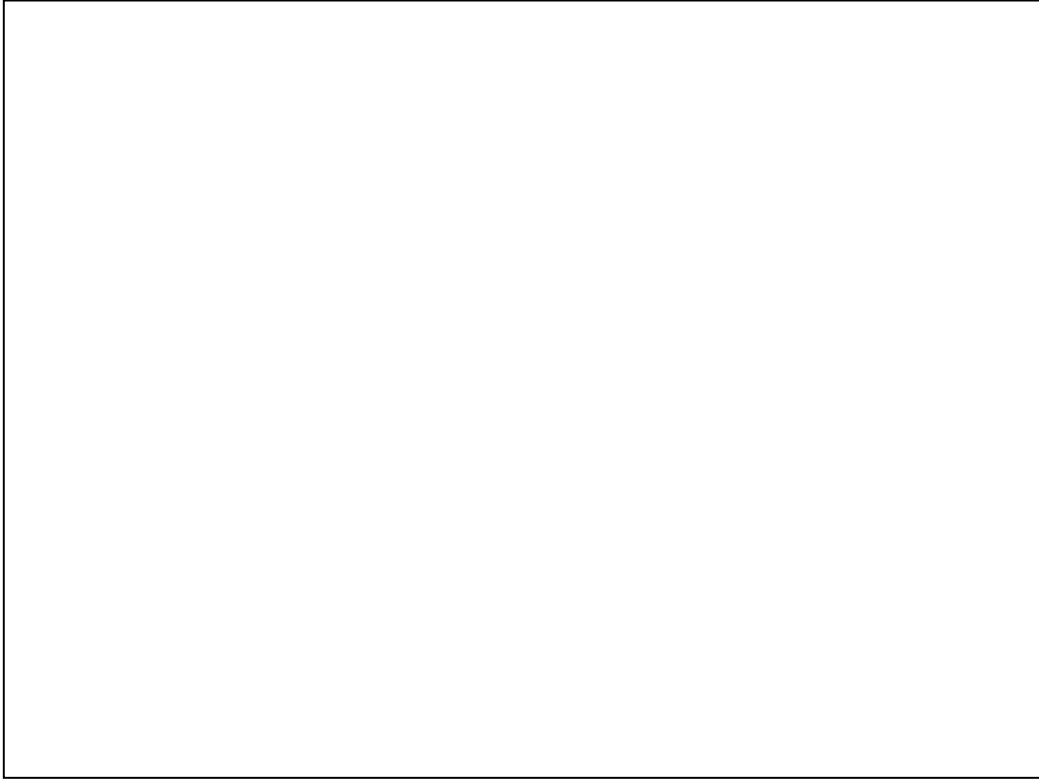












Case Study:

Case study: Software key protection

Long ago in a company far, far away... We needed to use commercial language crypto tools

What we found:

- Implemented FIPS 46 – DES in ECB only
- No protection against weak keys
- No provision for PRNG, seed generation, LFSRs
- No ability to securely erase keys/sub-keys
- Key protection methodology was a Ceaser cipher
- Only hash available was SHA
- No support for key management functions

Several years ago I was working on a project that required software based key storage.

So, we looked at the commercial language library crypto that was available, and discovered:

No crypto debug mode

Case study: Software key protection

What we found:

- Very difficult to use keys that weren't alphabetic
- No salt or seed support without hand-rolling
- Difficult to perform XOR functions
- Very poorly documented – Largely by sample code
- Several weaknesses in sample code
- Sample code encourage poor crypto practices
- Search engine found wide use of sample code
- No real x.509 support
- Touted by vendor as strong cryptography

Several years ago I was working on a project that required software based key storage.

So, we looked at the commercial language library crypto that was available, and discovered:

No crypto debug mode

